1. Introduction

A challenging problem for Artificial Intelligence (AI) is the modeling and application of abstract, open-textured rules that derive from real-world domains. An abstract, open-textured rule focuses on the essential qualities of a situation but must be interpreted in order for it to be directly applied to specific facts, objects, or instances. In domains that use these abstract rules, there are typically no available intermediate rules or concepts to connect the abstract rules to specific fact situations. Thus, there is an abstraction gap between the abstract rules and the concrete fact situations to which they apply. To bridge this gap – that is, to apply the abstract rules to the facts – one must draw upon additional knowledge and reasoning processes that are not specified in the domain rules. This dissertation focuses on a set of reasoning techniques, which I call *operationalization techniques*, that were observed to bridge the abstraction gap in one domain.

A number of real-world domains exhibit the abstraction gap between rules and specific facts, including legal reasoning and ethical reasoning. Consider, for instance, the following problem that confronted professional aerospace engineer Roger Boisjoly in 1986.

On January 28, 1986, the Space Shuttle Challenger was scheduled for lift-off. The evening before the launch, Roger Boisjoly recommended that the mission be postponed. He believed that the O-rings, part of the sealing mechanism in the booster rockets, were suspect and could possibly fail in the cold 26-degree temperature predicted for lift-off. Technical evidence was inconclusive, but there appeared to be a correlation between temperature and the capability of the O-rings to properly seal. As a professional engineer, Boisjoly believed that his duty was clear, as expressed in the National Society of Professional Engineers (NSPE) Code I.1., "Engineers shall hold paramount the safety, health, and welfare of the public in the performance of their professional duties." [NSPE, 1996] The public, in this case, was the Challenger astronauts and the schoolteacher Christa McAuliffe, and Boisjoly believed their lives could be in danger. On the other hand, NASA needed to impress the public with a successful mission. The space program had run into lean times in the 1980s. In addition, Boisjoly's employer, Morton Thiokol, was competing for a new contract with NASA. Thus, Boisjoly was also obligated to consider his client's and employer's interests, as expressed in NSPE Code I.4., "Engineers shall act in professional matters for each employer or client as faithful agents or trustees." Given the inconclusive nature of the evidence, Boisjoly could have given preference to his obligation to his client and his employer and given his approval for lift-off. However, Boisjoly stuck to his guns and vehemently and vocally opposed the launch. His recommendation was overruled by his employer, and Boisjoly, now obligated by NSPE Code II.1.a. " … If their professional judgment is overruled under circumstances where the safety, health, property or welfare of the public are endangered, they shall notify their employer or client and such other authority as may be appropriate," continued to try to persuade management to abide by his recommendation. The next day, Challenger was launched as scheduled but exploded just over a minute into its flight, killing the six astronauts and McAuliffe [Paraphrased from Harris *et al*., 1999, p. 4-6].

In retrospect, Boisjoly seems to have fulfilled his professional obligations as an engineer, although the crew of the Challenger, sadly, did not benefit from it. It may even appear that Boisjoly did the only thing that he could or should have done under the circumstances.

In general, the resolution of this type of engineering ethics dilemma, before the fact, is by no means clear or straightforward. To address such dilemmas the engineer must apply abstract, open-textured rules, i.e., *ethical codes*, to very specific fact situations. Doing this cannot involve a deductive process of applying the rules to the facts. The facts and rules are too disparate in specificity and the rules too imprecise for this to be possible. Rather, the engineer must employ heuristic techniques to address such situations. For instance, Boisjoly needed to interpret code I.1., the rule obligating him to protect public safety, within the context of the specific facts in order to determine whether it truly applied. Was the O-ring evidence strong enough to conclude that the "safety, health, and welfare" of the public was in danger? And could the astronauts and schoolteacher be considered the "public," given the fact that they had voluntarily accepted a professional role in which danger was inherent? Boisjoly's duty to his employer and client, as expressed in code I.4., was also uncertain. Was it clear that Boisjoly's duty to act as a "faithful agent or trustee" extended to a situation, such as this one, in which his technical judgment was compromised? And it could easily be argued (although in hindsight) that Boisjoly *was* acting as a faithful agent; the resulting disaster surely did more to harm his employer's and client's interests than canceling the launch would have done. Code II.1.a. is also subject to interpretation in these circumstances. Although Boisjoly clearly disputed his employer's position when his judgment was overruled, did he "notify … other authority as may be appropriate"? One might claim that he had an ethical obligation to escalate the situation to authorities that were independent of his company and client.

Even if Boisjoly had recognized that the situation caused a conflict between his obligation to the public and his obligation to his employer and client, his dilemma was not any easier to resolve. He would then have needed to decide whether one rule overrides another in the specific circumstances. Although the obligation to the public is "paramount" for engineers, meaning that it takes precedence over all other obligations, Boisjoly would have needed to conclude that the O-ring evidence strongly supported his belief the public was in danger. In this sense, Boisjoly had to infer or hypothesize unknown or unclear facts.

In contrast to this decision-making process, consider the following hypothetical syllogism of medical facts and rules:

1.	Lab-Test-X is positive.	(Facts of a case)
2.	If Lab-Test-X is positive then Organism-Y is present.	(Axiom-1)
3.	Organism-Y is present.	(*Modus ponens*, 1, 2)
4.	If Organism-Y is present then Patient has Condition-Z.	(Axiom-2)
5.	Patient has Condition-Z.	(*Modus ponens*, 3, 4)

Given the above axioms, or "rules," together with the facts of the case, i.e., that Lab-Test-X is positive, it is straightforward to chain the axioms together and arrive at the conclusion that the Patient has Condition-Z. All that is required are two applications of the inference rule *modus ponens*. The key is that an intermediate axiom exists (i.e., Axiom-1) that operates directly on the base facts to reach an intermediate conclusion. Given the intermediate conclusion, it is then possible to apply a more-abstract rule (i.e. Axiom-2) to reach a final conclusion. In essence, the problem is simplified because, in this as in many domains, axioms exist that support a process of deductive inference from specific facts to intermediate conclusions to final conclusions.

Of course, this particular syllogism is simple. Logic problems and AI problem solvers are usually far more complex than this. On the other hand, an assumption underlying many AI systems is that intermediate rules or concepts, perhaps based on uncertainty or probability, are available in the domain of interest.

Many AI systems reason by using formal deduction or rule chaining in which specific facts lead to conclusions through a series of intermediate hypotheses, conclusions, and rules. For instance, the landmark medical expert system, Mycin [Davis *et al*., 1977], employs rules that use symptoms, test results, and observations – all relatively specific, concrete forms of knowledge – as the conditions for asserting intermediate states and conclusions. The intermediate states can then be acted upon by more-abstract rules that lead to final diagnoses and suggested therapies. Although the degree of certainty regarding medical facts and diagnoses is imprecise in Mycin, as in many expert systems, the medical domain lends itself to a rule-chaining process in which a series of increasingly more-abstract rules are applied to specific facts and intermediate conclusions. PRODIGY [Minton *et al*., 1990; Veloso, 1992], a domain-independent problem solver that employs machine learning and case-based reasoning among other AI techniques, is another example of an AI system that assumes a domain of application in which a form of deduction is possible [Minton, 1988, p. 69]. Although the program employs general problem solving techniques in performing its planning tasks, the operators, goals, subgoals, and state descriptions of PRODIGY are built (or induced) from specific, well-defined domain knowledge and deduction underlies PRODIGY's operation.

CLASP [Yen *et al*., 1991], an AI programming environment that applies term subsumption to rule-based programming, is another example of a system that stresses formal deduction. Term subsumption refers to the process of deducing whether one term (or concept) subsumes, or is more general than, another. Such a formal process can only be successful in a domain in which the most abstractly defined concepts are connected to the basic facts via intermediate concepts or rules[1].

In summary, one of the key underpinnings of AI systems such as Mycin, PRODIGY, and CLASP is that they all operate in and depend on domains in which intermediate concepts or rules connect concrete facts to abstract or strategic rules or concepts. Furthermore, for these systems to be successful, the requisite intermediate knowledge must be directly provided or derivable from knowledge about the domains of interest. In other words, the domains addressed by these systems exhibit a strongly deductive or rule-chaining flavor.

On the other hand, there are many real-world domains, for instance the law, ethics, and political decision making, in which deduction or rule chaining is not possible because intermediate-level domain rules and concepts are not available. In these domains, rules are available almost exclusively at an abstract level, in the form of principles, policies, or laws. The difficulty in addressing and forming arguments in such domains using formal logic has long been recognized [Toulmin, 1958], and some practitioners in AI, particularly those interested in the legal domain, have also grappled with this issue. As pointed out by Ashley, "The legal domain is harder to model than mathematical or scientific domains because deductive logic, one of the computer scientist's primary tools, does not work in it." [1990, p. 2]

These *weak analytic domains* are typically characterized by the following attributes. First, the given "rules" (i.e., laws, codes, or principles) are available almost exclusively at a highly conceptual, abstract level. This means that the rules may contain *open-textured* terms [Twining and Miers, 1976; Gardner, 1987]. That is, conditions, premises, or clauses that are not precise or that cover a wide range of specific facts, or are highly subject to interpretation and may even have different meanings in different contexts. This contrasts with more-specific knowledge, such as

---

[1] Unless, of course, the domain is so shallow that all concepts may be formally and directly defined in terms of instances. Such a domain would certainly not be very interesting.

explicit goals, physical states, visual observables, and test results, modeled in the intermediate knowledge and rules of systems such as Mycin, PRODIGY, and CLASP. A second characteristic of weak analytic domains, closely related to the first characteristic, yet distinct, is that the actions prescribed by the given rules, i.e., the rules' conclusions, may also be abstract. Thus, even if one is able to determine that a particular rule applies to a given fact situation, the rule's consequent recommendation may be difficult to execute because it is highly conceptual or vague. For instance, how does one determine the action prescribed by NSPE code I.1., in which professional engineers are urged to "hold paramount" the safety, health, and welfare of the public? The prescribed action is clearly tied to the specific circumstances of a case to which it is applied. Third, abstract rules often conflict with one another in particular situations with no deductive or formal means of arbitrating such conflicts. That is, more than one rule may appear to apply to a given fact situation, but neither the abstract rules nor the general knowledge of the domain provide clear resolution.

1.1. Problem Solving and Analysis in Weak Analytic Domains

A problem solving approach that is often used to address problems in the weak analytic domains is *case-based reasoning* (CBR), the use of precedents, or past cases, in guiding decision making, constructing arguments, or suggesting actions in a new dilemma or problem. For instance, reasoning or arguing from precedents is an established technique in Anglo-American law, formalized by the rule of *stare decisis* ("let stand what has been settled"). This doctrine specifies that a conclusion reached by a court is "binding on the same court (or an inferior court) in a similar case" [Berman and Greiner, 1980]. AI researchers have followed this evidence and have developed a number of approaches that utilize, at least in part, case-based reasoning [Ashley, 1990; Branting, 1991; Rissland and Skalak, 1991; Rissland *et al.*, 1996].

Problem solving and analysis in engineering ethics also rely heavily on case-based reasoning. After performing an empirical analysis of hundreds of ethics cases published by a professional engineering society and decided by an ethical review board, I concluded that case-based reasoning plays a prominent role in analyzing ethics problems and justifying decisions [McLaren and Ashley, 1998]. Experienced ethical reasoners in engineering ethics appear to employ precedence as a tool to focus and guide their thinking in the analysis of new cases. More generally, my study revealed that the review board systematically applied operationalization techniques to bridge the gap between abstract principles and specific fact situations. The operationalization techniques involve the application of structured yet largely implicit heuristic techniques to the task of analyzing and justifying decisions in new cases, in particular, instantiating principles and past cases by connecting them to clusters of questioned and critical facts, hypothesizing unstated facts, updating principles over time, arbitrating between competing principles, grouping principles or past cases, defining or elaborating issues and principles from past cases, and reusing specific applications of these techniques from previous analyses. My analysis also indicated that temporal knowledge plays a role in the retrieval and analysis of cases. For example, in many instances the knowledge that one event occurred before, during, or after another event clearly influenced the board's analysis of a present case and the analogies it made to past cases.

1.2. Operationalizing Principles and Cases: An Overview of the Techniques

The techniques uncovered by my empirical analysis are listed in Figure 1-1. The application of a subset of these techniques can be identified in the analysis of every case. I call the techniques operationalization techniques because each serves to make a code or a case an operational component of an argument put forth by the ethical review board. This dissertation presents a computational model for implementing and reusing these techniques.

The gap between the abstract ethics codes and the cases' detailed factual circumstances presents difficulties for both ethical reasoners and modelers of ethical reasoning. With these techniques, the board justifies and explains the applicability and significance of relevant code provisions and past case decisions to the analysis of the current problem. These operationalizations provide constraints on and a context for the future use of codes and cases in analyzing new cases. In effect, the operationalizations make up for the lack of intermediate domain rules.

Mostow was the first to propose the notion of operationalization [1983]. His use of operationalization, however, focused on the comparatively well-defined domain of card playing. The engineering ethics domain is far less structured and much more complex.

Code Operationalizations

- ***Code Instantiation***. In the context of a case, a code's conditions may be implicitly interpreted and connected to selected facts of a case. In essence, the board instantiates the code when interpreting it by linking its more-abstract formulation to the selected specific facts of the case. To facilitate instantiation the board may employ one of the following techniques:
  - ***Rephrase the Code***. The board may rephrase or reword a code in order to more clearly align and link it with the specific circumstances of the case in question.
  - ***Define the Terms of the Code***. Specific terms or the language of a code may be ambiguous or ill-defined. The board may first define such terms before linking the code to the case facts.
  - ***Make the Code More Specific***. A typical tactic employed by the board is to narrow the conditions of a code. Instantiation is directly supportive of this technique, as each set of links to case facts provides a single, specific example of an application of the code.
- ***Apply a Hypothetical to a Code***. The board proposes hypotheticals to help establish the range of application of a principle. Hypotheticals are also used to provide unknown, yet likely, facts.
- ***Rewrite a Code***. A controversial or ill-defined principle may explicitly change or be reinterpreted over time by the board. For instance, the wording of a code may conflict with changing cultural norms and the board may recommend that it be formally changed.
- ***Define the Superior Code***. Sometimes two (or more) principles conflict with one another in the context of a particular case. The board identifies such an event and explains the underlying reasons for the "superiority" of one principle over others.
- ***Group Codes***. The board occasionally cites related codes as a group because they may all apply, at least abstractly, to the case in question and thus provide greater force to a particular conclusion. In effect, the set of codes becomes a "single code" in the context of the case.

Case Operationalizations

- ***Case Instantiation***. The board frequently cites previous cases that may be instrumental in deciding the case in question. In effect, the board instantiates a case when it cites the case as a precedent. The cited case is linked to the selected facts of the current case that indicate similarity between the cases. Two types of Case Instantiations are employed:
  - ***Cite an Analogous Precedent***. The board cites an analogous precedent to argue that its conclusion should be followed in the case in question. That is, the precedent case "rules" the present case.
  - ***Cite a Distinguishing Precedent***. The board sometimes cites a precedent to argue that, although it has similarities to the present case, its conclusion should not be followed because of relevant differences between the precedent and the present case.
- ***Define or Elaborate a General Issue or Principle***. A previous case is sometimes cited because it discusses a relevant general issue or principle. In other words, the previous case may not be factually or relevantly similar, but the analysis of the case raises issues relevant to the present case.
- ***Group Cases***. Cases are sometimes cited in tandem for a single illustrative purpose or to apply greater force to a conclusion. In effect, the set of cases acts as a "single case" in the context of a specific case.
- ***Reuse an Operationalization***. A past case may be cited in order to reuse a previous operationalization. For instance, a past case (or cases) may define terms of a code that are relevant to a new case. In effect, this technique reuses pieces of operational reasoning employed in analyses of past cases.

Figure 1-1: The Operationalization Techniques employed by the NSPE BER

1.3. The Dissertation Project in a Nutshell

My primary goal in this dissertation project was to explore and analyze the relationship between abstract rules and concrete facts in the domain of engineering ethics and to implement the above-mentioned problem solving techniques in a computational model. The program that resulted, SIROCCO (System for Intelligent Retrieval of Operationalized Cases and COdes), is designed to retrieve decided cases and previously applied principles in order to frame analyses of new engineering ethics cases. SIROCCO is not intended to reach conclusions for the new cases but, rather, to identify relevant information for the analysis of the cases. The ultimate goal (but not part of the dissertation project) is to deliver SIROCCO to engineers as a tool for improving access to an on-line resource of decisions in engineering ethics cases. Another long-term goal is to incorporate SIROCCO as a retrieval component in an intelligent tutoring system for engineering students.

Essentially, SIROCCO *operationalizes principles and cases* in the domain of engineering ethics. By "operationalizing" I mean that SIROCCO embodies a representation of the operationalizing information assigned by experts to a principle or case. SIROCCO makes this information computationally available for purposes of relevance assessment and explanation. Given an engineering scenario that raises an ethical question (or questions), SIROCCO predicts the facts, principles ,and past cases that the board of ethical review would regard as important in the analysis of the ethical problem. The model's predictions are based on the board's code-based analyses of past cases. All of the operationalization techniques found in Figure 1-1 are implemented, in some form in SIROCCO's architecture. SIROCCO is also capable of revealing the reasons underlying its predictions, both in terms of the program's internal structures and in terms more relevant to an end user.

SIROCCO's approach is based on an empirical study, briefly introduced above, of how experienced professional engineers use principles and specific fact situations to resolve ethical issues and to justify their decisions. In particular, I have studied published opinions of the National Society of Professional Engineers' Board of Ethical Review (NSPE BER) and its code of ethics [1958-1998]. Over a period of just over 40 years, the NSPE BER, composed of five to seven professional engineers annually, has written extensive explanations of how and why codes apply or do not apply to approximately 400 particular fact situations. The NSPE BER's analyses are published in hardcopy and on the Internet as an educational service to the entire engineering community. The NSPE's code of ethics, consisting of 75 individual codes, provides engineers with guidance on issues such as public safety, misrepresentation in advertising, conflicts of interest, and confidentiality. The NSPE BER database provides a valuable record of how engineering ethics experts believe engineering ethics codes apply to practical situations.

As part of the project, I also performed a controlled experiment with SIROCCO. The experiment compared the retrieval performance of SIROCCO with that of five other computational models, including an information retrieval system that uses a vector space model of the case and code texts and a version of SIROCCO that only partially (and in only a very weak sense) employs operationalization techniques. The experimental results show that SIROCCO outperformed all of the baseline measures in the retrieval task. Although some of the operationalization techniques were not tested in the experiment, in particular the ones that did not seem to support retrieval and that produced qualitative linguistic suggestions that were not easily verifiable, the techniques *Code Instantiation*, *Group Codes*, *Case Instantiation*, *Reuse an Operationalization*, and *Group Cases* are shown to improve the retrieval performance of the system. On the other hand, the experimental results also show that SIROCCO's temporal knowledge did not improve performance in an objectively verifiable way. I provide both qualitative and quantitative analysis of these results in the dissertation.

This research is related to AI and Law research, but it pioneers work in a domain with a less-explicit model of argumentation, and it addresses a wider breadth of domain coverage than predecessor AI and Law systems. This project makes a contribution to the literature of *interpretive case-based reasoning* [Kolodner, 1993, p. 86-92] by investigating the connection between abstract rules and concrete facts, by introducing and cataloging a set of techniques for case-based retrieval and analysis, and by testing the feasibility of using detailed, chronological facts to represent cases.

1.3.1. Statement of the Primary Thesis

SIROCCO's operationalization techniques allow it to make accurate predictions of the facts, principles, and past cases that are relevant to the analysis of new cases. Although all of the operationalization techniques have been implemented within SIROCCO, only a subset of the techniques is directly verifiable in an objective experiment. Therefore, the primary claim of the thesis is somewhat narrower than the general claim. In particular:

> Past decisions in engineering ethics cases use the following techniques to operationalize ethical principles and cases:
>
> 1. Instantiating and interpreting principles by linking them to clusters of questioned and critical facts

2. Hypothesizing unstated facts
3. Updating the principles over time
4. Arbitrating between competing principles
5. Grouping principles
6. Instantiating cases as precedents by linking them to clusters of questioned and critical facts
7. Defining or elaborating issues and principles from past cases
8. Grouping cases
9. Reusing specific applications of any of the above techniques from previous analyses

> A computational model of a critical subset of these operationalizations, in particular numbers 1, 5, 6, 8, and 9, can make accurate predictions of the principles and past cases that are likely to be important in the analysis of new cases.

The subset of operationalizations tested in the dissertation, especially the *Code Instantiations* and *Case Instantiations*, are critical because they are ubiquitous (yet implicit) in the review board's analyses.

1.3.2. Statement of the Secondary Thesis

The secondary claim of this dissertation is that SIROCCO's temporal knowledge assists it in making accurate predictions of the principles and past cases that are relevant to the analysis of new cases. More specifically:

> Temporal knowledge is integral in the retrieval and analysis of principles and past cases. A computational model incorporating temporal knowledge will improve the accuracy of the model's predictions and analysis.

1.3.3. An Introduction to the Computational Model: SIROCCO

SIROCCO's goal, given a new case to analyze, is to provide the basic information with which a human reasoner could answer the question posed in the case and then build an argument or rationale for that conclusion. The program utilizes the operationalization techniques in various ways in the service of this goal, including supporting the retrieval of codes and past cases, focusing the analysis portion of the program on the most fruitful codes and cases cited by the past cases, and providing additional suggestions not directly reflected in the suggested codes and cases produced by the program.

An example of SIROCCO's output is shown in Figure 1-2. The facts of the input case[2] and the question raised by that case are first displayed. The case involves an engineer who discovers serious safety problems in a building but does not report the safety problems to anyone except the client, because his client, the building owner, requests confidentiality. The question raised is whether it was ethical for the engineer to give preference to the client's confidentiality over the public's safety.

SIROCCO's output consists of various pieces of information, derived by the application of operationalization techniques, that could support a human in reasoning about and arguing this case: (1) a list of possibly relevant codes, (2) a list of possibly relevant cases, and (3) a list of additional suggestions. Although not illustrated in Figure 1-2, SIROCCO is also capable of displaying its reasons for selecting the suggested codes and cases; this capability is briefly discussed later in this section, and an excerpt is shown in Figure 1-3. An extended and detailed discussion of its operation is provided in Chapter 3.

SIROCCO accepts new cases in a detailed case-representation language, designed as part of the dissertation project and called the *Ethics Transcription Language* (ETL) (see Appendix Section A.1). The language represents the actions and events of a scenario as an ordered list (i.e., a *Fact Chronology*) of individual sentences (i.e., *Facts*), each consisting of (1) *Actors and objects*, instances of general actors and objects which appear in the scenario, (2) a *Fact Primitive,* the action or event in which the actor and/or object instances participated, and (3) a *Time Qualifier,* a temporal relation that specifies how a Fact relates to other Facts in time. A predefined set of general Actor, Object, Fact Primitive, and Time Qualifier types are used in the representation. At least one Fact in the Fact Chronology is designated as the *Questioned Fact*; this is the action or event corresponding to the ethical question raised in the scenario.

[2] Cases are named using a three part number: YY-Cn-Qn where YY is a two-digit representation of the year the case was decided, Cn is the case number within that year, and Qn is the question number of that case. Thus, 90-5-1 is a case corresponding to the first question of the fifth case decided in 1990 by the review board. Cases also have mnemonic titles; these will occasionally be used in the dissertation as well.

****
*** SIROCCO is analyzing Case 90-5-1
****

Facts:
Tenants of an apartment building sue the owner to force him to repair many defects in the building that affect the quality of use. The owner's attorney hires Engineer A to inspect the building and give expert testimony in support of the owner. Engineer A discovers serious structural defects in the building, which he believes constitute an immediate threat to the safety of the tenants. The tenants' suit has not mentioned these safety-related defects. Upon reporting the findings to the attorney, Engineer A is told he must maintain this information as confidential as it is part of a lawsuit. Engineer A complies with the request of the attorney.

Question:
Was it ethical for Engineer A to conceal his knowledge of the safety-related defects in view of the fact that it was an attorney who told him he was legally bound to maintain confidentiality?

****
*** SIROCCO has the following suggestions for evaluating
*** '90-5-1: Failure To Report Information Affecting Public Safety'
****

*** ***Possibly Relevant Codes:***

I-4: Act as a Faithful Agent or Trustee
III-4: Do not Disclose Confidential Information Without Consent
I-1: Safety, Health, and Welfare of Public is Paramount
II-1-A: Primary Obligation is to Protect Public (Notify Authority if Judgment is Overruled).
III-1-B: Advise Client or Employer When a Project Will Not Be Successful
III-1: Be Guided by Highest Standards of Integrity
II-1-C: Do not Reveal Confidential Information Without Consent
III-2-B: Do not Complete or Sign Documents that are not Safe for Public
II-1-E: Report Alleged Code Violations
II-5-A: Do not Falsify or Misrepresent Qualifications

*** ***Possibly Relevant Cases:***

76-4-1: Public Welfare - Knowledge of Information Damaging to Client's Interest
89-7-1: Duty To Report Safety Violations
84-5-1: Engineer's Recommendation For Full-Time, On-Site Project Representative

*** ***Additional Suggestions:***

o The codes II-1-A ('Primary Obligation is to Protect Public (Notify Authority if Judgment is Overruled).') and I-1 ('Safety, Health, and Welfare of Public is Paramount') may override codes III-4 ('Do not Disclose Confidential Information Without Consent'), I-4 ('Act as a Faithful Agent or Trustee'), and III-1 ('Be Guided by Highest Standards of Integrity') in this case. See case 76-4-1 for an example of this type of code conflict and resolution.

…

o The case 67-10-1 was cited by 76-4-1 to highlight or elaborate a general principle or common scenario. Since 76-4-1 has been suggested as possibly relevant to the present case, its cited case may also be relevant. Check whether the general scenario of the cited case is relevant to the present case:

'Engineer is involved in a professional situation in which the public welfare is at stake'

Figure 1-2: An Excerpt of SIROCCO's Output for Case 90-5-1

Cases stored in SIROCCO's case base are represented in the *Extended Ethics Transcription Language* (EETL) (see Appendix Section A.2), of which ETL is a subset. EETL provides all of the scenario representation elements described above but, in addition, provides a template and standard components for representing the analysis of a case, including its conclusion (i.e., ethical, unethical, or undecided), the protagonist whose action is questioned, the ethical review board's general argument structure, a linking of the critical facts of the case to the citations support by those facts, and a series of other details, such as which citations are more important and which citations are grouped together. In essence, EETL models arguments as a series of code and case operationalizations, some of which support the conclusion reached in the scenario, some of which conflict with the conclusion, and some of which simply provide relevant background information.

SIROCCO's case base consists of a subset of the NSPE BER cases that were analyzed during the empirical study. A total of 184 *foundational cases*, covering 135 fact situations[3] and culled from the grand total of 475 cases decided by the NSPE BER between 1958 and 1992, are included in SIROCCO's case base and were used to design, implement, and refine the program. All 75 of the NSPE BER ethics code provisions are encoded in SIROCCO. The goal was to include all of the cases covering a reasonable number of important ethics topics or codes (e.g., public safety) but also to provide some (minimal) coverage of cases outside of those topics. As such, the foundational case base includes all 135 of the transcribable cases that cite at least one code related to at least one of the following *Selected Topics*:[4] public safety, confidential information, duty to employer, credit for engineering work, proprietary interests, and honesty in reports and public statements. The foundational case base also includes 49 cases which do not cite any codes from the Selected Topics. Ethical topics and issues outside the scope of the Selected Topics group will be referred to as the *Non-Selected Topics*. A full listing of the Selected Topics and a partial listing of the Non-Selected Topics represented in the NSPE BER cases and in SIROCCO's case base are shown in Table 1-1.

SIROCCO's relatively broad domain coverage represents an important distinction and advancement in comparison to previous interpretive CBR systems such as HYPO [Ashley, 1990] and CATO [Aleven, 1997], which handled trade secrets cases exclusively; GREBE [Branting, 1991], which reasoned only about workers' compensation cases; CABARET [Rissland and

[3] A fact situation is defined by a single Fact Chronology . A case is defined as a Fact Chronology together with a chosen Questioned Fact from that chronology. Thus, if the fact situation raises multiple ethical questions, one fact situation can contain multiple cases.

[4] There are actually more than 135 cases out of the total of 475 that cite at least one code from the Selected Topics . However, some cases were excluded because they could not be accurately encoded into the Ethics Transcription Language.

Skalak, 1991], which processed only home-office tax deduction cases; and BankXX [Rissland *et al.*, 1996], which handled only Chapter 13 personal bankruptcy cases[5].

Table 1-1: Topics Addressed within the NSPE BER Cases and Represented in SIROCCO's Case Base

Category	Types of Ethical Topics in the Category
Selected Topics	Public safety, confidential information, duty to employer, credit for engineering work, proprietary interests, and honesty in reports and public statements.
Non-Selected Topics	Conflicts of interest, honesty in advertising, misrepresentation and omission of facts, criticizing other engineers, competence and qualifications, …

The foundational cases were represented with the support of a case-acquisition web site and were transcribed into EETL by a total of 12 independent case enterers. The web site contains a Participant's Guide with instructions on how to transcribe ethics cases into EETL, a reference shelf of useful materials, including the full vocabulary of EETL, and an example set of 47 transcribed fact situations. The case enterers submit each case as a structured set of tables; those tables are subsequently translated into SIROCCO knowledge structures by a PERL program. Virtually all of the foundational cases provided by the case enterers were at least slightly amended, to correct such things as poorly constructed Facts and incorrect usage of Fact Primitives. However, in most instances, the critical aspects of the representations provided by the case enterers were not altered.

SIROCCO takes the following approach. Given a case base of engineering ethics dilemmas, transcribed into EETL, the program:

1. Accepts a new fact situation and a question raised by that fact situation (together the fact situation and question are known as the *target case*), transcribed into ETL.

2. Retrieves and matches cases in the case base (i.e., the *source cases*) using a two-stage algorithm. Stage 1 is a fast and efficient, but somewhat coarse, retrieval based on matching, at various levels of abstraction, the Fact Primitives of the target case to the Fact Primitives of all of the source cases. Stage 2 is a more-expensive structural mapping that employs A* search and focuses on the most critical facts and on the chronology of facts.

[5] Note that Aleven claims that CATO "is not specific to trade secrets law" [1997, p. 41]. This claim might also be made about all of the other systems listed here. However, it is certainly true that none of these other systems actually *demonstrated* a wider domain coverage, as is done with SIROCCO in this dissertation.

3. Frames an analysis of the target case by suggesting codes, past cases, and operationalizations that may be relevant to the new case. To harvest its suggestions, the program applies various selection heuristics, strongly influenced by the operationalization techniques, to the results of the retrieval step.

The program begins, in step 1, by accepting a target case that has been transcribed into ETL. All of the cases in SIROCCO's case base (i.e., the source cases) are defined in EETL; these cases contain the analysis and operationalization information that could potentially be useful in the context of the target case.

The retrieval portion of the method (i.e., step 2) consists of a two-stage algorithm in which a computationally inexpensive initial stage retrieves a set of candidate source cases, followed by a more-expensive graph-mapping stage which evaluates the candidates in greater detail and selects a set of cases – and codes cited by those cases – that are most likely to be relevant to the input case. The first stage matches the Fact Primitives of the target case, and abstractions of those primitives at various predefined levels, to all of the cases in the case base by treating each case as a *content vector* and performing a weighted dot product calculation between the target and all of the source cases. Weights are applied to matches at the varying abstraction levels and to matches of a source case's critical and questioned facts. After the top *N* candidate source cases have been retrieved, with *N* being a user-specified parameter and the candidates have been sorted by descending dot product scores, the second stage employs a heuristic A* search for each target case/candidate source case pair, in order to find a structural mapping between the two. The case pairs are essentially treated as two sets of nodes and vertices, i.e., as graphs, and the program attempts to find a mapping of the Actors, Objects, Fact Primitives, internal fact phrases, and Time Qualifiers between the cases that satisfies structural consistency constraints (i.e., one-to-one mapping and parallel connectivity) [Gentner, 1983].

Both the accuracy and efficiency of the retrieval algorithm benefit from the use of *Code Instantiation* and *Case Instantiation* operationalizations defined by the case enterers, based on their interpretation of the board's rationale in citing a code provision or a case. An Instantiation relates a questioned fact, certain critical facts, and the temporal sequence of those facts to a citation. Instantiations help SIROCCO focus attention on the most important facts of a case. Instantiations improve the accuracy of Stage 1 by giving more weight to critical Fact Primitives. The Instantiations increase the match score (i.e., the dot product) of those source cases in which the most critical Fact Primitives were matched. Stage 2 is made both more efficient and more accurate because its structural mapping routine is focused on a subset of each source case's Fact

Chronology, meaning that search is reduced and the algorithm is able to focus on those facts that are most critical to the ethical evaluation, as defined by the Instantiations.

SIROCCO's retrieval algorithm is a novel amalgam of existing methods and new concepts. The two-stage algorithm and use of content vectors resembles MAC/FAC [Forbus *et al*., 1994]. However, SIROCCO extends this approach by employing operationalizations (i.e., *Code Instantiations* and *Case Instantiations*) to focus the search, by matching at multiple levels of abstraction, and by considering temporal relations in the structural mapping process. The idea for using A* search comes from GREBE [Branting, 1991].

The codes and cases chosen during the retrieval phase are passed to the Analysis phase (i.e., step 3) so that the final codes, cases, and suggestions can be selected and shown to the user. Three separate categories of heuristics are employed during this phase: *code-selection heuristics*, *case-selection heuristics*, and *additional suggestion heuristics*. For the code and case selection categories, SIROCCO accumulates evidence for each candidate code or case by trying to apply all of its heuristics for that category to the candidate. Example code-selection heuristics include: selecting codes that are cited with a frequency over a user-specified threshold in the top *N* cases and selecting codes that have Instantiations with a structural mapping score above a given threshold with at least a minimal number of critical facts. Example case-selection heuristics include: selecting cases that contain *Code Instantiations* that satisfied the code-selection heuristics and selecting cases that are grouped with other cases that have already been selected.

Candidates are then sorted by the number of heuristics that applied to each. A filtering mechanism identifies those candidates that appear to be the most promising. For codes, the filter is implemented by using code-collocation statistics[6]; a code is deleted from the candidate list if it has a lower number of applied heuristics than another code and if it does not collocate with that code above a certain threshold. For cases, several criteria are used, most importantly, a citation overlap function that deletes a candidate case if its code citations do not sufficiently overlap with SIROCCO's candidate codes for the target case. The codes that remain, after filtering, are those selected by SIROCCO for display to the user, as in the portion of SIROCCO's output in Figure 1-2 with the heading "Possibly Relevant Codes."

In the last stage of SIROCCO's analysis, the program attempts to apply additional suggestion heuristics to the final set of selected codes and cases. For instance, if SIROCCO finds a selected case in which the *Define the Superior Code* operationalization was applied, that is, a past case in which the ethics review board indicated that one code took precedence over another in that context, and the relevant codes have also been selected in the present case, SIROCCO suggests that this operationalization may apply to the current fact situation as well. This is precisely the type of suggestion found near the bottom of Figure 1-2, in which SIROCCO indicates that 76-4-1 is a past case in which public safety codes (i.e., II.1.a. and I.1.) "override" several codes, one of which deals with confidentiality (i.e., III.4.). Such a suggestion is potentially quite valuable, as it helps to put the cited cases in juxtaposition and highlights issues that should be considered by the user. SIROCCO is also capable of making other suggestions, by applying the operationalization techniques *Define or Elaborate a General Issue or Principle* (also shown in the analysis of 90-5-1 at the bottom of Figure 1-2), *Rewrite a Code*, and *Apply a Hypothetical to a Code*.

SIROCCO is also capable of explaining why it has suggested certain codes and cases as relevant to the target case. SIROCCO can explain its reasons in terms of the heuristics that were applied by the analysis phase or in terms of the mapping performed by Stage 2 between the Critical and Questioned Facts of source cases to the target case. An explanation excerpt for Case 90-5-1 is displayed in Figure 1-3. Recall that Code III.4. was suggested by SIROCCO as possibly relevant in Figure 1-2. In terms of the heuristics, this code was selected for a number of reasons: it was cited by 4 of the top 6 cases from Stage 1, the Critical Facts of two *Code Instantiations*, one from Case 76-4-1 and one from Case 89-7-1, map to the target case with high match scores, the Questioned Facts of those same two *Code Instantiations* also map to the target, and, finally, it is grouped with another highly rated code, I.4., in the two cases, 76-4-1 and 89-7-1.

Under the heading "Structural mapping Explanation," the excerpt also explains how the *Code Instantiation* of Case 76-4-1 maps to the target case. The Facts preceded by an asterisk are the Questioned Facts of the target and source, the Facts at the first indentation level are the Critical Facts of Case 76-4-1's *Code Instantiation*, and the Facts at the second indentation level are the target Facts that match to corresponding Facts of the *Code Instantiation*.

SIROCCO's capability to explain its reasoning is an important aspect of its functionality. This explanation capability offsets the not insignificant effort involved in representing the cases and provides a significant advantage over the five methods it is compared to.

---

[6] "Collocation" is defined as the percentage of time code x is cited when code y is cited.

III-4: Do not Disclose Confidential Information Without Consent

Heuristics Explanation:

----

o Cited by 4 of the 6 best surface matching cases.
o 100.0% match to 4 critical facts in case 89-7-1.
o 86.7% match to 3 critical facts in case 76-4-1.
o 100.0% match and ques. fact match (Source and Target: FACT-PRIMITIVE) in case 89-7-1.
o 86.7% match and ques. fact match (Source and Target: FACT-PRIMITIVE) in case 76-4-1.
o > 50.0% match in multiple cases: 89-7-1, 76-4-1.
o Grouped with code I-4 in case 89-7-1; Good match to critical facts.
o Grouped with code I-4 in case 76-4-1; Good match to ques. facts.

Structural mapping Explanation:

To Case 76-4-1 (Corresponding steps of Case 90-5-1 indented; * = Questioned Fact)

ENGINEER-DOE DISCOVERS-THAT ((FAILS-STANDARDS-…))

>>> ENGINEER-A DISCOVERS-THAT ((FAILS-STANDARDS-…))

ENGINEER-DOE REVIEWS-AND-ANALYZES DISCHARGE

>>> ENGINEER-A INSPECTS APARTMENT-BUILDING

*ENGINEER-DOE DOES-NOT-INFORM-THAT CONTROL-AUTHORITY …

*>>> ENGINEER-A DOES-NOT-INFORM-THAT ANYONE-ELSE …

…

Figure 1-3: An Excerpt of SIROCCO's Explanation of the Analysis of Case 90-5-1

It is useful to clarify here what SIROCCO is *not* intended to do. As can be seen by the output of the program in Figure 1-2, SIROCCO does not suggest a conclusion (i.e., ethical, unethical, or a qualified "middle" solution) or a course of action, nor does it present an argument, either pro or con, for a particular conclusion. Rather, the intent of the program is somewhat more modest (yet still very challenging): to retrieve information – codes, cases, and operationalizations – that are relevant in the analysis of the case. It is the duty of the human user to evaluate and apply the information retrieved by SIROCCO, i.e., to confirm the information's relevance, to develop an argument, and to come to a conclusion for a given case. The program is not designed to reach a conclusion for several reasons. First, reaching an ethical conclusion is the obligation of the human decision-maker. Even if I believed my model of ethical decision making were up to the task, having a computer program propose decisions oversimplifies the obligations of human beings and belittles their unique abilities. Second, although the fundamental argument components were identifiable in the NSPE case analyses (i.e., the operationalization techniques), a coherent, consistent, and structured argument strategy was not. Thus, having the program produce an argument based on observations of the NSPE case analyses did not appear to be feasible. Finally, producing relevant information, rather than an argument and a conclusion, is more in keeping with the ultimate aim of SIROCCO as a retrieval tool for engineers in accessing relevant ethical information and as a retrieval component in a tutoring system. As pointed out by Harris, Pritchard, and Rabins [1999, p. 22], studying engineering ethics cases and codes is important to stimulating the "moral imagination" but such study does not provide ready-made answers to complex ethical questions.

SIROCCO does not process the natural language descriptions of cases. Although the domain of engineering ethics is heavily linguistic, natural language processing capabilities in Artificial Intelligence are still fairly primitive. Instead, as discussed above, I devised a web site to facilitate untrained persons in transcribing cases into ETL and EETL. The web site provided a means of enforcing some measure of consistency across cases and also supported the execution of more objective experiments in which independent third-parties provided the data. Hopefully, however, in the not-too-distant future a tool will be developed for translating natural language into a structured form usable by a program, but this is clearly beyond the scope of this project.

Finally, SIROCCO does not explicitly implement all of the operationalization techniques. In other words, the program does not execute each technique as a distinct function, visible in the program's output. Rather, the program incorporates some of the techniques explicitly (such as those used to provide additional suggestions) and others more conceptually and implicitly. The way in which each technique is included in, or influences, SIROCCO's architecture is described in detail in Section 3.2.

1.3.4. A Summary of the SIROCCO Experiments

Experiments were performed with SIROCCO to test the primary and secondary hypotheses of the thesis. In particular, the experiments were designed (1) to test whether a core set of SIROCCO's operationalization techniques allow the program to make accurate predictions of the principles and past cases that are relevant to the analysis of new cases and (2) to test whether SIROCCO's temporal knowledge similarly supports accurate predictions. Also, several additional experiments were performed to test the efficiency and scalability of the computational model.

The data used for the SIROCCO experiments included the 184 foundational cases upon which SIROCCO was designed and developed together with a set of 58 *trial cases* that were decided later than the foundational cases. All of the trial cases were transcribed into EETL by two independent case enterers and the resulting case representations, unlike the foundational cases, were provided unaltered to SIROCCO for processing to ensure objectivity. The 58 trial cases were chosen from two pools of cases within a set of 77 cases decided by the NSPE BER between 1993 and 1998: 44 trial cases were chosen at random from 52 Selected Topics cases and 14 trial cases were chosen at random from 25 Non-Selected Topics cases. Two of the 44 original Selected Topics cases were rejected by one case enterer as not transcribable and these were replaced by two other Selected Topics cases chosen at random. Four additional cases were transcribed but not included in the trial set because they did not cite any codes that were cited a minimal number of times in the foundational set of cases.

Although the goal was not strictly to choose Selected Topics cases in the trial set at the same ratio as in the foundational set, the percentage of Selected Topics cases in the trial set (76%, 44 out of 58) was close to that of the foundational set (73%, 135 out of 184). The main objective was simply to select a significant majority of Selected Topics trial cases, since these constitute the majority of cases on which SIROCCO was "trained" and developed. On the other hand, selecting at least some cases outside of SIROCCO's primary area of knowledge allows an analysis of whether the program can handle, in at least reasonable fashion, an even wider range of cases.

To test the primary thesis, i.e., that SIROCCO's operationalization techniques make a difference in the accuracy of its predictions, SIROCCO was pitted against five competitor methods, shown in Figure 1-4. A worst-case baseline was established by the use of a random selection method, RANDOM, and a second, more-sophisticated random method, INFORMED-RANDOM, which selects codes and cases randomly but in accordance with historical citation frequency. Because engineering ethics is a novel domain with no comparable benchmark systems, it was necessary to find a method or methods that could be expected to perform well at SIROCCO's retrieval task, even though not specifically designed for such a task. *Full-text retrieval* was a natural place to look for such methods, since engineering ethics is a heavily linguistic domain and the full-text retrieval systems require no case representation. Also, *textual case-based reasoning*, a subarea of CBR in which full-text retrieval techniques are used, has recently attracted considerable attention in the literature as a viable alternative to traditional representation-based CBR, especially in domains in which the cases are available in text and are linguistically complex [Burke et al., 1997; Lenz, 1998; Brüninghaus and Ashley, 1998; Brüninghaus and Ashley, 1999]. MG [Witten *et al.*, 1999], a full-text retrieval system based on the relatively simple, yet powerful, vector space model, was selected as one competitor and a variant of that method, EXTENDED-MG, believed to be potentially more accurate than the base method, was also included. The last competitor method was an ablated version of SIROCCO, called NON-OP SIROCCO, which only very weakly employs operationalization techniques.

RANDOM:	In this method, codes and cases are randomly selected for each case.
INFORMED-RANDOM:	Here, codes and cases are selected randomly for each case, but the overall selection is in accordance with the distribution of code and case citations found within the corpus of NSPE BER cases decided between 1958 and 1992. In addition, only the most frequently cited 20 codes and 40 cases are considered for selection.
MG (Managing Gigabytes):	In this method, codes and cases are selected using a full-text retrieval method, specifically, the vector space model. Given the text of a new ethics case, MG converts the case into a vector in n-dimensional space – a representation that accounts for term frequency, term weights, and length of text – and retrieves codes by comparing the input case vector to the vectors representing all of the NSPE codes. The code vectors with minimal angles to the input vector are chosen. In like fashion, MG retrieves cases by comparing the input vector to vectors representing all of the NSPE cases.
EXTENDED-MG:	In this method, codes and cases are selected using a variant of the MG method. In particular, cases are retrieved as in MG, i.e., by comparing the input vector to vectors representing all of the NSPE cases, but codes are selected according to their frequency of citation in the top X selected cases.
NON-OP SIROCCO:	This is an ablated version of SIROCCO in which virtually all functionality related to operationalization is severed from retrieval. Specifically, this method uses Stage 1 retrieval and the basic dot product calculation, as described in Section 1.3.3, but it does not employ Stage 2 structural mapping or the analysis phase of SIROCCO. Thus, the method does not use most of SIROCCO's important operationalization knowledge, such as focusing on the Questioned and Critical Facts in Stage 1, using *Code* and *Case Instantiations* to focus structural mapping in Stage 2, reusing past operationalizations, and applying selection heuristics based on operationalizations in the Analysis phase.

Figure 1-4: The Five Methods Compared to SIROCCO to Test the Primary Thesis

Each method, including SIROCCO, was given the entire set of trial cases to process one-by-one, and the retrieval results of each method for each case were then compared to the code and case citations of the ethical review board for the same cases. To calculate overlap between the method's solution and the board's solution, a metric known as the *F-Measure* [van Rijsbergen, 1979, p. 173-176; Lewis *et al.*, 1996], an information retrieval metric that combines precision and recall, was employed. Two F-Measure values were computed for each method for each case, one representing exact matches of codes and cases between the method's solution and the board's and one representing inexact matches of codes and cases. Inexact matches of codes were determined based on a Code Hierarchy that clusters similar codes together and inexact matches of cases were determined using a citation overlap metric.

Not surprisingly, SIROCCO significantly outperformed both the RANDOM and INFORMED-RANDOM methods. These methods were included only to provide a low-level baseline. More interestingly, the results showed SIROCCO to be clearly more accurate in retrieving codes and cases than the MG and NON-OP SIROCCO methods. Although the evidence was somewhat weaker that SIROCCO is more accurate than EXTENDED-MG – in particular, SIROCCO was significantly more accurate in retrieving *exact* codes and cases, but it was inconclusive whether it was more accurate in retrieving *inexact* codes and cases – there was still enough supporting statistical data to conclude that SIROCCO is superior to EXTENDED-MG. Outperforming MG and EXTENDED-MG shows that SIROCCO is indeed a powerful retrieval method, because the full-text retrieval methods are strongly competitive alternatives for performing SIROCCO's task. The fact that SIROCCO outperformed EXTENDED-MG which, in turn, outperformed MG is also significant with regard to the primary thesis. I will argue that EXTENDED-MG actually makes weak use of one of the case operationalization techniques, *Reuse an Operationalization*, by selecting those codes that appear most frequently in the top-rated cases. Thus, by augmenting a full-text retrieval method with an operationalization technique, I was able to demonstrate superiority over pure full-text retrieval. In turn, SIROCCO, with its wide range of operationalization techniques, outperforms both the full-text retrieval system and the version augmented with an operationalization technique.

The fact that SIROCCO significantly outperformed NON-OP SIROCCO provides perhaps the strongest evidence that SIROCCO's operationalization techniques do, in fact, make a difference. SIROCCO and NON-OP SIROCCO share the same case representation and Stage 1 retrieval method. Where they differ, and critically so, is in their use of the operationalization techniques. NON-OP SIROCCO makes only weak use of the techniques – as with EXTENDED-MG, it also uses the *Reuse an Operationalization* technique by selecting codes that appear most frequently in the list of top-rated *N* cases – whereas SIROCCO makes extensive use of the operationalization techniques in performing its retrieval task.

Because I believed that some of the code and case citations made by SIROCCO could be considered reasonable, even though they had not been made by the board, I executed a second experiment in which I asked two evaluators, graduate students with a background in ethics, to evaluate the extra code and case citation made by SIROCCO and SIROCCO's closest competitor, EXTENDED-MG, for the trial cases. While the board provides the best available gold standard with which to benchmark the methods, the board may have missed some relevant cases or code citations. The board's composition changes over time. Not all of the board members are certain to be familiar with all of the past decisions (something a tool like SIROCCO could remedy.) In short, I believed that SIROCCO might have been able to identify relevant codes and cases that the board simply overlooked or did not consider. For this experiment, the evaluators were asked to indicate, for each additional code and case suggested by the two methods, whether the extra suggestion was reasonable or not.

After receiving the evaluator's responses, and accounting for inter-rater reliability, I recalculated SIROCCO's and EXTENDED-MG's F-Measures for the 58 trial cases, counting the extra citations rated as "reasonable" by the evaluators as board citations. This resulted in a 35% increase in SIROCCO's mean F-Measure of inexact matches, compared to only a 23% increase in EXTENDED-MG's mean F-Measure of inexact matches. This proved to be crucial, as SIROCCO's statistical performance on *both* exact and inexact matches for this experiment was now shown to be significantly better than EXTENDED-MG's. Thus, this experiment provides even more compelling evidence that SIROCCO's retrieval method is indeed superior to EXTENDED-MG's. Finally, I uncovered some specific situations in which SIROCCO, due to its unique retrieval approach and case base, suggested codes or cases that were not cited by either the board or EXTENDED-MG. Although anecdotal, these results are quite interesting, and I present and discuss an example of this phenomenon in Section 4.2.3.

To test the secondary thesis, i.e., that SIROCCO's temporal knowledge makes a difference in the accuracy of its predictions, the trial cases were processed by an ablated version of the program, NON-TEMP SIROCCO, that did not employ temporal knowledge. As with the initial experiments, the results of NON-TEMP SIROCCO were compared against the suggestions made by the ethical review board and the F-Measure calculated for each individual sample and as a mean value over all samples. These results were then compared to the output of the standard version of SIROCCO, which did apply temporal knowledge. The results were somewhat surprising, and with respect to my secondary thesis, disappointing. The differences between SIROCCO with and without its temporal knowledge were essentially negligible.

In the dissertation I discuss why I believed that temporal knowledge could make a difference in case retrieval, and I show an example to illustrate this point. I also provide some analysis and suggestions as to why SIROCCO's temporal knowledge did not make a difference in these experiments. For instance, it is possible that a majority of the trial cases simply did not require the use of temporal reasoning. Other suggestions that are discussed include possible inaccuracies in the case representations and idiosyncrasies of the SIROCCO implementation that may have failed to adequately capture the subtleties of the temporal relations. In the end, I am still convinced that employing temporal knowledge as a component in similarity assessment and retrieval is important in some domains and should be pursued by future interpretive CBR and CBR researchers.

Finally, experiments were run to test the trade-off between accuracy and efficiency produced by SIROCCO and to gauge how well SIROCCO scales up. To test accuracy versus efficiency, a test was executed in which SIROCCO first analyzed each trial case using only the other trial cases for its case base. As before, the F-Measure was calculated for each sample and as a mean but, in addition, the run time of all 58 trial cases was captured. Subsequently, a series of 19 additional tests were run. For each test, a random set of 10 cases from the foundational set were added to the case base (except for the final test, in which only 4 cases remained to be added), the entire trial set was again run through SIROCCO, and the F-Measure and timing results were taken. The results showed that, as SIROCCO's case base grew in size, accuracy generally increased while run time also increased, but in no worse than linear time.

A second accuracy-versus-efficiency experiment was run in which the number of cases passed from Stage 1 to Stage 2 (N, a user-specified parameter) was varied from 1 to 15. All of the trial cases were run at each value of N and, again, F-Measure values were calculated and timing results measured for each test. As noted in Section 1.3.3, Stage 2's computation naturally increases as more cases are passed to it from Stage 1, because it applies structural mapping to all of the candidate cases supplied to it. The results revealed that after $N = 6$ SIROCCO's accuracy did not improve; in fact, its accuracy decreased for all $N > 6$, while its run time increased considerably. Thus, it appears there is no benefit to running SIROCCO with an N value greater than 6.

The final experiment was intended to evaluate SIROCCO's capability to scale up. Because the program's case base was limited by the number of cases that had been transcribed into EETL (246: 184 foundational cases plus 58 trial cases plus the 4 unused trial cases), it was necessary to devise a strategy to artificially scale up the number of cases. This was achieved by running a PERL script over the Common Lisp foundational case file and generating new files in which case numbers and critical item names were slightly altered, so that SIROCCO would consider each new file to be a set of 184 new cases. Ten new foundational case files were generated, resulting in a total of 2,086 cases, including the original foundational cases and the trial cases. Ten tests were then run in which, for each test, the case base was increased by 184 cases. For each test all of the trial cases were provided as input and the total run time was captured. For this test, the run time increased linearly with the increased case base size, from 575 seconds for the original 242 cases to 1705 seconds for 2,086 cases, certainly a reasonable increase in time.

I argue in the dissertation that it is unlikely that a case base in engineering ethics, or perhaps in any weak analytic domain, would ever need to grow as large as 2,000 cases, so this is clearly a worst-case result. First, a smaller case base of carefully represented cases is likely to provide appropriate coverage for a wide range of scenarios, in the sense described by Smyth and McKenna [1999]. For instance, even conservatively assuming the need for having cases that cover 200 separate issues (many more than listed in, for instance, the NSPE Ethics Reference Guide [1996, p. 8]), a case base of 2,000 cases would provide approximately 10 exemplars of each issue, certainly more than SIROCCO requires to make good suggestions. Second, because interpretive CBR systems are typically expected to retrieve close, but imprecise, cases that humans then interpret, analyze, and adapt, there is less need for full, detailed coverage of possible scenarios. In other words, the competence of a single case is relatively wide in weak analytic domains because of the purpose they are put to – providing a general guidepost for a human analyst.

1.4. A Guide to the Remainder of the Dissertation

This section provides a brief guide to the remainder of the dissertation.

Chapter 2 presents additional background on the domain of engineering ethics, including how reasoning with principles appears to work, and shows specific examples of how the operationalization techniques are applied by the ethical review board, so that the reader can appreciate the nuances of the problem domain and SIROCCO's task. In addition, there is a discussion of how temporal reasoning appeared to be applied by the board.

Chapter 3 builds upon the introduction to SIROCCO provided in this chapter, as well as the discussion of the engineering ethics domain found in Chapter 2, by presenting a detailed view of the program and its architecture and how SIROCCO's approach implements the observed reasoning of the board of ethical review. An example case from the NSPE BER corpus is furnished to focus the discussion, and that case is used throughout the chapter to illustrate the functionality and features of the program. The chapter concludes by summarizing how SIROCCO specifically uses operationalization techniques, temporal reasoning, and explanation in the service of its goals.

Chapter 4 presents, in detail, the SIROCCO experiments and the results discussed briefly in the current chapter. Both quantitative and qualitative results and analysis are provided. The chapter also draws some important conclusions regarding SIROCCO which are expounded upon in Chapter 6, the final chapter of the dissertation.

Chapter 5 relates SIROCCO and my research to other computational work in the areas of case-based reasoning, analogy, and practical ethics. As previously mentioned, SIROCCO is most closely aligned with work in the area of interpretive CBR and the emphasis in this chapter is on how SIROCCO extends and improves upon that research.

Finally, Chapter 6 concludes the dissertation by discussing the relative success of this research vis-à-vis the primary and secondary theses. Also discussed are the contributions of the research and lessons learned. This chapter concludes with a discussion of possible future work.

2. The Example Domain: Engineering Ethics

The case of Roger Boisjoly and the Space Shuttle Challenger, discussed in the previous chapter, illustrates both the importance of decision making in the engineering field and the difficulty of applying ethical principles to that decision making. This chapter briefly introduces the domain of engineering ethics and presents an overview of the empirical study and analysis of a particular engineering ethics subdomain, the NSPE BER case opinions.

2.1. What is Engineering Ethics?

The professional responsibilities of an engineer extend beyond the myriad technical decisions that he or she must make. Engineers play an important role in protecting and helping the public, and that role requires moral responsibility, imagination, and courage, in addition to technical skills [Harris *et al.*, 1999, p. 7]. Although most engineering scenarios are not as high-profile or dramatic as the Challenger case [Pinkus *et al.*, 1997] – indeed, none of the remaining cases discussed in this dissertation are quite so gripping – ethical decision making is nonetheless integral to the professional life of an engineer.

Engineering ethics is a domain that relies heavily on the concept of a *principle*. A moral principle is a general normative rule of behavior. According to philosophical theories (e.g., [Kant, edited and published in 1969]) and psychological theories (e.g., [Colby and Kohlberg, 1987a; Colby and Kohlberg, 1987b]), moral principles play important roles in the ethical life of humans. First, they tell us whether a particular situation raises ethical issues. Having knowledge of and sensitivity to moral principles helps us to recognize situations in which our actions may be ethically questionable. Second, moral principles provide us with some guidance on which facts in an ethical dilemma[7] are relevant to its resolution. Typically, not all of the facts in a given situation are ethically relevant; moral principles help to differentiate the relevant from the irrelevant facts. Finally, moral principles provide us with guidance regarding appropriate actions to take in resolving ethical problems. Principles typically prescribe a general course of action, such as “tell the truth” or “be fair.” (Note that Carol Gilligan [1982] challenges the central role of principles, endorsed by Kohlberg, in moral problem solving. Her research indicates that a woman’s approach to moral reasoning is oriented more toward relationships and caring than toward principles.)

[7] I will use the words “dilemma” and “problem” interchangeably in discussing ethical problems. By both “ethical dilemma” and “ethical problem” I mean a situation in which more than one action may be taken by a protagonist where the choice of action has ethical ramifications.

It is important to separate the professional ethics required of an engineer from personal ethics. It is certainly true that engineering ethics has a foundation in personal ethics. For instance, the public safety codes discussed in connection with the Challenger case (i.e., I.1. and II.1.a.) are clearly rooted in the general principle “avoiding harm,” one of four principles, along with fairness, veracity, and producing benefit, stressed as “good dispositions whose tendency is directly to countervail the limitation of human sympathies” [Warnock, 1971, p. 79][8]. Engineering ethics is inextricably tied to personal ethics: the urge to make ethical decisions in one’s profession is tied to one’s desire to be an ethical human being [Harris *et al.*, 1999, p. 8].

On the other hand, there are significant differences between professional ethics, in particular engineering ethics, and personal ethics [Harris *et al.*, 1999, p. 8-11]. The standards and rules of engineering ethics, and the ethics of any professional domain, constitute a specialized type of morality known as *role morality.* Such a system of morality is framed and dictated by the particular obligations of a role one assumes. For instance, the engineer has ethical obligations to the public, to clients, and to employers. In addition, the rules that guide and judge professional behavior are typically defined by professional communities or societies. For instance, in engineering the National Council of Examiners for Engineering and Surveying (NCEES) has recommended “Model Rules of Conduct” as ethical guidance for state boards of registration. As another example, the NSPE, introduced in Chapter 1 and the domain focus of this dissertation, is an engineering society, open to engineers in any discipline, that defines its own set of principles. No such organized communities or societies define the standards and principles of personal ethics.

2.1.1. Ethical Problem Solving and Decision Making in Engineering

Instead of measuring a principle’s "weight" quantitatively or applying principles deductively, decision makers in engineering ethics appear to reason symbolically and qualitatively with principles. They identify the facts that are relevant in light of applicable principles, resolve conceptual issues by defining terms of the principles and their application to case facts, and engage in moral reasoning (e.g., use past cases for "line-drawing" comparisons, employ creative middle-way solutions) [Harris *et al.*, 1999]. In other words, engineers, like practitioners in medical ethics and other practical fields of endeavor, appear to rely heavily on specific circumstances – i.e., cases – to resolve, interpret, and apply ethical principles [Jonsen and Toulmin, 1988; Strong, 1988; Arras, 1991].

---

[8] In an important and influential book in bioethics, Beauchamp and Childress suggested a similar set of four general principles: autonomy, beneficence, nonmaleficence, and justice [1978].

A system of explicit, middle-level ethical principles tailored to the role morality of engineering ethics is known as a *code of ethics.* Although an engineering code of ethics provides "rules" of ethical behavior for practicing professional engineers, typically the principles are stated in an abstract and sometimes quite complicated fashion. As a result, applying the codes to particular fact situations is not straightforward. Engineering codes exemplify all of the characteristics of the weak analytic domains, discussed in Chapter 1. In particular, the engineering codes and conclusions to the codes are available almost exclusively at a highly abstract level, and the codes can conflict with one another with no formal means of arbitrating between them. Consider, for example, the following code from the National Society of Professional Engineers [1996]:

> "*Code II.5.a.* Engineers shall not falsify or permit misrepresentation of their ... academic or professional qualifications. They shall not misrepresent or exaggerate their degree of responsibility in or for the subject matter of prior assignments. Brochures or other presentations incident to the solicitation of employment shall not misrepresent pertinent facts concerning employers, employees, associates, joint ventures or past accomplishments with the intent and purpose of enhancing their qualifications and their work."

This middle-level ethical principle specializes the more general principle of “honesty” in an engineering context. Each of the three sentences in the code deals with a different aspect of "misrepresentation of an engineer," and each sentence covers a wide range of possible circumstances. The precise circumstances that support application, however, are not specifically stated. Knowing whether this code applies to a particular fact situation requires that one must recognize the applicability of and interpret open-textured terms and phrases in the code, such as “misrepresentation” and “intent and purpose of enhancing their qualifications.”

Even if one is able to determine that a code applies to a specific situation, another problem is that the engineering ethics codes sometimes conflict with one another in specific circumstances. For instance, suppose that an engineer determines that her employer’s company brochure contains a “misrepresentation” of the company’s capabilities, as covered by NSPE Code II.5.a, above. However, suppose also that the “misrepresentation” is relatively minor, i.e., it is not likely to have much, if any, impact on a potential client’s decision making, and it would cost the company substantial money to change and reprint the brochure. What should our engineer do? Push for a change to the brochure? One of the codes discussed earlier, in connection with the Challenger case, might also be said to apply here, i.e., “Engineers shall act in professional matters for each employer or client as faithful agents or trustees” (NSPE Code II.4.). This code places an obligation on the engineer to do what is best for his or her employer and, in this example, this could be interpreted to mean not pushing for a change to the brochure. When engineering codes conflict – as they do in the particular circumstances of this hypothetical situation – which of the conflicting codes should one follow?

Engineering codes, while providing important guidance for the engineering profession, are highly abstract, open-textured, and subject to interpretation. The abstract specification of the codes makes it hard to determine the particular circumstances in which they apply. In addition, it is difficult to know the precise action that a code advocates. In engineering ethics, the specific context of a case helps to interpret a code. Thus, answering the questions posed in the previous paragraph entails a careful examination of all of the specific facts of a case. In other words, specific circumstances are essential for understanding, interpreting and applying principles, as well as arbitrating between principles that conflict with one another. This reliance on in-context interpretation and application is the key to the operationalization techniques that are the central theme of this dissertation.

Harris and his colleagues emphasize the use of cases in studying and understanding engineering ethics. In fact, their textbook begins with the presentation and discussion of several famous engineering ethics dilemmas, including the Challenger case discussed above. Throughout the book the authors use cases to illustrate the application of principles, the use of various ethical problem solving techniques, and the important issues a professional engineer may encounter. In effect, Harris et al advocate a *casuistic* approach to resolving engineering ethics problems. Casuistry is a form of ethical reasoning in which decisions are made by comparing a problem to paradigmatic, real or hypothetical cases [Jonsen and Toulmin, 1988; Strong, 1988; Arras, 1991]. Harris and his colleagues describe the significance of cases as follows:

> "[Cases'] importance cannot be overemphasized. It is by studying cases that we can most easily develop the abilities necessary to engage in constructive ethical analysis. Cases stimulate the moral imagination by challenging us to anticipate the possible alternatives in resolving them and the consequences of those alternatives. Through cases we learn to recognize the presence of ethical problems and to develop the analytic skills necessary to resolving them. A study of cases is the most effective way to see that the codes cannot provide ready-made answers to all moral questions that professional engineering practice generates and that the individual must become a responsible moral agent. Finally, the study of cases convinces us that there may be some unresolvable uncertainties in ethical analysis and that in some situations rational and responsible professionals may disagree about what is right." [1999, p. 22].

The importance of cases is also apparent in the database and documentation of NSPE BER cases, the domain focus of this dissertation. Cases are used to "supplement and amplify" the codes so that engineers may more fully understand their application [NSPE, 1958-1998, Volume II, p. iii]. A high percentage of the decided opinions of the NSPE BER reference past cases. Finally, the NSPE BER has provided a "Case Index" in which all of the codes in the NSPE code of ethics are cross-referenced with the cases that cite these codes [NSPE, 1996, p. 18-20]. In short, it is clear that the NSPE BER takes a stance similar to that of Harris *et al.*, with respect to cases.

The following section discusses the empirical analysis of the NSPE BER cases that led to the identification of the operationalization techniques, the recognition of the importance of temporal constraints, and the development of SIROCCO.

2.2. The Empirical Analysis of the NSPE BER Cases

Although both codes and cases are clearly integral to the decision making and justifications of the NSPE board of ethical review, it is an empirical question *specifically* how the codes and cases are used to solve and justify cases. Thus, I undertook a systematic analysis of the NSPE BER database of cases and documentation in order to ascertain the approach used by the review board and to develop a computational model that can leverage elements of its reasoning.

As discussed briefly in Section 1.3, the NSPE Board of Ethical Review is a committee of five to seven professional engineers that has met yearly since 1958 to discuss and decide engineering ethics dilemmas[9]. The individual members of the review board are appointed by the NSPE president to serve a 3-year term. BER members can be reappointed, but they cannot succeed themselves for more than one term. The dilemmas addressed by the NSPE BER are submitted by members of the professional engineering community. The purpose of the NSPE BER is not to mete out punishment to offending engineers and organizations, but rather to provide an educational resource to the engineering community. Thus, the actual names of the individuals and organizations involved in these dilemmas are not provided in the cases published in hardcopy or on the Internet. The NSPE Board of Directors approves the opinions "with the hope that they will serve to make the profession's ethical principles a living and dynamic force." [NSPE, 1958-1998, Volume II, p. iii]

Over the course of more than 40 years, over 400 particular fact situations and have been addressed, decided, and published by the NSPE BER. Each published fact situation is presented and structured by the following subsections: (1) A *title*, that provides both a year-sequence label

[9] The NSPE BER officially began in 1954, but it did not start publishing cases until 1958. It is unknown whether the board actually convened and decided cases between 1954 and 1957 [Personal Communication with Arthur Schwartz, the NSPE's legal counsel, and Mary Ann Cannon, the NSPE's Legal Administrative Assistant.]

(e.g., 90-5 is the fifth fact situation of 1990) and a mnemonic description of the primary issue of the fact situation (e.g., "Failure to Report Information Affecting Public Safety"); (2) the *facts*, a brief description of the events of the scenario, usually between one and three paragraphs in length; (3) an *ethical question* (or questions) raised by the facts, each question corresponding to a distinct case[10]; (4) a list of *code references* that are (or may be) relevant to the fact situation; (5) a *discussion* of the fact situation by the review board that represents a consensus analysis and includes citations to codes and past cases; (6) the *conclusion* reached by the board for each of the ethical questions; and, optionally, (7) a *dissenting opinion* by one or more board members.

2.2.1. The Analysis Process and General Results

A total of 371 NSPE BER fact situations (comprising 475 cases) decided between 1958 and 1992 were chosen as the basis for the empirical analysis. The intent was to use these cases as the case base for the computational model to be developed subsequently. Cases decided between 1993 and 1998 were excluded from the analysis so that they could be used as the trial set for the experiments. The reason for choosing a larger set of earlier cases for the computational model's case base is discussed in detail in Section 4.1.4. Briefly, this division of foundational and trial cases was chosen to set up a more realistic evaluation (i.e., increasing the chances that the trial set of later cases would have the actual earlier cases available for citation) and to assure that a reasonably large pool of cases could be cited by all of the trial cases.

All of the 371 fact situations were carefully read. For approximately 50 of the fact situations the following steps were taken: (1) Text in the discussion section that either explicitly or implicitly cited or discussed codes, past cases, or important issues was underlined or highlighted. (2) An informal indication of the purpose of each citation or discussion was written next to the underlined or highlighted text, e.g., "making an analogy to a past case," "interpreting some terms of the code," "deciding which code takes precedence in the context of this case," etc. After the initial mark-up of the texts, a second pass was used to correlate the citations of the different fact situations to ascertain whether a canonical set of techniques were applied. At this stage the operationalization techniques discussed in Section 1.2 were identified and cataloged. Next, to verify the general applicability of the operationalization techniques, a substantial number of the 371 NSPE BER cases that were not included in the original set of approximately 50 cases were reread. Rereading these cases confirmed that the operationalization techniques were, in fact, identifiable and repeated across a large segment of the foundational fact situations. Thus, it was

[10] As discussed in Section 1.3.3, a fact situation that raises multiple ethical questions comprises multiple cases, one for each question.

concluded that the NSPE BER implicitly applies the operationalization techniques in resolving dilemmas and/or justifying its conclusions.

In addition to identifying the operationalization techniques, the empirical analysis also indicated that the board employs a structured, yet evolving and dynamic, resolution process in analyzing cases. In particular, the review board appears to resolve cases by:

1. Selecting potentially relevant codes and past cases.
2. Interpreting the potentially relevant codes and past cases in the context of the new case.
   - For a code or case that is clearly applicable or analogous to the new case, establishing how the code or case impacts the decision in the new case (e.g., an unviolated code supports a conclusion that the questioned action is ethical; a similar past case with a few distinguishing facts may support a conclusion opposite from that of the past case, etc.).
   - For a code or case that is marginally applicable or analogous to the new case, determining in what way the code is useful or relates to the discussion of the case (e.g., the code relates to the actions of someone other than the main protagonist, the past case raises an important ethical issue that merits discussion, etc.)
   - For a relevant past case, reapplying (and, perhaps, reinterpreting) the code and case citations of the past case to the new case.
3. Given the interpretations of Step 2, deciding whether the protagonist's action(s) were ethical, unethical, or ethical in a qualified way (e.g., "Engineer A was not unethical, provided he did X (where X is an action not stated in the facts).").
4. Recording the interpretations of Step 2 and the decision of Step 3 for use in future case analyses.

In this dialectical process, codes and cases interact in an essential way; neither satisfactorily supports decision making and justification without the other. New codes are sometimes introduced, old codes are reworded, and new interpretations of codes (and cases) are provided in the context of new cases. The changed and reinterpreted codes sometimes lead to case decisions that differ from very similar or virtually identical circumstances in earlier cases – a nonmonotonic reasoning process. In essence, the codes – and principles from which the codes are defined – act like "guideposts," focusing engineers on important issues and dimensions of fact situations, but not providing definitive resolutions without interpretation. The codes' evolving abstract nature – requiring grounding in the specific circumstances of cases – defies definition in terms of deductive rules.

Finally, the empirical analysis indicated that temporal knowledge factors into the decision making and justifications of the review board. Knowing the sequence of actions and events in an engineering ethics scenario is important in deciding whether codes are relevant, violated, or not violated. Likewise, temporal sequence can be important in drawing an analogy from one case to another. Similar patterns of facts, including matching temporal relations, appear to help in drawing analogies between cases. Small differences in the chronological sequence of actions and events across cases can either invalidate an analogy altogether or turn a relevantly similar analog into a case with distinguishing characteristics.

To illustrate these findings, the following sections present examples of the mark-up and analysis of actual NSPE BER cases from the empirical study. In particular, these sections show specific examples of (a) how the operationalization techniques are applied by the review board and (b) how temporal knowledge affects the decision making of the board.

2.2.2. How Does the Review Board Use the Operationalization Techniques?

This section shows the way in which the review board employs operationalization techniques. Several NSPE BER cases that were analyzed during the empirical analysis are used as examples.

Before discussing the examples, it is worth briefly mentioning that the analysis of the NSPE BER discussions was necessarily somewhat subjective. Many conclusions rest upon implicit assumptions of what the board meant and how it ties stated facts to its analysis. However, in any analysis of free-form natural language text, such assumptions are unavoidable.

The first example is Case 90-5-1, the public safety case discussed in the previous chapter (i.e., in Section 1.3.3 and shown in Figure 1-2). The facts of the case, the ethical question it raises, and the relevant code provisions are shown in Figure 2-1. Recall that the case involves an engineer who discovered serious structural defects during a building inspection. The engineer reported the defects to his client but did not report the problems to an outside authority, because his client requested confidentiality. The question is whether it was ethical for the engineer to conceal his knowledge of the safety-related defects in deference to his client's confidentiality.

Facts of Case 90-5-1: Failure to Report Information Affecting Public Safety

Tenants of an apartment building sue the owner to force him to repair many defects in the building that affect the quality of use. The owner's attorney hires Engineer A to inspect the building and give expert testimony in support of the owner. Engineer A discovers serious structural defects in the building, which he believes constitute an immediate threat to the safety of the tenants. The tenants' suit has not mentioned these safety-related defects. Upon reporting the findings to the attorney, Engineer A is told he must maintain this information as confidential as it is part of a lawsuit. Engineer A complies with the request of the attorney.

Question:

Was it ethical for Engineer A to conceal his knowledge of the safety-related defects in view of the fact that it was an attorney who told him he was legally bound to maintain confidentiality?

References:

Code II.1.a. – "Engineers shall at all times recognize that their primary obligation is to protect the safety, health, property and welfare of the public. If their professional judgment is overruled under circumstances where the safety, health, property or welfare of the public are endangered, they shall notify their employer or client and such other authority as may be appropriate."

Code II.1.c. – "Engineers shall not reveal facts, data or information obtained in a professional capacity without the prior consent of the client or employer except as authorized or required by law or this Code."

Figure 2-1: The Fact Situation, Question Raised, and Relevant Codes in Case 90-5-1

The most-relevant portion of the review board's analysis of Case 90-5-1 is shown in Figure 2-2. Less-relevant portions are not shown, and the absence of this text is indicated by ellipses. Text that appears to indicate the use of an operationalization technique by the board is underlined, and the particular technique that is applied is shown on the right side of the figure with an arrow (or arrows) pointing to the relevant underlined text. In the following paragraphs, each of the applied operationalization techniques is discussed.

The board appears to apply the operationalization technique *Cite an Analogous Precedent* (a type of *Case Instantiation*) in the first two paragraphs of Figure 2-2. Such a technique is applied when the board wishes to draw attention to the relevant similarities between the current case and a past case in order to argue that the conclusion reached in the past case should also be reached in the present case. In the first sentence of their discussion, the board stresses the importance of an engineer's obligation to protect public safety. Next, the board explicitly cites Case 84-5-1[11] as an example of a scenario in which this obligation was the overriding consideration in deciding the case. The facts of the past case are then summarized. In the next paragraph, the underlined text beginning "the Board noted that Section II.1.a. admonishes engineers to recognize that their primary obligation is to protect the public" is critical in summing up the decision in Case 84-5-1

[11] This case citation (as well as all other case citations in the NSPE BER texts discussed in this section) has been modified to conform to the adopted case nomenclature used in the dissertation. In particular, a "1" was appended to "Case 84-5," signifying a citation to the first question of fact situation 84-5. Note that when a case enterer actually transcribes a case, it is his or her responsibility to determine which specific case (i.e., question) of a fact situation is referenced in the citing text.

and in drawing an analogy to the current case. Although the specific facts of Case 84-5-1 are quite different from Case 90-5-1 – and 84-5-1 does not involve the issue of confidentiality, as does 90-5-1 – the overriding issue of public safety is shared by both cases. In 84-5-1, the board ruled that the engineer's actions were unethical, and that decision appears to have supported, at least in part, a similar conclusion in 90-5-1 (the conclusion is shown at the bottom of Figure 2-2).

The same text that summarizes the decision in Case 84-5-1 could also be said to invoke the operationalization technique *Reuse an Operationalization.* This technique is used when the board cites a past case in order to reuse an operationalization that was applied in that case. The basic idea is that the operationalization from the board's analysis of the past case is likely to also be relevant in the context of the new case. The passage beginning "the Board noted (in 84-5-1) that Code II.1.a. …," indicates that II.1.a. was interpreted and linked to the facts of Case 84-5-1 in the analysis of that case (i.e., a *Code Instantiation* was applied). By discussing that *Code Instantiation* in the context of Case 90-5-1, the board indicates that the Instantiation is also relevant to – in other words, it may be "reused" in – the current case.

The next operationalization technique that the board appears to apply in Figure 2-2 is *Define or Elaborate a General Issue or Principle.* This technique is used to define or raise a relevant issue by citing a past case in which the issue was involved. The cited past case is typically not similar enough to the present case to consider it analogous or distinguishing, but the general issue raised in that case is relevant in the context of the present case. The citation and discussion of Case 82-2-1 in Figure 2-2 appears to fulfill this purpose. Case 82-2-1 involves the "obligation of (an engineer) not to reveal information of the client without prior consent of the client," and it is cited by 90-5-1 for this reason. However, not only are the facts of 82-2-1 (see case summary in Figure 2-2) significantly different from those in 90-5-1, but 82-2-1 also does not involve the central issue of 90-5-1: public safety. Thus, Case 82-2-1 appears to have been cited simply to make the reader aware of the issue of confidentiality, and not to draw attention to the similarities between 82-2-1 and 90-5-1.

The central theme of Case 90-5-1 – the conflict between the public safety code II.1.a. and the confidentiality code II.1.c. – is addressed by the next operationalization technique: *Define the Superior Code*. This technique is used to arbitrate between two or more conflicting codes. The board acknowledges the conflict between Codes II.1.a. and II.1.c. in the following passage: "It is clear that there may be facts and circumstances in which the ethical obligation of engineers in protecting the public health and safety conflict with the ethical obligation of engineers to maintain the right of confidentiality…" In the final paragraph of Figure 2-2, the board makes it clear that Code II.1.a. takes precedence over Code II.1.c. in this case: "… in cases where the public health and safety is endangered, engineers not only have the right but also the ethical responsibility to reveal such (confidential) facts …"

The obligation of the engineer to protect the public health and safety has long been acknowledged by the Code of Ethics and by the Board of Ethical Review … A good example is BER Case 84-5-1. There, a client planned a project and hired Engineer A to furnish complete engineering services for a project. Because of the potentially dangerous nature of implementing the design … Engineer A recommended to the client that a full-time, on-site project representative should be hired for the project. After reviewing the completed project plans and costs, the client indicated to Engineer A that the project would be too costly if such a representative were hired.

Case Instantiation (Cite an Analogous Precedent)

Engineer A proceeded with the work on the project even though he had recommended that a full-time, on-site project representative should be hired. In discussing the issue … the Board noted (in 84-5-1) that Code II.1.a. admonishes engineers to recognize that their primary obligation is to protect the public safety, health, property and welfare. Under the facts, Engineer A did not recognize this primary obligation. … For that reason, Engineer A was in violation of Code II.I.a.

Reuse an Op (Code Instantiation)

Although the public health and safety clearly is the most basic and fundamental ethical obligation … other important ethical obligations exist ... One important ethical consideration is the obligation of engineers not to reveal information of the client without the prior consent of the client.

The Board has had reason to consider this ethical issue on occasion. In BER Case 82-2-1, Engineer A offered home inspection services ... Engineer A performed this service for a client for a fee and prepared a one-page written report, concluding that the residence was in generally good condition requiring no major repairs, but noting several minor items needing attention. Engineer A submitted his report to the client showing that a carbon copy was sent to the real estate firm handling the sale of the residence. The client objected ... In concluding that Engineer A acted unethically … the Board concluded that although it did not appear from the facts that Engineer A had acted with some ulterior motive … the principle of the right of confidentiality on behalf of the client predominated.

Define or Elaborate a General Principle or Issue

Given these two cases, it is clear that there may be facts and circumstances in which the ethical obligation of engineers in protecting the public health and safety conflict with the ethical obligation of engineers to maintain the right of confidentiality in data and other information obtained on behalf of a client…

Define the Superior Code

It appears that Engineer A, having become aware of the imminent danger to the structure, had an obligation to make absolutely certain that the tenants and public authorities were made immediately aware of the dangers that existed… Engineer A had an obligation not to reveal facts, data or other information in a professional capacity without the prior consent of the attorney. However, there were valid reasons why Engineer A should have revealed the information ...

Unlike the facts presented in BER Case 82-2-1, there is not any conflict or potential conflict of interest that exists between owner and attorney with regard to the information. ...

Code II.1.c. makes a clear exception concerning the obligation of engineers not to reveal facts obtained in a professional capacity without the client's consent. That exception allows the disclosure of such information in cases authorized by the Code or required by law. We believe that in cases where the public health and safety is endangered, engineers not only have the right but also the ethical responsibility to reveal such facts to the proper persons…

Code Instantiation

Conclusion:

It was unethical for Engineer A to not report the information directly to the tenants and public authorities.

Figure 2-2: The NSPE BER Analysis of Case 90-5-1 and the Operationalization Techniques Applied

The final operationalization technique indicated in the figure is a *Code Instantiation*. This technique is used to interpret a code's conditions and to discuss how the facts of the case satisfy those conditions. In the analysis of Case 90-5-1, the board makes it clear that the engineer in this case was compelled by the conditions of Code II.1.c. not to reveal his client's confidential information. However, in the final paragraph of the analysis, the board also cites two exceptions to this application of the code: "(II.1.c.) allows the disclosure of such information in cases authorized by the Code or required by law." This interpretation of Code II.1.c. is critical to the board's final determination that the engineer was unethical in not reporting the safety hazard he discovered. In other words, despite being told that "he must maintain (the structural defects) as confidential," the engineer's primary obligation is to report the structural defects to an authority, as "authorized by the Code," in particular Code II.1.a. dealing with public safety.

As a side note, it is worth briefly mentioning that SIROCCO's output for Case 90-5-1, shown in Figure 1-2, compares favorably with the review board's analysis. First, note that SIROCCO suggests both of the codes cited by the review board in Figure 2-1 (i.e., II.1.a. and II.1.c.), as well as citing additional public safety and confidentiality codes that could be considered relevant. Note also that the program suggests Case 84-5-1 as a possibly relevant case. Finally, SIROCCO correctly identifies the conflict between public safety and confidentiality that exists in this case.

The second example is Case 89-5-1. The fact situation, the ethical questioned raised, and the relevant code references for this case are shown in Figure 2-3. The case involves an important and commonly occurring issue in engineering ethics: a conflict of interest. Engineer A is retained by a construction contractor to perform engineering services for the city of Downstream. The contractor and Downstream become entangled in a financial dispute over the project Engineer A participates in. Engineer A completes his work, but two years later he is contacted by Downstream and asked to assist the city in filing an arbitration claim against the construction contractor. Engineer A agrees to help the city, even though he used to work for the contractor. The question is whether it was ethical for Engineer A to provide claim services to the city of Downstream in this manner.

The most relevant portions of the review board's analysis of Case 89-5-1 are shown in Figure 2-4. The board begins its analysis by stressing, in abstract terms, Engineer A's obligation to tell the truth. This is important to the analysis because the board assumes that part of Engineer A's promised services to Downstream will include his testimony at an arbitration hearing. They discuss the obligation to tell the truth in connection with Code I.3., which requires engineers "to issue public statements only in an objective and truthful manner." This discussion effectively amounts to a *Code Instantiation* of Code I.3., as the review board stresses (in the sentence beginning "Obviously in circumstances where the engineer is providing testimony…") the relevance of this obligation to the particular circumstances of the case at hand, i.e., an engineer who provides official testimony is obligated to tell the truth.

Facts of Case 89-5-1: Conflict of Interest - Claim Services to City

In the early stages of a project, Engineer A, a geophysical engineer, was retained by a construction contractor to make field compaction tests in connection with work to be performed for the city of Downstream. The job specifications stated that the contractor would be responsible for retaining the geophysical engineer with the approval of the city engineer. The frequency of the testing would be determined by the city engineer. During the course of the work, the contractor ran into financial difficulty, alleging that there was excessive testing and that the soil borings did not represent actual conditions, and asked the city for additional funds. Two years later, long after Engineer A's services were completed, the city brought an arbitration action against the contractor. Engineer A was requested by the city to assist the city in developing a claim against the contractor. Engineer A agrees and provides claim review and analysis services for the city.

Question:

Was it unethical for Engineer A to provide claim services to the city?

References:

Code I.3. - "Engineers, in the fulfillment of their professional duties, shall issue public statements only in an objective and truthful manner."

Code II.3.a. - "Engineers shall be objective and truthful in professional reports, statements or testimony. They shall include all relevant and pertinent information in such reports, statements or testimony."

Code II.3.c. - "Engineers shall issue no statements, criticisms or arguments on technical matters which are inspired or paid for by interested parties, unless they have prefaced their comments by explicitly identifying the interested parties on whose behalf they are speaking, and by revealing the existence of any interest the engineers may have in the matters."

Code II.4.a. - "Engineers shall disclose all known or potential conflicts of interest to their employers or clients by promptly informing them of any business association, interest, or other circumstances which could influence or appear to influence their judgment or the quality of their services."

Code III.4.b. - "Engineers shall not, without the consent of all interested parties, participate in or represent an adversary interest in connection with a specific project or proceeding in which the Engineer has gained particular specialized knowledge on behalf of a former client or employer."

Figure 2-3: The Fact Situation, Question Raised, and Relevant Codes in Case 89-5-1

While acknowledging the importance of telling the truth, as exemplified by Code I.3., the board also suggests that there is an overriding consideration in this case. In particular, it suggests that the engineer should not "participate in an adversary interest in connection with a specific project … in which the engineer has gained particular knowledge on behalf of a former client or employer." (see the second sentence of the third paragraph.) This essentially summarizes the conflict of interest that is central to this case. The board appears to apply the operationalization technique *Define the Superior Code* in paragraphs two through five by giving precedence to Code III.4.b. over Code I.3. and concludes that Engineer A's actions were unethical. Although the board does not explicitly state that Code III.4.b. takes precedence over I.3., their intent is clear because (a) they focus their discussion on Code III.4.b. in the final three paragraphs and (b) their final conclusion suggests the overriding importance of III.4.b. The board also suggests that this conflict could have been avoided altogether if Engineer A had not agreed to participate in the legal proceedings or if he had promised not to divulge any information related to the contractor's interest if he did agree to provide testimony.

•••

As technical experts, engineers attempt to provide the most accurate and truthful information available in order to best serve the public interest. This approach is consistent with Code I.3 requiring engineers to issue public statements only in an objective and truthful manner. While the importance of this Fundamental Canon cannot be overstated, as with all provisions of the Code, it must be read in the context of the entire Code. As we have noted on numerous occasions, the Code of Ethics should not be viewed as an absolute, but rather, should be read as a series of interwoven concepts which must be balanced among one another.

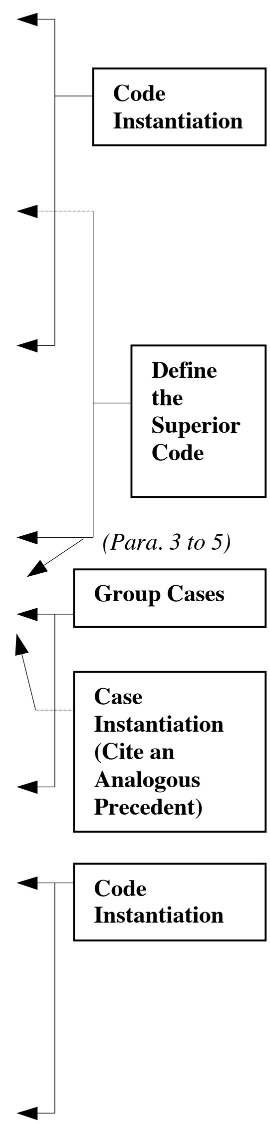
One of the basic ethical dilemmas contained in the Code relates to the duty to be objective and truthful and the obligation to protect privileged and confidential information of a client. This situation frequently arises when an engineer is called upon to provide testimony as an expert witness or in some other capacity for a party who has an interest which is in conflict with that of the engineer's present or former client. Obviously in circumstances where the engineer is providing testimony under oath, the engineer has an ethical and legal obligation to provide honest, truthful and complete testimony to the best of the engineer's ability. To do otherwise would be to act in violation of both the Code of Ethics and the laws which govern our common law system.

Nevertheless, the fact that an engineer is requested to participate in some capacity in connection with a legal or arbitration proceeding does not necessarily require the engineer to either (1) agree to become involved in that legal proceeding, or (2) divulge factual information which may bear upon the interests of a present or former client. These two actions are extremely difficult to distinguish and for that reason, both the Code of Ethics (Code III.4.b.) and this Board (Cases 76-3-1, 82-6-1) have taken them together and determined that it would not be appropriate for an engineer, without the consent of all interested parties, to participate in or represent an adversary interest in connection with a specific project or proceeding in which the engineer has gained particular knowledge on behalf of a former client or employee.

Although both earlier cases interpreted the language contained in Code III.4.b. under circumstances in which the engineer was providing direct testimony in court or statements before a governmental entity, we believe the intent of Code III.4.b. is to cover the broad array of circumstances in which an engineer is called upon to provide assistance to another party whose interests are adverse to those of a present or former client of the engineer. In fact, we believe that the assistance and services provided in the instant case, that of "claims review and analysis" may raise more sinister issues, because unlike testimony in open court or at a public hearing, the assistance offered by Engineer A in an arbitration proceeding is not subject to the rules of civil procedure and evidence.

… While we recognize that under certain circumstances the ethical commitment of the engineer to a former client or a former employer may change, we do not believe under the facts presented in this case that either a sufficient amount of time has passed or that the circumstances have been altered to dilute the ethical obligations owed by Engineer A to his former client, the contractor.

•••



Conclusion:

It was unethical for Engineer A to agree to perform claims review and analysis services for the city.

Figure 2-4: The NSPE BER Analysis of Case 89-5-1 and the Operationalization Techniques Applied

In the third and fourth paragraphs of Figure 2-4, the board applies a *Group Cases* operationalization technique by combining Case 76-3-1 and Case 82-6-1 as examples of "cases (that) interpreted the language contained in Code III.4.b. under circumstances in which the engineer was providing direct testimony in court or statements before a governmental entity." The *Group Cases* operationalization technique is used to support a single illustrative purpose or to apply a greater force to a conclusion. The citation of these cases can also be viewed as two instances of the operationalization technique *Cite an Analogous Precedent*. The circumstances of each case involves very similar facts and issues to that of Case 89-5-1: an engineer is confronted by a conflict of interest in testifying against his or her current or former client. In both of the past cases, the engineer's conduct was found to be unethical and, by analogy, that ruling also supports the conclusion in the present case.

The final operationalization technique that appears to be applied in the analysis of Case 89-5-1 is a *Code Instantiation* of III.4.b. In the final two paragraphs of Figure 2-4, the board extends the application of Code III.4.b. from legal proceedings to arbitration proceedings (and, in fact, to "the broad array of circumstances in which an engineer is called upon to provide assistance to another party whose interests are adverse to those of a present or former client of the engineer"). The board also states, in the last paragraph, that sufficient time has not passed or circumstances have not changed enough "to dilute the ethical obligations owed by Engineer A." In essence, the board links particular facts of Case 89-5-1 (i.e., the fact that Engineer A used to work for the construction contractor, the fact that he likely gained "specialized knowledge" through that work, and the fact that he subsequently performed work for Downstream that is "adversarial" to the contractor's interests) to Code III.4.b. in these paragraphs.

The final example of the application of operationalization techniques is Case 83-1-1, and it is displayed in Figure 2-5. Notice that this case is part of a fact situation that comprises three cases, one for each of three questions raised in the scenario. In Fact Situation 83-1, Engineer A works for Engineer B and is told that he is going to be terminated. After being told of his termination, but before the termination takes effect, Engineer A solicits work from clients of Engineer B. During the same period, Engineer B distributes a brochure listing Engineer A as one of his key employees and continues to distribute the brochure after Engineer A is terminated. The fact situation raises the following three questions: (1) Was it ethical for Engineer A to solicit Engineer B's clients while still in his employ? (Case 83-1-1); (2) Was it ethical for Engineer B to distribute a brochure listing Engineer A as a key employee after he had notified him of his termination but before he was actually terminated? (Case 83-1-2); (3) Was it ethical for Engineer B to distribute the brochure after Engineer A was actually terminated? (Case 83-1-3). The first case, 83-1-1, is discussed in this section, while the second and third cases are discussed in the next section (Section 2.2.3).

Fact Situation 83-1: Conflict of Interest – Duty of Loyalty of Terminated Engineer to Employer – Misleading Brochure

Engineer A worked for Engineer B. On November 15, 1982 Engineer B notified Engineer A that Engineer B was going to terminate Engineer A because of lack of work. Engineer A thereupon notified clients of Engineer B that Engineer A was planning to start another engineering firm and would appreciate being considered for future work. Meanwhile, Engineer A continued to work for Engineer B for several additional months after the November termination notice. During that period, Engineer B distributed a previously printed brochure listing Engineer A as one of Engineer B's key employees, and continued to use the previously printed brochure with Engineer A's name in it well after Engineer B did in fact terminate Engineer A.

Question 1: (Case 83-1-1)

Was it ethical for Engineer A to notify clients of Engineer B that Engineer A was planning to start a firm and would appreciate being considered for future work while still in the employ of Engineer B?

Question 2: (Case 83-1-2)

Was it ethical for Engineer B to distribute a brochure listing Engineer A as a key employee in view of the fact that Engineer B had given Engineer A a notice of termination?

Question 3: (Case 83-1-3)

Was it ethical for Engineer B to distribute a brochure listing Engineer A as a key employee after Engineer A's actual termination?

References:

Code I.4. - "Engineers, in the fulfillment of their professional duties, shall: Act in professional matters for each employer or client as faithful agents or trustees."

Code II.5.a. - "Engineers shall not falsify or permit misrepresentation of their, or their associates', academic or professional qualifications. They shall not misrepresent or exaggerate their degree of responsibility in or for the subject matter of prior assignments. Brochures or other presentations incident to the solicitation of employment shall not misrepresent pertinent facts concerning employers, employees, associates, joint venturers, or past accomplishments with the intent and purpose of enhancing their qualifications and their work."

Code III.3.a. - "Engineers shall avoid the use of statements containing a material misrepresentation of fact or omitting a material fact necessary to keep statements from being misleading; statements intended or likely to create an unjustified expectation; statements containing prediction of future success; statements containing an opinion as to the quality of the Engineers' services; or statements intended or likely to attract clients by the use of showmanship, puffery, or self-laudation, including the use of slogans, jingles, or sensational language or format."

Code III.4.a. - "Engineers in the employ of others shall not without the consent of all interested parties enter promotional efforts or negotiations for work or make arrangements for other employment as a principal or to practice in connection with a specific project for which the Engineer has gained particular and specialized knowledge."

Code III.7. - "Engineers shall not compete unfairly with other engineers by attempting to obtain employment or advancement or professional engagements by taking advantage of a salaried position, by criticizing other engineers, or by other improper or questionable methods."

Figure 2-5: The Fact Situation, Questions Raised, and Relevant Codes in Case 83-1-1

The case presented before the Board raises a number of significant points that have heretofore not been specifically addressed. In BER Case 77-11-1, the Board ruled that four engineers who founded a new firm did not violate the Code of Ethics by generally seeking work from former clients of their previous employer, but were in violation of the Code with regard to projects for which they had particular knowledge while working for their former employer. Although at first glance the facts in Case 77-11-1 appear to be quite similar to the instant case, they are distinguishable on two very important points: (1) In the instant case Engineer A notified "current" and not former clients of Engineer B and offered professional services to them. (2) Engineer A was still employed by Engineer B when Engineer A notified the clients and others of the offer of professional services. We are therefore now asked to decide whether one engineer in the employ of another who is aware of a pending termination may ethically contact "current" clients of an employer and offer professional services to the client without informing the employer.

Case Instantiation (Cite a Distinguishing Precedent)

An engineer is expected to act, at all times in professional matters for the employer, as a faithful agent and trustee (Code I.4.). That requires the engineer to recognize both a duty of loyalty and good faith. An essential aspect of those is the duty to disclose. Certainly it is not possible for an engineer to meet those obligations to the employer if the engineer is engaging in such promotional activity to the employer's detriment. We do not mean to suggest that an employee who severs all ties with the employer and then seeks to contact clients of the employer in order to offer engineering services is in violation of the Code. To the contrary, those were the facts of Case 77-11-1 and that case remains a proper interpretation of the Code. Nor do we wish to suggest any restraint exists upon one's absolute right to select in all cases, the engineer of one's choice. As we noted in Case 77-11-1, "We have often held that (the Code) is not to be interpreted to give an engineer or firm a right to prevent other engineers from attempting to serve former clients of other firms." Nevertheless, for the above-noted reason, it is concluded that Engineer A violated Code I.4. by failing to act as a faithful employee.

Code Instantiation

Another issue related to the conduct of Engineer A is whether Engineer A violated Code III.7. by competing with Engineer B using "questionable methods." It seems obvious that by failing to act as a faithful employee and by failing to disclose the actions to Engineer B, Engineer A engaged in questionable methods of competition. Even if Engineer A was not certain that the actions constituted unethical conduct, Engineer A knew or should have known that they were problematic and dubious and raised the possibility of an ethical violation. Therefore, we are of the view that Engineer A was in violation of Code III.7.

Code Instantiation

A related question under the facts of this case is whether Engineer A violated a duty of disclosure to all interested parties by entering into promotional efforts for work as a principal in connection with work for which Engineer A had gained a particular and specialized knowledge. The facts do not indicate whether Engineer A was attempting to secure work through particular and specialized knowledge gained. Assuming that in fact Engineer A had gained such knowledge and then sought such work without full disclosure to the employer, Engineer B, it appears that Engineer A would have violated Code III.4.a. of the Code. Again, Engineer A owes duties of loyalty, good faith, and disclosure to the employer for which the breach constitutes a violation of the Code. As an employee of Engineer B, Engineer A could not ethically use proprietary information concerning clients, trade secrets, or other valuable information of the employer without full disclosure to the employer.

Apply a Hypothetical to a Code

• • •

Conclusion:

Question 1 (Case 83-1-1). It was unethical for Engineer A to notify clients of Engineer B that Engineer A was planning to start a firm and would appreciate being considered for work while still in the employ of Engineer B.

Figure 2-6: The NSPE BER Analysis of Case 83-1-1 and the Operationalization Techniques Applied

The board's analysis of Case 83-1-1 is shown in Figure 2-6. This analysis involves two operationalization techniques that have not been previously discussed. In the first paragraph of the analysis, the board employs *Cite a Distinguishing Precedent*, a type of *Case Instantiation*. This operationalization technique is used to draw similarities between a present and a past case, but ultimately its purpose is to distinguish the relevant differences between the cases.

The board cites Case 77-11-1 as similar to Case 83-1-1 because it involves ex-employees of a firm who solicit work from former clients of that firm. However, the board also distinguishes the two cases on the basis of two facts: "(1) In (Case 83-1-1) Engineer A notified 'current' and not former clients of Engineer B and offered professional services to them. (2) Engineer A was still employed by Engineer B when Engineer A notified the clients and others of the offer of professional services." These distinguishing facts appear to be critical to the board's conclusion: Engineer A is found to be unethical in soliciting clients of Engineer B (see the bottom of Figure 2-6), while in the past case (77-11-1), the engineers were not found to be unethical in soliciting former clients of their former employer.

The last paragraph of the analysis of Case 83-1-1 contains the other operationalization technique that has not yet been discussed. In this paragraph, the board appears to use the operationalization technique *Apply a Hypothetical to a Code*. This technique is used to establish the range of applicability of a code or to hypothesize unknown but likely facts that help to lead to a conclusion. In this instance, the review board assumes that Engineer A uses "particular and specialized knowledge" gained while performing work for Engineer B in order to solicit work from clients of Engineer B. Such a fact would lead to a violation of Code III.4.a., the conflict of interest code discussed in connection with the last case example.

Two additional operationalization techniques are indicated in the case analysis of Figure 2-6. First, a *Code Instantiation* of I.4. appears to be applied in the second paragraph of the analysis. Here, the board argues that Engineer A's duties of loyalty, good faith, and disclosure – all necessary aspects of his obligation to be a "faithful agent and trustee" (Code I.4.) – are violated by his solicitation of his employer's clients. In other words, the board expands on and interprets Code I.4. and then links the code to the facts of the case that violate it. In particular, the following facts are linked to a violation of Code I.4.: (1) Engineer A was employed by Engineer B and (2) Engineer A solicited Engineer B's clients while still employed by Engineer B. Finally, another *Code Instantiation*, of Code III.7. (see the text of this code in Figure 2-5), appears to be applied in paragraph three of the board's analysis. Here, the board interprets the phrase "questionable methods" to apply to Engineer A's solicitation of Engineer B's clients and hence concludes that Code III.7. is violated. Again, the fact that Engineer A solicits Engineer B while still employed by Engineer B underlies, and is linked to, the code violation.

2.2.3. How Does the Review Board Use Temporal Knowledge?

The examples in the previous section illustrate how the board uses operationalization techniques in their analysis and justification of ethical dilemmas. In this section, several brief examples are presented and discussed to illustrate how the review board uses temporal knowledge and reasoning in their analyses.

First, consider how temporal knowledge can impact an analogy drawn between two cases. Recall the use of the operationalization technique *Cite a Distinguishing Precedent*, discussed in the previous section and depicted in Figure 2-6. Although Cases 83-1-1 and 77-11-1 were deemed by the review board to be similar on a number of basic facts, the cases were also distinguished by two critical comparison points, each of which involved temporal relationships. First, the cases were distinguished because 83-1-1 involved the solicitation of *current* clients versus *former* clients in Case 77-11-1. Second, the engineer in Case 83-1-1 *was still* employed when he solicited these clients, while the engineers in Case 77-11-1 *were not still* employed. In essence, the two cases are distinguishable because of mismatches in the temporal relationships between the facts of the cases.

Temporal reasoning was also critical in the board's evaluation of the actions of Engineer B in Cases 83-1-2 and 83-1-3 (see questions 2 and 3 in Figure 2-5). The question raised in Case 83-1-2 is whether it was ethical for Engineer B to continue to distribute a brochure citing Engineer A as a key employee after Engineer A had been informed of his termination. The question raised in Case 83-1-3 is whether it was ethical for Engineer B to continue to distribute the brochure after Engineer A had actually been terminated.

The board's discussion of Cases 83-1-2 and 83-1-3 is shown in Figure 2-7. The underlined text in the figure corresponds to the board's apparent use of temporal reasoning in analyzing and deciding the two cases. The ovals on the right indicate and point to the board's temporal reasoning that is relevant to each case.

In analyzing these two cases, the board focuses on Codes II.5.a. and III.3.a. (see Figure 2-5 for the text of these codes) and an interpretation of the open-textured terms contained in each. For instance, in evaluating Code II.5.a., the board states that it must interpret "whether (1) Engineer B in fact misrepresented 'pertinent facts' and (2) whether it was the 'intent and purpose' of Engineer B to 'enhance the firm's qualifications and work.' Both prongs must be present for a violation of Code II.5.a. to exist."

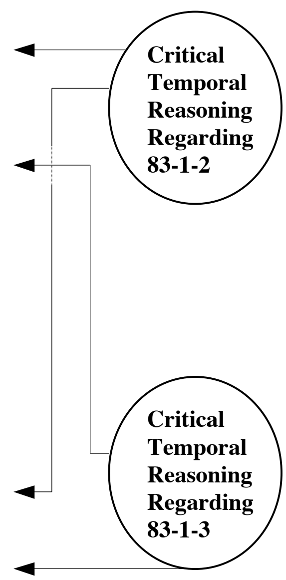
• • •

The other dimension of this case is the actions of Engineer B. Code II.5.a. of the Code specifically states that brochures or other presentations incident to the solicitation of employment shall not misrepresent pertinent facts concerning employers, employees, associates . . . with the intent and purpose of enhancing their qualifications and their work. Thus, the Code provision requires the Board to interpret that provision to determine whether (1) Engineer B in fact misrepresented "pertinent facts" and (2) whether it was the "intent and purpose" of Engineer B to "enhance the firm's qualifications and work." Both prongs must be present for a violation of Code II.5.a. to exist.

"Pertinent facts" are those facts that have a clear and decisive relevance to a matter at hand. Another way to characterize pertinent facts is as those that are "relevant and highly significant." It is not unusual for an engineering firm that seeks to promote itself for business reasons to include in such a brochure a statement of the firm's experience, its history, its qualifications, and the names and qualifications of the members of the firm. The names of the firm's members are often quite significant to the client selecting the firm. The client may be familiar with an individual member of the firm and the selection of that firm may be based on the presence of that engineer in the firm as represented in the brochure. It is clear, therefore, that the inclusion of the name of Engineer A in the firm's brochure constituted a misrepresentation of "pertinent facts."

The second point of inquiry is whether it was the "intent and purpose" of Engineer B to "enhance the firm's qualifications and work" by including Engineer A's name in the promotional brochure after Engineer A had left the firm. The facts presented in the case appear to demonstrate that Engineer B acted with "intent and purpose" in distributing the misleading brochure. Certainly, Engineer B was well aware of the impending termination of Engineer A. Engineer B was the very person who terminated Engineer A. Engineer B distributed the brochure while Engineer A was still employed but had been given a notice of termination by Engineer B. That could easily mislead potential clients into believing that Engineer A, noted as a key employee, would be available in the firm for consultation on future projects. Moreover, Engineer B distributed the brochure after Engineer A had left the firm. That is a clear misrepresentation of a pertinent fact with the intent to enhance the firm's qualifications and as such constitutes a violation of the Code.

Code III.3.a. states in part that "Engineers shall avoid use of statements containing a material misrepresentation of fact or omitting a material fact necessary to keep statements from being misleading; statements intended or likely to create an unjustified expectation...." Although that Code appears to provide Engineer B with the appropriate guidance under the facts in this case, we are of the view that a requirement that Engineer B insert an addendum or an amendment in the brochure informing prospective clients that Engineer A would soon be leaving the firm is both impracticable and unnecessary. That would be a burden to all firms from the standpoint of both time and cost. We do believe that during the interim period between Engineer A's being given notice of termination and his actual cessation of employment, Engineer B had an obligation, during negotiations with a prospective client, to inform the client of Engineer A's pending termination. However, once Engineer A had been formally dismissed, Engineer B had an ethical obligation to cease using the brochure with Engineer A's name in it entirely.



Conclusion:

Question 2 (Case 83-1-2). It was not unethical for Engineer B to distribute a previously printed brochure listing Engineer A as a key employee provided Engineer B apprised the prospective client during the negotiation of-Engineer A's pending termination.

Question 3 (Case 83-1-3). It was unethical for Engineer B to distribute a brochure listing Engineer A as a key employee after Engineer A's actual termination.

Figure 2-7: The NSPE BER Analysis of Cases 83-1-2 and 83-1-3 and the Importance of Temporal Reasoning

Temporal reasoning is used to help interpret and link the facts of the two cases to the open-textured terminology of the codes. For instance, in evaluating the question raised in Case 83-1-2, the board indicates that "intent and purpose to enhance the firm's qualifications and work" is present because Engineer B clearly knew that Engineer A would be terminated; in fact, he was the "very person who terminated Engineer A." Further, the board concludes that by distributing the brochure *during* Engineer A's termination notice period, Engineer B "could easily mislead clients." In deciding this question, the board finds Engineer B's actions to be questionable but not wholly unethical if he apprises prospective clients of Engineer A's pending termination.

The review board is less forgiving in their decision on Case 83-1-3 and, again, this appears to be because of temporal considerations. The board states unequivocally that Engineer B's distribution of the brochure citing Engineer A as a key employee *after* Engineer A left the firm "is a clear misrepresentation of a pertinent fact with the intent to enhance the firm's qualifications and as such constitutes a violation of the Code." In other words, the board believes that the passage of time between the notification that Engineer A was to be terminated and the actual termination is critical to determining Engineer B's ethical obligations in this matter.

Case 89-5-1, discussed in the previous section (see Figure 2-3), also involved temporal reasoning by the review board. In deciding whether it was ethical for Engineer A to provide claim services for the city of Downstream and against a construction contractor for whom he had previously worked, the review board apparently applied a *Code Instantiation* of III.4.b. to the facts of the case to try to resolve the issue (see the bottom of Figure 2-4). As part of this Instantiation, the board considered the question of whether enough time had passed between the completion of the engineer's work for the contractor and the time of the claim services. The general idea was that the engineer's obligation to his former employer could be considered completed if enough time had passed. The board did not believe that sufficient time had passed to make that assertion in Case 89-5-1. Thus, it appears that in some situations the board considers durations of actions and events and how these impact the applicability of codes.

Finally, consider Case 76-5-1, shown in Figure 2-8. In this case, Engineer A provides engineering services for the XYZ Development Company in connection with a subdivision. XYZ declares bankruptcy and pays Engineer A a portion of the payment for his services. PDQ buys the subdivision property and decides to make major changes to the original plans. Engineer A negotiates with PDQ to update the plans, but the negotiations collapse when Engineer A claims that he should be paid the remainder of money owed to him by XYZ from the original project. After terminating negotiations, PDQ retains Engineer B to revise the plans. The question raised in this case is whether it was ethical for Engineer B to accept the assignment under these circumstances.

> *Facts of Case 76-5-1: Supplanting - Use of Second Engineer After Bankruptcy of Client*
>
> The XYZ Development Company retained Engineer A for engineering services related to a subdivision project. During the course of the work by Engineer A the XYZ Development Company was declared bankrupt and underwent bankruptcy proceedings. Engineer A filed a claim for payment for work done along with other creditors and was paid a proportion of the amount due him in accordance with the bankruptcy procedure. The PDQ Development Company bought up the property and plans to develop it with some major changes from the original development plan. Engineer A contacted PDQ to offer a continuation of his services.
>
> During the negotiations between Engineer A and the PDQ firm, Engineer A claimed that under the agreement with PDQ he should be paid for his original work for the XYZ firm over and above the amount received from the bankruptcy proceeding. The PDQ firm refused this condition and negotiations were terminated. The PDQ firm then retained Engineer B to proceed with the engineering services to complete the revised development plan. Engineer A contends that Engineer B was unethical by his action in taking the assignment under these circumstances.
>
> *Question:*
>
> Was Engineer B unethical in taking the engineering assignment under the circumstances?
>
> *References:*
>
> *Code 11(a).* - "The Engineer will not attempt to supplant another engineer in a particular employment after becoming aware that definite steps have been taken toward the other's employment."

Figure 2-8: The Fact Situation, Question Raised, and Relevant Code in Case 76-5-1

At the time this case was decided, Code 11(a) stipulated that an engineer "will not attempt to supplant another engineer in a particular employment after becoming aware that definite steps have been taken toward the other's employment." The analysis of the review board regarding this case, shown in Figure 2-9, focuses on this code and its application in the context of the current case. The board discusses and analyzes the temporal aspects of the code by referring to a past case, 62-18, that states that "the 'supplanting' rule cannot apply when the client has terminated the services of the engineer before retaining another." The board then applies these temporal constraints to the facts of Case 76-5-1 by stating that "when Engineer B entered the scene, Engineer A had no contract with PDQ and negotiations had been terminated. Thus, the client was free to turn to another engineer." For this reason, the board ultimately concludes that Engineer B was not unethical in taking the engineering assignment from PDQ.

In summary, these examples illustrate that the NSPE BER sometimes employs temporal reasoning in order to analyze and decide engineering ethics dilemmas. Temporal knowledge can be helpful in determining when ethics codes apply and when analogies across cases are valid.

In previous cases we have made it abundantly clear that §11(a) does not give an engineer an exclusive right or claim on a particular client or for a particular project. As we said as early as Case 62-10-1, and affirmed in Case 73-7-1, "there can be no question but that the client has a right to change from one consulting engineer to another." And we have also noted that the "supplanting" rule cannot apply when the client has terminated the services of the engineer before retaining another (Case 62-18-1). Also, in Case 64-9-1 we restated the principle that for the "supplanting" standard to apply the facts must demonstrate that the complaining engineer either had a contract for the work, or had been selected for negotiation by the client for the particular work, citing a similar result in Case 62-18-1. (For other supplanting instances see Cases 59-2-1 and 65-8-1.)

Applying these same observations to the facts at hand, it is clear that when Engineer B entered the scene, Engineer A had no contract with PDQ and negotiations had been terminated. Thus, the client was free to turn to another engineer.

Critical Temporal Reasoning Regarding 76-5-1

It is not material to the ethical question that Engineer A had not received full 71-10-1 payment for his previous services. In Case we made the point that nothing in §11(a) expresses the idea that the ethical standard should turn on whether or not the engineer had been paid his full fee. Even if that standard were to be applied (as contended by two dissenting members in Case 59-2-1 if the owner terminated the engineer for unjust causes) the rationale would not apply here because Engineer A had been paid "in full" to the extent that the law provided. In other words, the very purpose of the bankruptcy proceeding is to wipe out all debts by whatever amounts can be recovered and paid from the assets of the bankrupt.

Accordingly, Engineer A did not in fact have any further legal claim against the original client, nor certainly did he have any legal claim against the PDQ firm, never having had a contract with it. He was free, of course, to endeavor to have the PDQ firm pick up the part of his fee which was wiped out by the bankruptcy, but when that effort failed he had no further standing to prevent the PDQ firm from proceeding with a new engineer for its purposes.

Conclusion:

Engineer B was not unethical in taking the engineering assignment under the circumstances.

Figure 2-9: The NSPE BER Analysis of Case 76-5-1 and the Importance of Temporal Reasoning

Representing the Temporal Information: A Brief Preview of SIROCCO's Representation

As we shall see in the next chapter, SIROCCO's representation language allows a case enterer to specify both case facts and the temporal relationships between those facts. For instance, the representation of Case 83-1-1 (Figure 2-5), as provided by a case enterer, includes the facts: (1) Engineer A is employed by Engineer B; (2) Engineer B has clients; (3) Engineer B informs Engineer A of his termination; and (4) Engineer A solicits Engineer B's clients[12]. SIROCCO's temporal representation also allowed the case enterer to specify that fact (4) occurred *during* facts (1) and (2). That is, Engineer A solicited Engineer B's clients *while* he was still employed by Engineer B, the crux of the ethical issue in Case 83-1-1. This issue is essentially captured by *Code Instantiations* of Codes I.4. and III.7., which are linked to facts (1),

[12] Appendix B, section B-5 presents a detailed discussion of how fact situation 83-1 was transcribed into the representation language.

(2), and (4) in the case enterer's representation. The other examples discussed in this section were similarly represented as facts, temporal relationships between those facts, and Instantiations.

However, as we shall see in Chapter 4, the capturing of facts and temporal relationships in Instantiations, as illustrated above, did not make an overall impact on SIROCCO's capability to retrieve codes and past cases. Why did this occur when the examples in this section – as well as a more detailed example of SIROCCO to be presented in Section 3.3 – seem to indicate that temporal knowledge *is* important? A number of possible reasons will be proposed and discussed. For instance, perhaps the case enterers were not sensitive to the subtleties required to specify temporal relationships. Perhaps temporal knowledge is not appropriately emphasized in the representation: temporal relationships are only implicitly part of an Instantiation, i.e., as "background" knowledge about the facts that are linked to cited codes and cases, and they are all treated equally. These – and other – possibilities will be discussed in Section 4.3.2.

2.2.4. Limitations of the NSPE BER Codes and Cases

In closing the discussion of the NSPE BER subdomain, it is important to note that the NSPE BER codes and cases are not without drawbacks [Harris *et al.*, 1999, p. 16-17]. First, the NSPE BER case analyses tend to focus exclusively on the NSPE codes. There is little or no discussion of more general principles, such as honesty, fairness, and veracity. Second, there are very few dissenting opinions in the analyses of NSPE BER cases. Almost every case represents a consensus of the board. There are times in ethical analysis when disagreement and unresolved issues are more important than reaching consensus. Finally, most of the NSPE BER cases focus on issues faced by independent consulting engineers rather than engineer employees of corporations. Also, the cases are mostly related to civil engineering and certain important topics, such as "whistle blowing," are sparsely represented.

On the other hand, because the NSPE BER cases tend to be ordinary – in contrast to dramatic and highly publicized cases such as the Challenger case discussed in Chapter 1 – most engineers can relate to these ethical dilemmas from their everyday practice. Also, the analyses of the board are usually clear, concise, and carefully reasoned. Finally, the NSPE BER cases potentially provide a vital learning resource to the engineering community. The vitality and usefulness of the published cases could be further enhanced by providing intelligent access to the cases, as proposed and implemented in this dissertation. Such access might help stimulate discussion and educate engineers far more effectively than access to a static database would.

3. SIROCCO: Retrieving and Analyzing Cases, Codes, and Operationalizations

SIROCCO was developed to test the primary and secondary theses of the dissertation and to determine how well the operationalization techniques could be employed computationally to improve retrieval and organize the resulting information. Ultimately, SIROCCO is intended to be an intelligent assistant to users who are charged with the responsibility of analyzing and deciding engineering ethics cases. The program is designed to retrieve information – past cases, codes, and operationalizations – and provide that information so that the user can develop an argument or rationale for a particular conclusion to an ethical dilemma.

The previous two chapters introduced SIROCCO's high-level architecture and explained how the board of ethical review decided cases and justified conclusions. Tying these two threads together, this chapter presents a detailed view of SIROCCO's architecture. SIROCCO's design is greatly influenced by and utilizes the reasoning techniques that were directly observed and inferred from the board's case analyses.

After presenting the details of SIROCCO's operation, illustrated by an extended example, the chapter then focuses on how the most critical features of the program are implemented. In particular, it describes how the operationalization techniques inform and are implemented within the architecture, how temporal knowledge is used during SIROCCO's retrieval, and how the program explains its reasoning,

3.1. SIROCCO's Architecture

A general overview of SIROCCO's architecture was first presented in Section 1.3.3. In this section, I briefly review the architecture and illustrate it in greater detail by following a trace of SIROCCO processing an example case.

A diagram of SIROCCO's architecture is shown in Figure 3-1. Briefly to recap, the program begins retrieval and analysis by accepting a new engineering ethics case, called the *target case*, that has been transcribed into the Ethics Transcription Language. ETL represents a case as a chronology of Facts, each of which describes an action or event that occurred in the scenario. Each Fact is composed of a Fact Primitive, a verb phrase that describes an action or event, a set of Actors or Objects that execute or are acted upon by the Fact Primitive, and a Time Qualifier, which describes how that Fact is related to other Facts in time. In addition to the target case, SIROCCO accepts a series of parameters that customize the behavior of each phase of the algorithm.

The program's primary goal is to find the cases, codes, and operationalizations that are most relevant to the target case. All of this information is accessible to SIROCCO through its case base (i.e., the *source cases*), shown in the upper right portion of Figure 3-1. The source cases, have been transcribed into the Extended Ethics Transcription Language. This superset of ETL provides elements to model the board's rationale. Each source case may be linked to relevant cases by case operationalizations and to relevant codes by code operationalizations (see the dashed arrows in Figure 3-1). The operationalizations represent the fundamental components used by the board to argue or rationalize their decision in a source case.

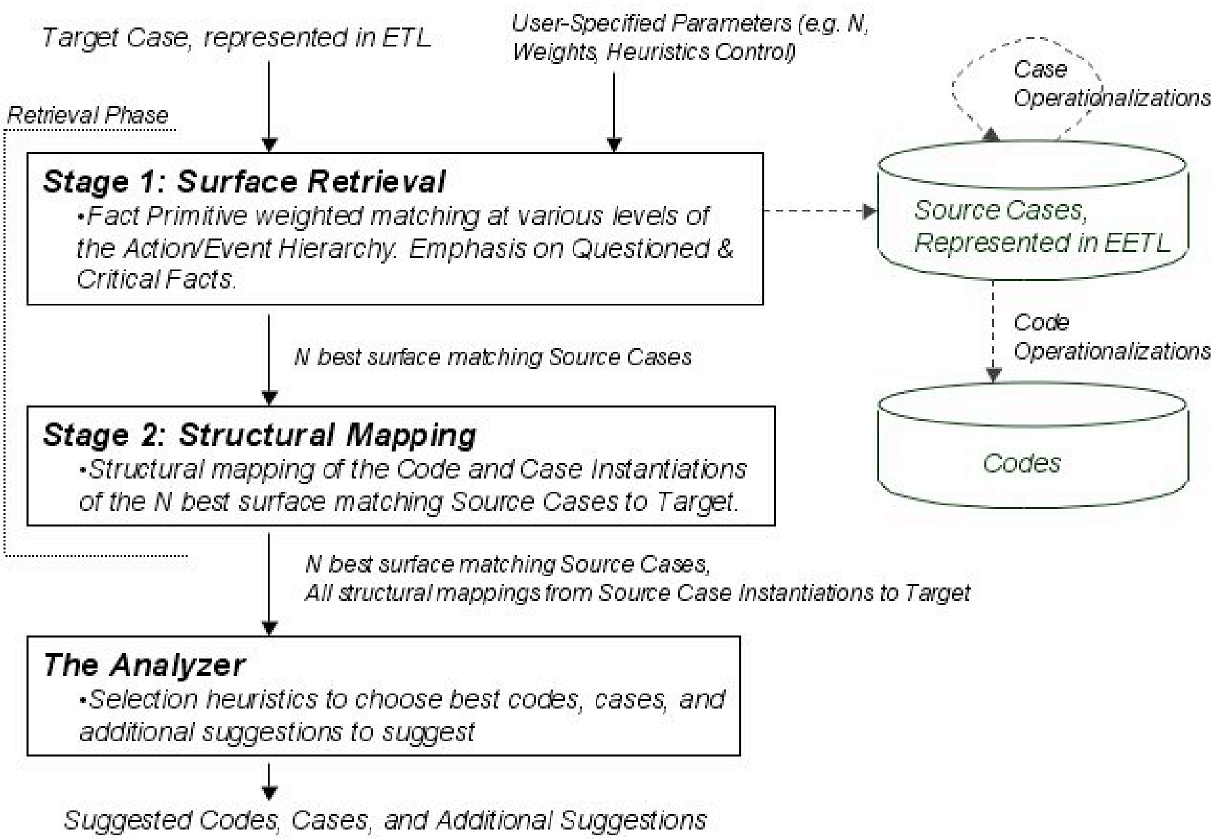


Figure 3-1: SIROCCO's Architecture

The program begins by retrieving a set of source cases that match, at least superficially, to the target case. This is achieved by retrieving and scoring all cases that contain at least one of the target's Fact Primitives, or an abstraction of one of the primitives. Each source case is essentially scored by the degree to which its Fact Primitives overlap the Fact Primitives of the target. Fact Primitives are cast in a hierarchy known as the Action/Event Hierarchy and matching between source and target may occur at higher levels of abstraction in that structure. The source cases are also scored by the degree to which their Critical and Questioned Facts match the Facts of the target case. The scoring formula is discussed in detail later in this chapter.

After scoring all of the source cases, Stage 1 of SIROCCO passes the top *N* cases to a second stage of retrieval, known as structural mapping, which performs a graph mapping of the target case to each of the *N* cases, using A* heuristic search. SIROCCO's graph-mapping operation does not encompass all Facts found in the target and source; not only would this be computationally expensive, it would also ignore important operationalization knowledge, represented in EETL, that certain Facts are more critical than others. Instead, the graph-mapping operation focuses on mapping the target case to the *Code Instantiations* and *Case Instantiations* found in the top-rated source cases. As first defined in Section 1.3.3 and Figure 1-1, an Instantiation is a type of operationalization which relates the Questioned Fact, certain critical Facts, and the temporal sequence of those Facts to a code or case citation.

The structural mappings from the target to the Instantiations of each source case in Stage 2 are scored and sorted according to percentage of match and are passed to the next phase of the program, the Analyzer. The Analyzer selects and displays the final output of the program (i.e., suggested codes, cases, and operationalizations) by using a separate set of heuristics for each information type. In making the program's final selections, the heuristics take into account the results of both Stage 1 and Stage 2, as well as other operationalization information associated with the top-rated cases.

SIROCCO was developed in Common Lisp on a Power Macintosh. Loom [MacGregor, 1990; Brill, 1993], a knowledge representation language that runs under Common Lisp, was used to represent all of the components of EETL, as well as the Action/Event Hierarchy and Code Hierarchy.

3.1.1. An Example Target Case and Some Potentially Relevant Source Cases

To illustrate the operation of SIROCCO, the processing of a target case example is traced in its entirety in this chapter. The example case is 90-5-1, the public safety scenario originally introduced in Chapter 1 (in Figure 1-2) and shown in its entirety in Chapter 2 (in Figure 2-1 and Figure 2-2). For convenience, the facts of the example case are reprinted in Figure 3-2.

Facts of Case 90-5-1:

Tenants of an apartment building sue the owner to force him to repair many defects in the building that affect the quality of use. The owner's attorney hires Engineer A to inspect the building and give expert testimony in support of the owner. Engineer A discovers serious structural defects in the building, which he believes constitute an immediate threat to the safety of the tenants. The tenants' suit has not mentioned these safety-related defects. Upon reporting the findings to the attorney, Engineer A is told he must maintain this information as confidential as it is part of a lawsuit. Engineer A complies with the request of the attorney.

Question:

Was it ethical for Engineer A to conceal his knowledge of the safety-related defects in view of the fact that it was an attorney who told him he was legally bound to maintain confidentiality?

Figure 3-2: The Example Target Case, 90-5-1

In Case 90-5-1, Engineer A is confronted with conflicting obligations. He has an obligation to protect the public, but he also has an obligation of confidentiality to his client. Engineer A discovers serious safety problems in a building he is asked to inspect, but he does not report the safety problems to anyone because his client requests confidentiality. As discussed in Chapter 2, the review board concluded that Engineer A's action was unethical, because his obligation to public safety outweighed his obligation of confidentiality in these circumstances. The board justified their conclusion by applying, for example, the code operationalization *Define the Superior Code* (arguing that the public safety code II.1.a. takes precedence over the confidentiality code II.1.c.) and the case operationalization *Cite an Analogous Precedent* (drawing an analogy to the source case 84-5-1).

Three source cases that are potentially relevant to Case 90-5-1 are depicted in Figure 3-3[13]. The first case, 76-4-1, is the one most strikingly similar to the target. In this case, Engineer Doe is hired to inspect waste produced by a manufacturing plant. As in Case 90-5-1, the engineer does, in fact, discover that a safety hazard exists, i.e., the manufacturing waste is below established standards. Doe's client, the XYZ Corporation, terminates his contract and instructs him "not to render a written report." As in the target case, Doe does not write a report or contact any outside authority about his findings, presumably to protect his client's confidentiality.

The second case, 99-1-1, also involves a scenario in which an engineer, Engineer Smith, performs an inspection for a client, discovers that a safety hazard exists, and is told by his client not to report his findings. However, in several respects, this case is fundamentally different from the target case and source Case 76-4-1. First, unlike Cases 90-5-1 and 76-4-1, Engineer Smith proceeds to report the safety hazard to the appropriate authority, the Environmental Protection Agency (EPA). Second, there appears to be an issue of competence involved in this scenario. It is stated that Smith has "limited experience" in environmental engineering, the field in which he renders an evaluation. Third, and most critically, the case does not raise a direct question about Engineer Smith's actions. Rather, it questions the actions of Engineer X, a senior engineer with the EPA, who handles Smith's report. Engineer X believes, perhaps correctly, that Smith's analysis is faulty, and he makes harshly critical comments about Smith and his methodology to colleagues at the EPA. Thus, the question in this case is whether it is ethical for Engineer X to publicly criticize Engineer Smith, a fundamentally different issue from the conflict between public safety and confidentiality raised in Cases 90-5-1 and 76-4-1.

The final case, 84-5-1, is similar to the target in the sense that an engineer's judgment is overruled by a client in a situation in which safety is at issue. Confidentiality, however, is not as central an issue to this case as it is to Cases 90-5-1 and 76-4-1. Here, an engineer develops plans for a project and, because of the potential dangers involved in the project's construction phase, recommends to his client that an "on-site representative" be hired to oversee the work. The client rejects this proposal because of costs, yet the engineer proceeds with the project nonetheless. In this situation, it is not clear that a standard or law has been or will be violated, so there is seemingly no compelling obligation for the engineer to keep "confidences," as in Cases 90-5-1 and 76-4-1. However, the fact that the engineer's judgment is overruled, resulting in potential public harm, is highly relevant. In fact, as shown in Figure 2-2 of Chapter 2, this case was actually cited by the board as an analogous precedent in its analysis of Case 90-5-1.

Throughout this chapter the example target and source cases are employed to illustrate the representation and functionality of SIROCCO. As we will see, the similarities and differences between the example target case and source cases, described above, will factor prominently in SIROCCO's similarity assessment. Case 76-4-1 will be correctly recognized as highly similar to the target throughout SIROCCO's retrieval and analysis algorithm. On the other hand, Case 99-1-1 will initially be considered as a good match by Stage 1 but will ultimately, and appropriately, be rejected later in that stage. Finally, Case 84-5-1 will initially be rejected by Stage 1 but then later selected by the Analyzer as a potentially relevant case.

---

[13] Cases 76-4-1 and 84-5-1 are actual cases from the NSPE BER corpus of cases. Case 99-1-1 is a hypothetical variation of case 76-4-1, devised to illustrate how changes to critical aspects of a scenario can change SIROCCO's processing of that scenario.

Facts of Case 76-4-1:

Facts: The XYZ Corporation is advised by a State Pollution Control Authority that it has 60 days to apply for a permit to discharge manufacturing wastes into a receiving body of water. XYZ is also advised of the minimum standard that must be met.

In an effort to convince the authority that the receiving body of water will still meet established environmental standards after receiving the manufacturing wastes, the corporation employs Engineer Doe to perform consulting engineering services and submit a detailed report.

After completion of his studies but before completion of any written report, Doe concludes that the discharge from the plant will lower the quality of the receiving body of water below established standards. He further concludes that corrective action will be very costly. Doe verbally advises the XYZ Corporation of his findings. Subsequently, the corporation terminates Doe's contract with full payment for services performed, and instructs him not to render a written report to the corporation.

Thereafter, the control authority calls a public hearing, and the XYZ Corporation presents data at the hearing to support its view that the present discharge meets minimum standards. Doe learns of the hearing and XYZ's presentation but does not report his earlier contradictory findings to the authority.

Question:

Was it ethical for Doe not to report his findings to the authority upon learning of the hearing?

Facts of Case 99-1-1:

The ABC Corporation hires Engineer Smith as a consultant to inspect their facility and assure that its manufacturing waste meets Environmental Protection Agency (EPA) standards. Engineer Smith has limited experience in environmental engineering.

After completion of his inspection, Smith concludes that smog from ABC's plant violates the Clean Air Act. Smith verbally advises the ABC Corporation of his findings. Subsequently, the corporation terminates Smith's contract, pays him in full, and instructs him not to render a written report. Engineer Smith does not write a report, but he does contact the EPA about his findings.

After being contacted by Smith, the EPA performs an independent inspection and determines that, in fact, the ABC Corporation's discharge does meet established standards. Engineer X, a senior engineer for the EPA, makes harshly critical comments to his colleagues at the EPA about Engineer Smith and the methodology he applied in his inspection.

Question:

Was it ethical for Engineer X to make critical comments about Engineer Smith to his colleagues?

Facts of Case 84-5-1:

The client plans a project and hires Engineer A to furnish complete engineering services for the project. Because of the potentially dangerous nature of implementing the design during the construction phase, Engineer A recommends to the client that a full-time, on-site project representative be hired for the project. After reviewing the completed project plans and costs, the client indicates to Engineer A that the project would be too costly if such a representative were hired. Engineer A proceeds with his work on the project.

Question:

Was it ethical for Engineer A to proceed with his work on the project knowing that the client would not agree to hire a full-time project representative?

Figure 3-3: Three Example Source Cases, 76-4-1, 99-1-1, and 84-5-1

3.1.2. Knowledge Representation in SIROCCO: An Ontology of Engineering Ethics

SIROCCO cannot process the natural language descriptions of input target cases such as 90-5-1. Instead, a web site (www.pitt.edu/~bmclaren/ethics) was devised to enable untrained (but graduate-level) persons to transcribe cases into the Extended Ethics Transcription Language, the knowledge representation language of SIROCCO's cases. The web site contains a Participant's Guide with instructions on how to transcribe ethics cases into EETL and a Reference Shelf of useful materials, including the full vocabulary of EETL and an example set of 47 transcribed fact situations. A comprehensive description of the web site, along with a full reprint of the Participant's Guide, is provided in Appendix B.

The Extended Ethics Transcription Language essentially provides an ontology of the engineering ethics domain. It includes the basic components and structures for representing the factual description of a case (including the raw facts, the temporal relationships between facts, the types of actors and objects involved, and the question raised by the case) and an analysis of a case. The Ethics Transcription Language is a subset of EETL that provides the components necessary for the factual description of a case only. A target case presented to SIROCCO for analysis must be represented in ETL (i.e. a factual description only), while a source case in SIROCCO's case base must be represented in full EETL (i.e., a factual description *and* an analysis representation). The constraints on the ontology are defined by the ETL grammar (see Figure 3-5 below) and the valid attribute values of an analysis representation (see "The Extended Ethics Transcription Language (EETL)," below, and Appendix sections B7, B8, and B9). Other critical structures in SIROCCO's ontology are two abstraction hierarchies, the *Action/Event Hierarchy* and the *Code Hierarchy*.

In addition to the distinction between EETL and ETL, there is a representational distinction between fact situations and cases in SIROCCO. A fact situation is a representation of the actions and events of a scenario but is not focused on a *particular* questioned action or event of that scenario. A case, on the other hand, is a fact situation with a particular fact selected as the questioned action or event. Thus, if there are multiple questions raised in one scenario, a fact situation may have several associated cases. Note that it is typically more convenient to refer to cases, so fact situations are mentioned and discussed in the dissertation only when it is necessary to do so. Figure 3-4 provides a general schematic of the use of EETL and ETL and the relationship between target cases, source cases, and fact situations.

This section begins with a description of ETL, followed by a discussion of EETL, an introduction to the Action/Event Hierarchy and the Code Hierarchy, and, in conclusion, a discussion of the elements of the representation that were ultimately not used by SIROCCO's algorithm. Throughout, the example target case, 90-5-1, is used to illustrate the representation.

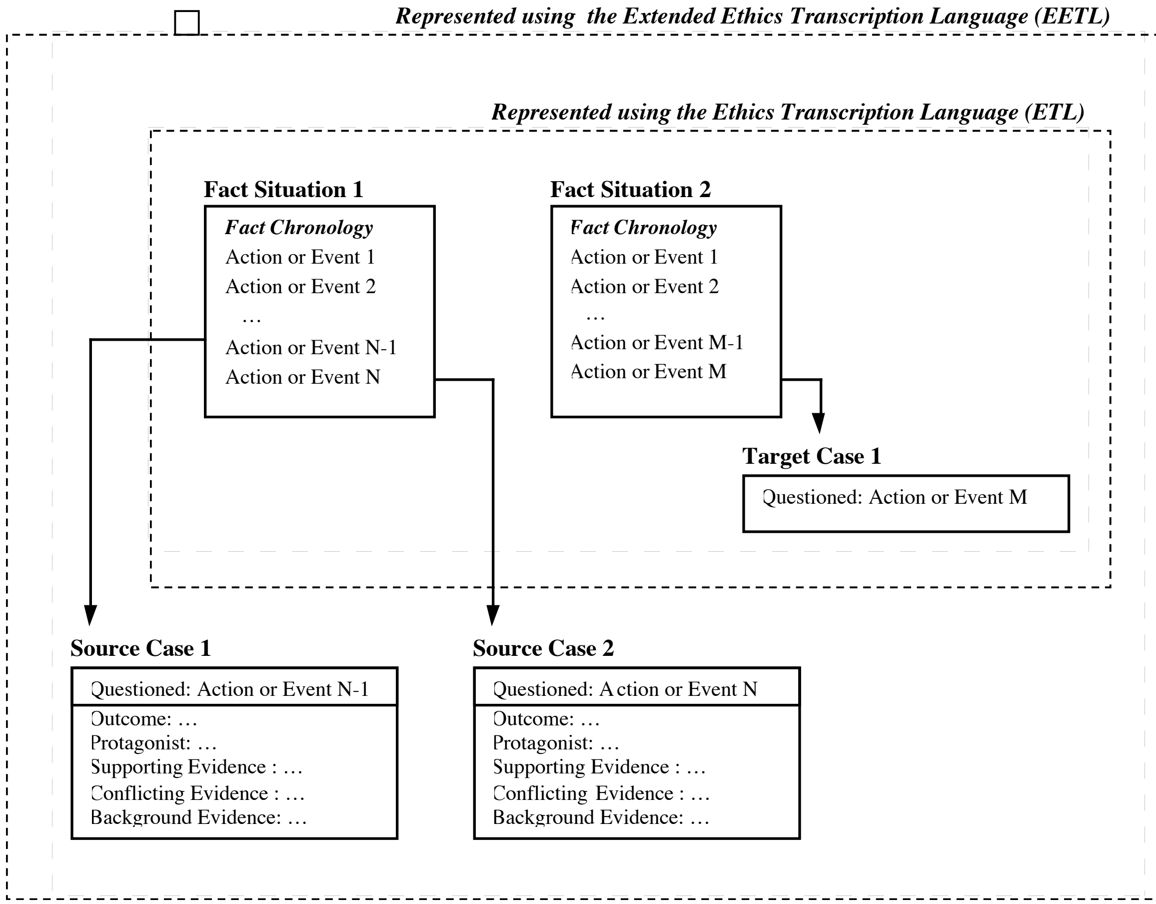


Figure 3-4: The Use of EETL and ETL and the Relationship between Fact Situations and Cases

The Ethics Transcription Language (ETL)

The standard vocabulary of the Ethics Transcription Language comprises: (1) *Actor and object Types*, a list of the types of actors and objects that may appear in the engineering ethics scenarios; (2) *Fact Primitives*, a list of the actions and events in which the actors and objects may participate; and (3) *Time Qualifiers*, a list of temporal relations that specify how the actions and events relate to each other in time. There are presently 70 Actor and Object Types, 190 Fact Primitives, and 10 Time Qualifiers in the definition of ETL. The entire set of Actors, Objects, Fact Primitives, and Time Qualifiers in the current version of SIROCCO is provided in Appendix A.

ETL allows a case to be described as an ordered list (i.e., the Fact Chronology) of short sentences, each of which is a Fact and satisfies the grammar of a <Fact> as shown in Figure 3-5.

<Fact-Chronology> := <Fact> [<Fact> …]
<Fact> := <Fact-#> <Fact-Phrase> [(Questioned Fact <X>)]
<Time-Qualifier> [,<Time-Qualifier>, ...]
<Fact-Phrase> := <Fact-Primitive> [<Fact-Modifier>] <Actor-Or-Object>
[<Actor-Or-Object> | (<Fact-Phrase>)]
[<Actor-Or-Object> | (<Fact-Phrase>)]
<Fact-#> := <Positive-Integer>
<Fact-Primitive> := *An instance of a Fact-Primitive*
<Actor-Or-Object> := *An instance of an Actor or an Object*
< Fact-Modifier> := partially | substantially | limited | extensive
<Time-Qualifier> := Pre-existing fact |
After the start of <Fact-#> [, <Fact-#>, ...] |
Starts at the same time as <Fact-#> [, <Fact-#>, ...] |
<Time-Period> after the start of <Fact-#> [, <Fact-#>, ...] |
After the conclusion of <Fact-#> [, <Fact-#>, ...] |
Immediately after the conclusion of <Fact-#> [, <Fact-#>, ...] |
<Time-Period> after the conclusion of <Fact-#> [, <Fact-#>, ...] |
Ends <Fact-#> [, <Fact-#>, ...] |
Occurs during <Fact-#> [, <Fact-#>, ...] |
Occurs as part of <Fact-#> [, <Fact-#>, ...] |
Occurs concurrently with <Fact-#> [, <Fact-#>, ...] |
<Time -Period> := <Y> Days | <Y> Weeks | <Y> Months | <Y> Years
<X> := Empty | <Positive-Integer>
<Y> := Many | Several | <Positive-Integer>
<Positive-Integer> := 1 ... N

Key: | = Alternative; [ ] = Optional; < > = Grammar element
Regular font indicates literal placement of language (e.g., "Pre-existing fact ")
Italicized font indicates a general description (i.e., "*An instance of a Fact-Primitive* ")

Figure 3-5: The Grammar for the Ethics Transcription Language (ETL)

In essence, the Fact Primitive in each Fact Phrase is a verb phrase that indicates a specific action or event involving actors, objects, or similarly constituted Fact-Phrases. It is treated, in effect, like a function with up to three arguments (Fact-Primitive arg1 [arg2] [arg3]), where arg1 is the Actor-Or-Object serving as the subject of the verb phrase. An important objective was to allow the case enterers to transcribe the cases in as natural a way as possible. Thus, in the Fact Chronologies, human case enterers put arg1 before the Fact Primitive as they would the subject of a verb in a sentence. Each of the Facts is listed in a table in approximate chronological order as indicated by its Fact-#. Time Qualifiers specify additional detail about the chronological ordering.

A Fact Chronology for Case 90-5-1 is shown in Figure 3-6. The corresponding table of actors and objects is shown in Figure 3-7. The case enterer has designated Fact 12 as the Questioned Fact; this fact corresponds most closely to the question posed by the board in the natural language version of Case 90-5-1 (Figure 3-2). The web site's Participant's Guide, which is reproduced in its entirety in Appendix B, provides instructions on how to perform this task.

1. Apartment Building **<may be hazardous to safety>**.	Pre-existing fact
2. Apartment Building Owner **<owns>** Apartment Building.	Occurs during 1
3. Residents of Apartment Building **<reside in>** Apartment Building.	Occurs during 1, 2
4. Residents of Apartment Building **<file a lawsuit or arbitration action against>** Apartment Building Owner **<because>** (Apartment Building **<may be hazardous to safety>**).	Occurs during 3
5. Apartment Building Owner **<is legally represented by>** Owner's Attorney**.**	Occurs during 4
6. Owner's Attorney **<hires the services of>** Engineer A **<for>** (Engineer A **<inspects>** Apartment Building).	Occurs during 4, 5
7. Engineer A **<inspects>** Apartment Building.	Occurs during 6
8. Engineer A **<discovers that>** (Apartment Building **<fails standards and may be hazardous to safety.>**)	Occurs during 7
9. Engineer A **<knows>** (Government Authority **<should be informed about the hazard or potential hazard>**).	Occurs during 8
10. Engineer A **<informs>** Owner's Attorney **<that>** (Apartment Building **<fails standards and may be hazardous to safety.>**)	Immediately after the conclusion of 8
11. Owner's Attorney **<instructs>** Engineer A **<to>** (Engineer A **<withholds information from>** Anyone Else **<regarding>** Apartment Building).	After the conclusion of 10
12. Engineer A **<does not inform>** Anyone Else **<that>** (Apartment Building **<fails standards and may be hazardous to safety.>**) *[Questioned fact]*	After the conclusion of 11

Figure 3-6: The Fact Chronology of Case 90-5-1

1. Apartment Building –> **House.**
2. Apartment Building Owner –> **Client.**
3. Residents of Apartment Building –> **General Public.**
4. Owner's Attorney –> **Attorney.**
5. Engineer A –> **Engineer.**
6. Anyone Else –> **General Public.**
7. Government Authority –> **Government Authority.**

Figure 3-7: The Actors and Objects in Case 90-5-1

Fact Primitives have alternate forms (e.g., inverse, plural, negative) that allow the case enterer to choose the appropriate expression for specific situations. For instance, in step 12 of Figure 3-6, the negative form of the Fact Primitive "… informs … that …" is employed. All of the alternate forms are linked to the *base form* of that Fact Primitive, and SIROCCO treats the alternate forms as equivalent to the base form for purposes of initial retrieval[14].

[14] Note, however, that a positive-to-negative mapping between target and source can be specified by the user to yield a lesser match score during structural mapping.

Fact Primitives are of three types: Events, States, and Terminating Events, to ensure that the Time Qualifiers are used in a consistent manner. Events have relatively short duration. States have relatively long duration. Terminating Events are special events that typically end a State. In selecting Fact Primitives, case enterers are instructed to take the primitive's type into consideration. For example, <is employed by> in Fact 3 in Figure 3-6 is a State primitive. It properly represents a relatively long term of employment during which some of the other events of this case occur. If the case facts had focused on the events of being offered and accepting employment, then certain Event Fact primitives would be more appropriate (e.g., <is offered employment by>, <accepts an offer of employment from>).

Fact Modifiers may be attached to Fact Primitives to indicate partial participation in states or events. "Partially" and "substantially" indicate that an actor was responsible for a quantifiable portion of the fact (i.e., either less than half or more than half, respectively). "Limited" and "extensive" indicate that the actor was responsible for a portion of the fact that can be expressed only in a qualitative way.

After the case enterer has identified the Facts and marked the Questioned Fact or Facts, he or she assigns Time Qualifiers to clarify the chronological relationships between each Fact and other Facts in the Fact Chronology. The Time Qualifiers provided in ETL are those that appeared to be the most common and representative of the engineering ethics scenarios, according to the engineering ethics case study. In addition, an attempt was made to provide Time Qualifiers that are naturally expressed by humans. Each of the 10 Time Qualifiers represents a disjunctive group of one or more of Allen's temporal relations [1983] and, as we shall see during the discussion of SIROCCO's Stage 2 retrieval in Section 3.1.4, the Allen relations provide a formal way to propagate temporal relations throughout a chronology and thus to use those relations in the structural mapping algorithm. Figure 3-8 provides a table of all of the Time Qualifiers and their mappings to Allen's temporal relations.

In order to make temporal specification easier for case enterers, Allen's temporal relations weren't used directly. First, it was believed that some of the Allen relations aren't typically or naturally expressed by humans when they discuss time periods. For instance, how often does one say that event A "meets" event B? Second, using the relations directly would have forced the case enterers to provide disjunctive groups of relations between Facts, a difficult and unnatural task, instead of providing single relations between Facts as with the Time Qualifiers. Of course, abstracting the temporal relations introduces some imprecision. On balance, however, this appeared to be an important and necessary trade-off to achieve the goal of having independent case enterers transcribe the cases.

SIROCCO's Time Qualifiers	Allen's Temporal Relations
Pre-existing fact[15]	B, C, Fi, M, O
After the start of ...	A, D, F, Mi, Oi
Starts at the same time as ...	S, Si
<X time> after the start of ...	D, F, Oi
After the conclusion of ...	A, Mi
Immediately after the conclusion of ...	Mi
<X time> after the conclusion of ...	A
Ends ...	F, Mi, Oi
Occurs during ... / Occurs as part of ...	D, F, S
Occurs concurrently with ...	E

Figure 3-8: Mapping of SIROCCO's Time Qualifiers to Allen's Temporal Relations

As an example of how the Time Qualifiers map to Allen's temporal relations, consider "After the start of …" from Figure 3-8. This qualifier specifies that a Fact may be after (A), during (D), finishes (F), met by (Mi), or overlapped by (Oi) another Fact. The "After the start of …" qualifier and its relation to the Allen temporal relations can be visualized by inspecting the chart of the Allen relations shown in Figure 3-9. Notice that A, D, F, Mi, and Oi all have starting times that are later than the base time interval, depicted with a "y" at the top of the figure.

On the web site, the Time Qualifiers are not described in terms of Allen's relations, but, instead, in terms that are more relevant to the transcription task. In particular, each qualifier has associated information that is intended to guide the case enterer's choice – for example its intended use, what one needs to know to apply it, and links to other possible qualifiers and to case examples.

[15] "Pre-existing fact" defines a global time relation that starts before all other Facts in the chronology, except other Facts that are also pre-existing. All of the other qualifiers explicitly designate a Fact or Facts (depicted by ellipses (…) in the table) for which the specified time relation holds.

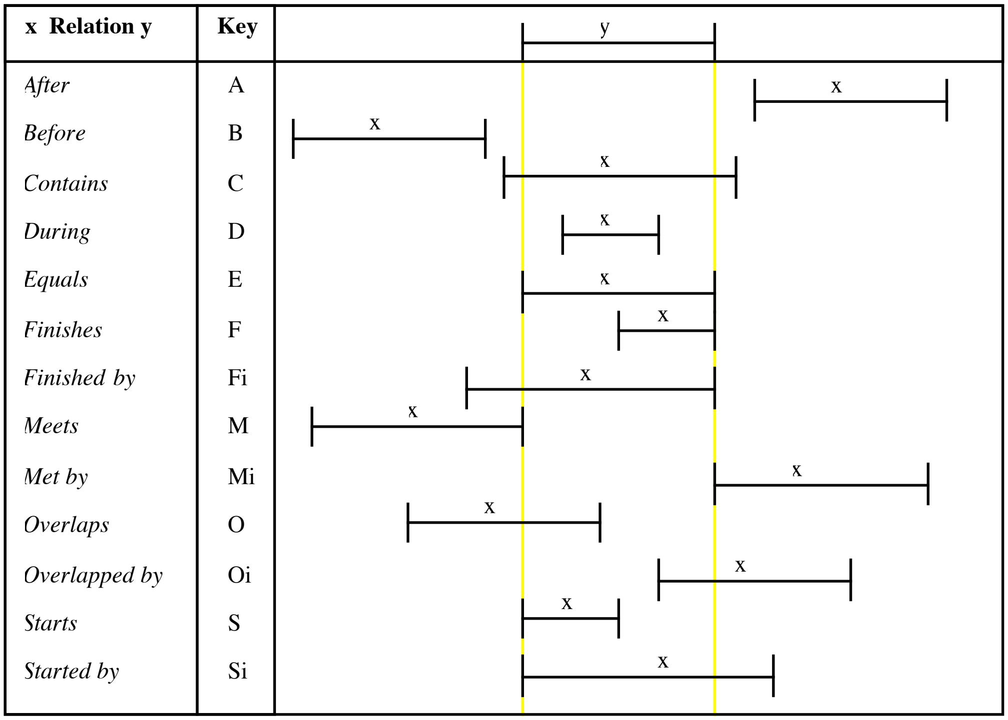


Figure 3-9: Allen’s Temporal Relations [Koomen, 1989, p. 3]

Each Fact in the chronology must have at least one Time Qualifier but may have more than one. In the Fact Chronology of Case 90-5-1, Figure 3-6, perhaps the most critical temporal relationship is that between Facts 11 and 12, in which Engineer A is told by the Attorney that he must keep the potential safety hazard confidential (in Fact 11) and then he does not inform anyone of that critical information (in Fact 12). This sequence of events is indicated by the “After the conclusion of 11” Time Qualifier assigned to Fact 12. Although this sequence did not prove compelling to the board in their conclusion – they decided in favor of the engineer’s obligation to public safety over confidentiality – it is certainly arguable that this event sequence was instrumental in prompting the board to consider the confidentiality obligation.

The goal is for the case enterer to represent the important events in the case as accurately as possible, given the somewhat limited set of Fact Primitives, Actors, and Objects. In some instances, a case enterer may not be able to be completely accurate, but he or she is asked to do his or her best. Throughout the development of SIROCCO and the transcription of the foundational cases, the ETL vocabulary was gradually supplemented. For instance, the case enterers identified important missing Fact Primitives and Actor or Object Types. However, after the transcription of the 184 foundational cases, the size of the vocabulary appeared to have leveled off, at least for the types of engineering ethics cases found in the Selected Topics group.

During the period in which the trial cases were transcribed, no changes were made to the ETL vocabulary.

The Fact Chronology of Figure 3-6 is only one interpretation of the facts of Case 90-5-1. For a variety of reasons, different case enterers may produce different interpretations. Despite the step-wise instructions of the Participant's Guide, a case enterer must make a number of judgments including:

1. *Deciding whether certain facts should be represented at all*. It is not always necessary, or even possible, to represent every sentence of a fact situation, assigning Fact Primitives to every piece of information. Some of the facts in the natural language text of the case may simply not be relevant to the ethical dilemma.

2. *Recording facts that are implied but not explicitly stated in the text*. Often, critical facts are not explicitly stated but are nonetheless relevant and should be recorded. For instance, the text of Case 90-5-1, shown in Figure 3-2, does not state that Engineer A "knows that a government authority should be informed about the safety hazard." Yet it appears reasonable, even important, to specify this fact, as is done in step 9 of the Fact Chronology of Figure 3-6.

3. *Deciding whether to reference other facts within a fact*. The arguments of a Fact Primitive may be filled by other Fact Phrases as in Facts 4, 6, 8, 9, 10, 11, and 12 in Figure 3-6. This supports flexibility in expression, alternative formulations, and explicit linking of facts to one another.

4. *Deciding on a particular Fact Primitive when multiple Fact Primitives may be possible*. ETL provides multiple ways to express the same or similar concepts. Similar primitives are clustered together in the Action/Event Hierarchy (to be discussed) and SIROCCO uses this clustering information to assess similarities between similar primitives.

An important issue addressed in this dissertation is the empirical question of whether ETL is expressive enough, and SIROCCO's similarity metric flexible enough, to support the program in doing an acceptable job of case retrieval despite the limitations and need for the subjective judgments discussed above. The empirical evidence, discussed in Chapter 4, indicates that the language supports SIROCCO's retrieval algorithm at least well enough to out perform several competing retrieval methods.

The Extended Ethics Transcription Language (EETL)

Representing a case to input as a target problem for SIROCCO's analysis involves the application of the Ethics Transcription Language and the steps described above. Submitting a case to the case base, on the other hand, requires all of the above, plus a representation of the board's analysis of the case using the Extended Ethics Transcription Language. The additional representation required to complete a case for inclusion in SIROCCO's case base is discussed in this section.

In summary, the EETL analysis representation provides a *conclusion* (e.g., Ethical, Unethical, or Unknown) and a *justification* or *argument* for that conclusion. For instance, a representation of the board's analysis for example Case 90-5-1 is depicted in Figure 3-10. As can be seen at the top of the figure, the board concluded that Engineer A, the Questioned Actor in this case, acted unethically by not informing anyone of the public safety problems he discovered while inspecting the apartment building. Engineer A's questioned, and ultimately unethical, action is represented by Fact 12 in the Fact Chronology, and a link to that Fact is provided at the top of the analysis representation.

The case enterer provides a summary of the board's argument or justification in support of its conclusion. Three tables are used for this purpose. The first table is filled with the codes and cases that were cited by the board and that, in the circumstances of the current case, support the board's conclusion. The second table is populated with the cited codes and cases that conflict with the board's conclusion in these circumstances. The citations in this table run counter to the board's conclusion but were nonetheless overridden in the final decision. The third table contains codes and cases that were cited for informational or background purposes. These elements may have relevance to the case, but they do not provide strong evidence either for or against the board's conclusion.

For instance, consider the categorization of the board's code and case citations in the analysis of Case 90-5-1 in Figure 3-10. (In reviewing the analysis representation, it is suggested that the reader refer back to Figure 2-2, an abbreviated version of the NSPE BER's actual analysis of this case.) Code II.1.a. and Case 84-5-1[16] are listed as citations that support the conclusion that Engineer A's action was unethical. Code II.1.a. states:

> Engineers shall at all times recognize that their primary obligation is to protect the safety, health, property and welfare of the public. If their professional judgment is overruled under

[16] In the tables, a case "name" is read as the value in the Case column appended by the value in the Q# column. For instance, the case in the first table is read as "84-5-1," because the case reference is "84-5" (from the Case column) and the question number is "1" (from the Q# column).

circumstances where the safety, health, property or welfare of the public are endangered, they shall notify their employer or client and such other authority as may be appropriate.

Questioned Fact(s): Fact 12

Questioned Actor or Actors: Engineer A

The Board's Conclusion: Unethical

The board cited the following evidence in support of their conclusion:

Code	Code Status	How Cited	Grouped With	Over rides	Why Relevant?	Why Violated, Not Violated, Changed, or Not Applicable?
II.1.a	Violated	Explicitly discussed	None	II.1.c	^ Engineer's judgment is overruled in a particular professional circumstance. [11] Overruling the Engineer's judgment may lead to the endangerment of the safety, health, property or welfare of the public [8, 9] ^	^ In the given situation, Engineer does not hold paramount the safety, health, property, and welfare of the public [12] ^
Case	**Citation Type**	**How Cited**	**Grouped with**	**Q #**	**Why Relevant?**	**Why Distinguished or Analogous?**
84-5	Analogous Precedent	Explicitly discussed	None	1	^ % Engineer's judgment is overruled in a particular professional circumstance. % [11] % Engineer complies with the client's judgment % [12] ^	^ % Engineer's complicity could result in a potentially dangerous situation % [1, 8, 9] ^

The board cited the following evidence that conflicts with their conclusion:

Code	Code Status	How Cited	Grouped With	Over rides	Why Relevant?	Why Violated, Not Violated, Changed, or Not Applicable?
II.1.c	Not violated	Explicitly discussed	None	None	^ Engineer has a client [6] Engineer obtains confidential facts, data, or information through work for the client. [7, 8]^	^ Engineer does not reveal confidential facts, data, or information to unauthorized parties [12] ^

The board cited the following background information that neither directly supports nor directly conflicts with their conclusion:

Case	Citation Type	How Cited	Grouped with	Q #	Why Relevant?	Why Distinguished or Analogous?
82-2	Relevant, But Not Controlling	Explicitly discussed	None	1	^ % Engineer has a client % [6] % Engineer obtains confidential facts, data, or information through work for the client. % [7, 8] % However, in the present case there is no issue of "conflict of interest" as there was in 82-2 % [6, Inference based on facts] ^	^ NA ^

Figure 3-10: The Analysis Representation of Case 90-5-1

The board interprets this code as requiring Engineer A to report the safety violations, particularly considering that the engineer's professional judgment has been overruled (i.e., by the attorney). Furthermore, the board argues that Code II.1.a. takes precedence over the confidentiality code, II.1.c. in these circumstances; this is represented in Figure 3-10 by the inclusion of II.1.c. in the row headed by II.1.a. and beneath the column titled "Overrides." Case 84-5-1 is cited as an analogous precedent. In this previously decided case, an engineer's judgment was also overruled in circumstances in which public safety might have been at risk. Similarly, in the prior case the engineer also did not attempt to dispute the overruling of his judgment by, for instance, contacting appropriate "authorities." The board came to the conclusion that the engineer's action in Case 84-5-1 was unethical, citing Code II.1.a., and the board held the circumstances and conclusion of the previous case as an exemplar for the conclusion in Case 90-5-1.

On the other hand, Code II.1.c., the confidentiality code, states:

> Engineers shall not reveal facts, data or information obtained in a professional capacity without the prior consent of the client or employer except as authorized or required by law or this Code.

As has been discussed, this code argues against the board's conclusion. That is, to report the safety violations he has discovered, the engineer must, at the same time, violate his client's confidences. The board acknowledges the tension between the two codes, but they also clearly indicate that Code II.1.a. predominates in this situation. The conflict of Code II.1.c. with the conclusion is represented by the presence of that code in the second table of Figure 3-10.

The final table contains those citations that provide relevant and useful background information, but do not directly impact the conclusion reached by the board. For instance, consider Case 82-2-1, found in the third table of Figure 3-10. This past decided case seemed to be cited by the board to introduce and discuss the importance of confidentiality. However, the particular circumstances of the prior case are significantly different; in particular, there is no issue of public safety, so this past case was not interpreted, by the case enterer, as providing precedential support either for or against the board's decision in Case 90-5-1.

Each row of the tables provides additional detail about each citation. Most important are the last two columns of each row. The "Why Relevant" column is intended for sentences that explain why the code or case would be considered relevant to the current case. The last column, with different titles for code citations versus case citations, is intended for sentences that provide evidence that the citation is violated, not violated, changed, or not applicable (for codes), or distinguished or analogous (for cases).

In these last two columns, the *Code* and *Case Instantiations* are effectively defined[17]. The case enterer creates the Instantiations by linking the Facts of the Fact Chronology to declarative sentences that represent the cited codes and cases. For a code citation, the declarative sentences are rephrasings of the original text of the code. The declarative code sentences are predefined and provided on the web site, to be copied by the case enterer as he or she transcribes a case. For a case citation, the sentences are an interpretation of the key issues of the past case. These sentences are not predefined; they are the personal interpretation of the case enterer and are constructed from scratch.

Consider the first row of the first table in Figure 3-10, the row representing the citation of Code II.1.a. This code is *relevant* for two reasons. First, the "Engineer's judgment is overruled in a particular professional circumstance," and this general statement is best supported by Fact 11 of the Fact Chronology of Case 90-5-1 (Figure 3-6), the fact in which the attorney instructs Engineer A not to reveal the public safety problems he has discovered. Second, "Overruling the Engineer's judgment may lead to the endangerment of the safety, health, property or welfare of the public" is best represented by the eighth and ninth Facts of the Fact Chronology. These two facts state that Engineer A discovers the problems that may lead to a potential safety hazard and that the engineer knows that a government authority should be alerted of such facts. Finally, the code is *violated* under these circumstances because "In the given situation, Engineer does not hold paramount the safety, health, property, and welfare of the public." This general statement is supported in specific terms in the Fact Chronology by the last fact, in which Engineer A withholds his knowledge of the public safety problems. Altogether, the Facts 8, 9, 11, and 12 form a *Code Instantiation* operationalization for Code II.1.a. in Case 90-5-1.

By filling in tables like those in Figure 3-10, the case enterer records the particular Facts in the problem that seem to best answer the questions: Why is the code provision relevant? Why was it violated? Why was it not violated? Why is the past case relevant? Why is the past case an analogous precedent? Why is it a distinguishing precedent? Answering these questions by specifying the relevant Facts defines the *Code* and *Case Instantiations* in that problem. The case enterers are instructed to do the best they can at identifying the Facts the review board regards as important for explaining the relevance and disposition of the cited codes and cases.

In many instances, the Facts of the Fact Chronology do not, by themselves, provide adequate support for the sentences of the code and case citations. For this reason, there are several special-purpose *citation support elements* that can also be used: "Hypo," "Unstated Assumption," and

---

[17] Note, however, that the columns are not explicitly called " *Instantiations*" in Figure 3-10. This is because *Instantiations* are internal SIROCCO constructs. They are mapped from the user-friendly table representations, but are

"Inference Based on Facts." A "Hypo" is used when the board explicitly states that they are making an assumption in order to claim that a particular code or case is relevant to, violated, did not violate or is analogous to the target case. Typical phrases that introduce a "Hypo" in the analysis texts are: "Assuming that…", "Provided that …" and "If we assume that …" When a "Hypo" is used, the corresponding quotation from the actual text of the analysis is copied to the table, along with the "Hypo" indicator. An "Unstated Assumption" is applied by the case enterer when *nothing* in the analysis text appears to directly support the cited code or case; yet, by virtue of the board having cited this code or case, it is clear the board must be making a supporting assumption. In other words, an "Unstated Assumption" is like a "Hypo" except that the board does not explicitly state the assumption in the analysis text. Finally, the support element "Inference Based on Facts" is used when the board appears to be inferring new facts based on the stated facts. That is, the board asserts some unstated fact that was apparently derived as a logical conclusion of the stated facts.

The other columns of the analysis representation provide various additional information that describes the code and case citations. For instance, the column headed by "Code Status" indicates whether a cited code is violated, not violated, not applicable, changed, or unknown in the present circumstances. The same column for cases, titled by "Citation Type," indicates whether a cited case is an analogous precedent, distinguishing precedent, or a relevant (but not controlling) citation. The column headed by "Grouped With" indicates, for both codes and cases, those citations that are cited in tandem because of similarity or their combined support for a particular position (i.e., this is effectively how the *Group Codes* and *Group Cases* operationalization techniques are represented).

All of the possible values for all of the columns in the tables are provided in Appendix A. The Participant's Guide section of Appendix B reproduces the instructions that were given to the case enterers for filling in these tables.

The tables in Figure 3-10 provide a form the case enterer fills out to annotate those aspects of the review board's analysis of a problem that relate most closely to the operationalization techniques. This information will be of use in helping SIROCCO retrieve relevant cases in the future.

---

not labelled *Instantiations*, as such, for the case enterer.

The Action/Event Hierarchy

Although Fact Primitives are specified at the most detailed level in the Fact Chronologies, SIROCCO is able to reason with the primitives at a more abstract level using a structure known as the *Action/Event Hierarchy*. The Action/Event Hierarchy, developed through an analysis of the NSPE corpus of cases, is a characterization and abstraction of the most-important actions and events that typically occur in engineering ethics scenarios. As will be discussed, the Action/Event Hierarchy is an important component of SIROCCO's retrieval algorithm. Cases may potentially be retrieved and matched based on similarity at higher levels of the hierarchy; in fact, both stages of SIROCCO's retrieval algorithm utilize the Action/Event Hierarchy to perform some form of inexact matching. This abstraction hierarchy helps with the problem, discussed earlier, that different case enterers may have different interpretations of the facts, as different Fact Primitives can be matched at higher levels of the abstraction hierarchy.

A portion of SIROCCO's Action/Event Hierarchy is shown in Figure 3-11. Fact Primitives are displayed in italics as the leaves of the hierarchy. Abstract categories are the inner nodes of the hierarchy. The depth of the hierarchy ranges from 2 to 5 levels and there are a total of 33 first level abstraction categories (i.e., those categories directly beneath the Fact-Primitive-Root). Note also that the hierarchy is a network and not a tree; for instance, "reviews-and-analyzes" is classified in two abstraction categories. The Action/Event Hierarchy is shown in its entirety in Appendix A.

The Fact Primitive "inspects," found within the abstraction category "Perform-Engineering-Analysis-or-Testing-Work" in Figure 3-11, is used in the seventh fact of Case 90-5-1's Fact Chronology (see Figure 3-6). When SIROCCO's analysis of the 90-5-1 is traced later in this chapter, it will be shown how the Action/Event Hierarchy helps SIROCCO improve the match between the example case and a case that uses a primitive from the same abstraction category.

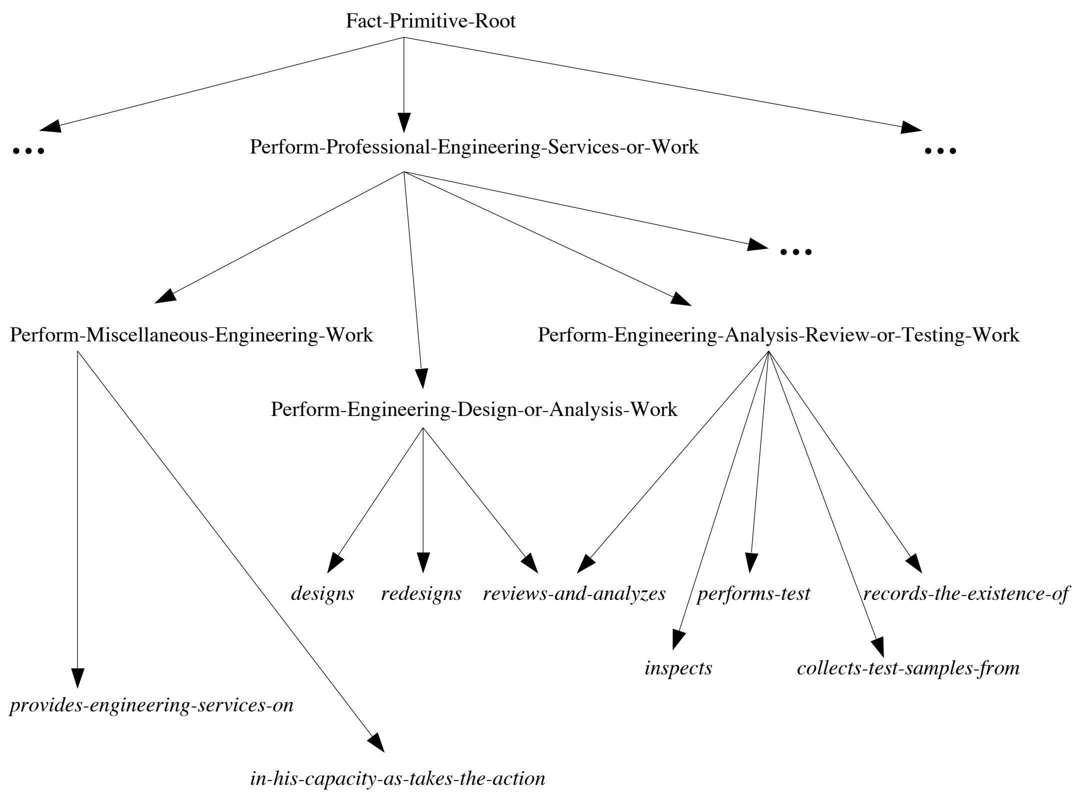


Figure 3-11: A Portion of SIROCCO's Action/Event Hierarchy

Code Representation and the Code Hierarchy

Other than the natural language that describes a code, each individual code in SIROCCO is represented in a purely extensional way. In particular, each code is represented by the set of *Code Instantiations* that links that code to specific Facts of the cases in which it is cited. There is no antecedent or consequent to a code, as in standard production rule representation.

The provisions of the NSPE code of ethics are cast in an abstraction hierarchy, the Code Hierarchy, which is an adaptation of the NSPE's subject reference list [1996, p. 8]. Essentially, the Code Hierarchy groups related codes together according to similarity of the issues they address. The Code Hierarchy is used by SIROCCO to assess the citation overlap of possible case citations to the input cases, and it also provides important similarity information used to quantify SIROCCO's accuracy and to compare SIROCCO to the other methods. In essence, the Code Hierarchy provides a means for assessing inexact matches between a pair of codes. For instance, two different codes found in the same abstract code category are considered an inexact match for purposes of calculating the F-Measure used in the experiments (see Section 4.1.3).

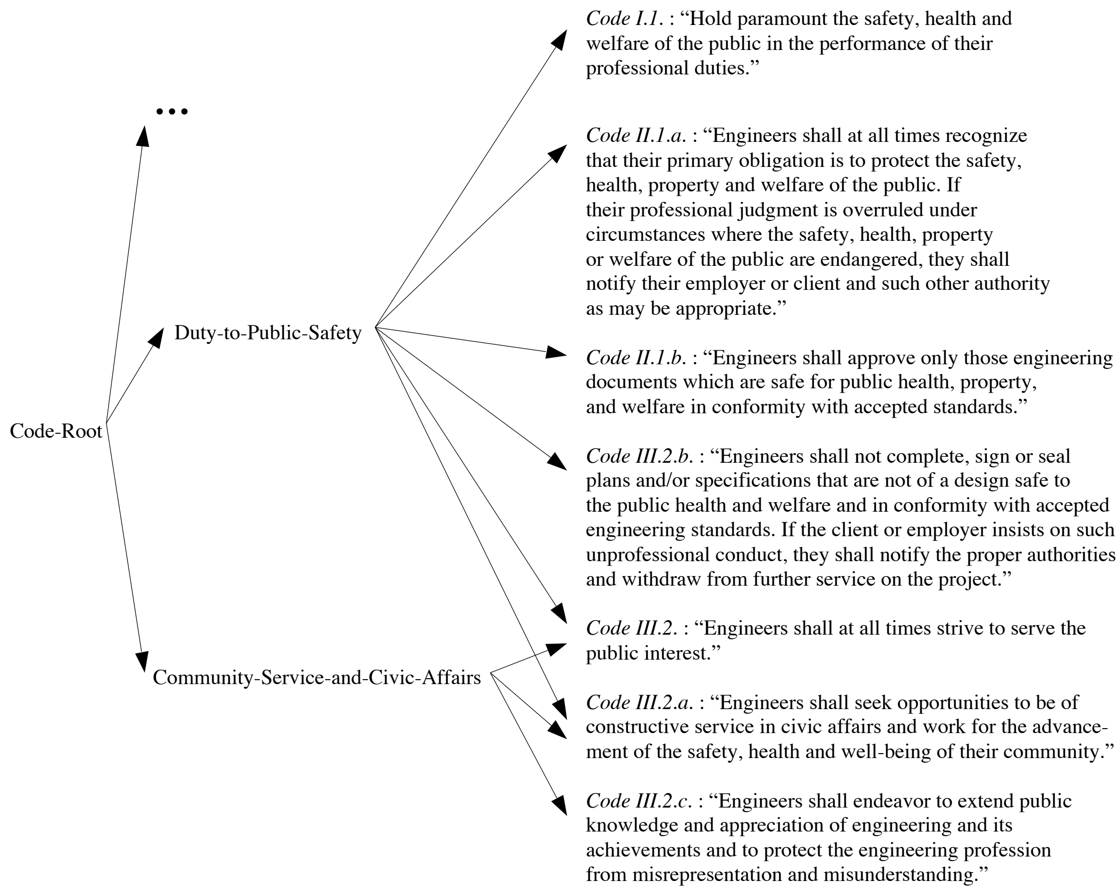


Figure 3-12: A Portion of SIROCCO's Code Hierarchy

A portion of the Code Hierarchy is shown in Figure 3-12. Codes are the leaves of the hierarchy with code names italicized. Abstract categories are the inner nodes of the hierarchy. There are a total of 75 codes in the hierarchy, 31 abstraction categories, and the maximum depth of the hierarchy is 4. As with the Action/Event Hierarchy, the Code Hierarchy is a network and not a tree. Note, for instance, that Codes III.2. and III.2.a. are classified as both "Duty-to-Public-Safety" and "Community-Service-and-Civic-Affairs" types. The Code Hierarchy is shown in its entirety in Appendix A.

Because the NSPE BER cases were decided over such a long period of time, both the content and naming of the codes has changed from 1958 to the present. At two separate times during the 40-year existence of the code, in 1964 and again in 1981, the review board reorganized and completely renamed all of the codes. However, the vast majority of the content of the original codes remains, in some form, in the present code of ethics. Most of the changes made by the review board over the years involved (a) adding new codes and (b) modifying or extending the language of existing codes. There are very few instances in which codes were deleted altogether from the code of ethics.

The inconstancy of the codes is handled in the following way in the SIROCCO model. The code of ethics as of July, 1993[18] and its present naming convention are used as the basis for all cases discussed in this dissertation. For cases decided prior to 1981, when the present convention was established, the transcription translation program translates each old citation to the July, 1993 version of that citation. That is, the case enterers provide the code citations that were applied at the time a case was decided, and then the translation process maps those citations to SIROCCO's current code set. Although this normalization process effectively allows SIROCCO to ignore the differences between old and new codes, the changes to the codes are handled, at least to a minimal extent, by SIROCCO's application of *Rewrite a Code* operationalizations. In particular, when the board discusses code changes in their analyses, and those changes are recorded by the case enterer, SIROCCO has the capability to cite the code changes in the context of new cases. On the other hand, small changes to the code language are, for the most part, not highly critical to SIROCCO's retrieval objective. Even in instances of changes to the codes, it is important that SIROCCO point the user to past cases and codes that *might* be relevant to a new situation.

In instances in which the old code citations do not map to new citations, SIROCCO carries the old citation as part of the representation of a case. Outdated citations may be presented to the user as suggested codes, but they are not used in estimating the value of a case citation or other operationalizations. In addition, the unmapped old code citations are ignored in assessing SIROCCO's (and the other models') accuracy.

Components of the Knowledge Representation Not Used by SIROCCO's Algorithm

The Extended Ethics Transcription Language was designed, developed, and deployed for case acquisition prior to the completion of the SIROCCO algorithm. Although it was originally believed that all of the knowledge representation of EETL would provide support for retrieval and analysis in SIROCCO, a number of the structures and components discussed in this chapter were ultimately not incorporated into the program's reasoning process. In this section, I briefly mention the unused structures and discuss why they are not used.

Perhaps the most significant unused aspect of the representation are the Actor and Object types. Originally, it was believed that these types would provide contextual information in assessing the similarity between Facts of different cases. For instance, one case involving an

[18] Some minor language changes and a couple new codes were added in 1996, but these are not included in the present version of SIROCCO.

"Engineer" and an "Engineering Manager" and another involving an "Engineer" and an "Engineering Firm" might be considered similar based on similarity between actors and roles (i.e., an employee-employer relationship). Although there are a few notable exceptions, the Actor and Object information did not appear, in general, to provide the intended benefit. Rather, the Fact Primitives and the mapped *consistency* of Actors and Objects across different pairs of matching facts seemed to provide most of the useful similarity information. Perhaps some exceptions could have been implemented. For instance, a set of codes stipulates how an engineer should behave when employed concurrently in both public and private capacities. Cases in which these codes apply tend to have a standard set of Actor and Object types. Thus, for instance, one can imagine some type of similarity assessment in which a new case with an "Engineer" who is employed by both a "Governmental Body" and a "Firm" might be rated a close match to the cases which cite the public/private sector codes, irrespective of the other Fact Primitives involved. For the present version of SIROCCO, however, these types of special situations were considered *ad hoc* and were not included in the algorithm.

Internal Fact-Phrases, i.e., those fact structures that are nested within Facts, are treated as atomic, rather than complex, structures by SIROCCO. That is, an internal Fact-Phrase may map to another Fact-Phrase, an Object, or an Actor of another Fact, but the structure within the Fact-Phrase is not evaluated or used as part of the similarity assessment. Fact-Phrases are mapped positionally to other Fact-Phrases, Objects, or Actors that are contained within a matching top-level Fact. Originally, it was thought that Fact-Phrases, such as the internal phrase "Apartment Building <fails standards and may be hazardous to safety>" used in Fact 8 of Figure 3-6, could be used to more precisely match Facts of other cases. Although internal structure clearly provides more precise information about the actions and events of a case, it is computationally expensive to recursively match these types of structures (which, theoretically, may have many nested levels). In addition, it was unclear that the payoff in increased accuracy would be worth the cost in run time of a recursive matching scheme.

Fact Modifiers are provided to represent partial participation in actions and events. The possible Fact Modifiers are "partially," "substantially," "limited," and "extensive." The original idea was that modified Fact Primitives would yield weaker matches to unmodified Fact Primitives. Ultimately, however, it was unclear just how this information could factor into SIROCCO's similarity assessment. In addition, the case enterers applied modifiers very sparingly. Thus, the version of SIROCCO's algorithm discussed in the dissertation does not use them.

A number of the elements of the EETL analysis representation are also not used by SIROCCO. For instance, each code and case citation can be assigned an *importance level* (i.e., "More Importance," "Less Importance"), and each code and case can be defined as explicitly discussed or referenced only (see the column headed "How Cited" in Figure 3-10). Both of these elements were intended to provide a way of placing more or less emphasis on a particular code or case citation in an argument or justification made by the board. Again, it was unclear how this information would be included in SIROCCO's similarity assessment, and thus it is unused in the current program.

Finally, the citation support elements "Inference Based on Facts" and "Unstated Assumption" are not used by the SIROCCO algorithm. These support elements tend to undermine the *Code* and *Case Instantiations* created by linking Facts to the citations. This is because these elements are generally applied when none of the Facts from the Fact Chronology provide direct support for sentences of the citations. However, because of the complex and unknown reasoning underlying these two citation support elements, as well as the fact that the case enterers used these elements very sparingly, they were ultimately ignored for the purposes of developing SIROCCO's retrieval and analysis algorithm.

3.1.3. Stage 1: Surface Retrieval

In this and the following sections, the individual phases of SIROCCO are explained in detail and the example target case, 90-5-1, and source cases, 76-4-1, 99-1-1, and 84-5-1, are used to illustrate the behavior of the computational model and the output it produces.

The retrieval phase of the program is composed of two stages, *Stage 1: Surface Retrieval* and *Stage 2: Structural Mapping.* In Stage 1, SIROCCO retrieves the source cases that appear, at least superficially, to be the most promising precedent cases. Stage 1 is superficial in the sense that it doesn't account for case structure – that is, the relationship between and within Facts – but it does account for abstract Fact Primitive matches and matches to important source case Facts. SIROCCO's Stage 1 algorithm is summarized in Figure 3-13. The major steps of the algorithm (i.e., steps 1, 2, 3, 4, and 5) are executed in sequence. All of these steps, as well as the important sequential substeps of step 2, are named in the figure to allow ease of reference. The input parameters are in italics throughout.

SIROCCO Stage 1: Surface Retrieval

```
INPUT:    Target-Case, N, CV-Weight-List, QF-Weight-List, CF-Weight-List
OUTPUT:   The top N Source Cases, sorted by the Weighted Dot Product Calculation

1. Create Content Vectors to Represent the Target Case
   For each abstraction level (FACT-PRIMITIVE, FACT-GROUP, SIBLING-GROUP, FACT-ROOT)
      Generate a content vector for Target-Case at the current level

2. Calculate Dot Product and Apply Abstraction Level Weighting to all Source Cases
   For each abstraction level (FACT-PRIMITIVE, FACT-GROUP, SIBLING-GROUP, FACT-ROOT)
      2(a). Retrieve Source Cases and Calculate Dot Products for each at the Current Level
            For each Element in the content vector of the Target-Case at the current level
               For each Source-Case containing Element (key the content vector hash table)
                  Current-Level-Dot-Product [Source-Case]  =+
                     Source-Fact-Count * Target-Fact-Count
      2(b). Adjust all Dot Products for Possible Multiple Parents
      2(c). Normalize all Dot Products for Current Level
            For each retrieved Source-Case
               Current-Level-Dot-Product [Source-Case]  =/  Total Count of Content Vector Elements
      2(d). Tally Combined Dot Products by using Weighted Sum
            For each retrieved Source-Case
               Dot-Product [Source-Case]  =+
                  CV-Weight-List [CV-Level] *
                     (Current-Level-Dot-Product [Source-Case]  /  Max-CV-Dot-Product)

3. Apply Questioned Fact Weighting to all Source Case Dot Products
   For each abstraction level (FACT-PRIMITIVE, FACT-GROUP, SIBLING-GROUP, FACT-ROOT)
      For each Questioned-Fact in the Target-Case
         For each Source-Case in which Questioned-Fact is also a questioned fact in Source-Case
                        & no questioned fact update yet
            Dot-Product [Source-Case]  =+  Max-Dot-Product  *  QF-Weight-List [CV-Level]

4. Apply Critical Fact Weighting to all Source Case Dot Products
   For each abstraction level (FACT-PRIMITIVE, FACT-GROUP, SIBLING-GROUP, FACT-ROOT)
      For each Fact in the Target-Case
         For each Source-Case in which Fact is a critical fact & no critical fact update yet
            Dot-Product [Source-Case]  =+  Max-Dot-Product  *  CF-Weight-List [CV-Level]

5. Return the Top N Source Cases
   Sort all Source Cases by descending dot product (i.e., sort Dot-Product [*])
   Return the top N Source Cases
```

Figure 3-13: SIROCCO's Stage 1 Algorithm

Stage 1 is superficial in the sense that it focuses solely on fact matching between the target case and the source cases and ignores the structural characteristics of the graphs that represent each case. Essentially, Stage 1 compares each new target problem to all of the source cases that share at least some abstract factual similarity with the target. Comparison is performed at each of four predefined levels of abstraction, shown in Figure 3-14, and the results are combined. The value of such an approach is that both exact and inexact similarity are factored into the overall similarity assessment. In addition, by predefining the abstraction levels and pre-storing each source case's information at each of those levels, the similarity computation is relatively fast.

Fact Primitive:	Exact matches between Fact Primitives.
Fact Group:	Matches between Fact Primitives that share the same parent abstraction category, i.e., matches one level up the Action/Event Hierarchy.
Sibling Group:	Matches between Fact Primitives that share the same abstraction category two levels up the Action/Event Hierarchy.
Root Group:	Matches between Fact Primitives at the "root group" level, i.e., one level below the root of the Action/Event Hierarchy.

Figure 3-14: SIROCCO's Predefined Abstraction Levels for Matching Fact Primitives

SIROCCO enlists a variety of knowledge to assist in this task. First, the program uses the knowledge found in the Action/Event Hierarchy. In particular, it performs retrieval and matching at each of the predefined abstraction levels. SIROCCO calculates match scores between the target case and all of the source cases that share at least one element at any of the predefined abstraction levels. SIROCCO stores, in advance, a specialized representation of all of the source case facts and their abstractions in hash tables, thus making the retrieval and computation relatively fast. At run time, the program uses the Fact Primitives of the target – and the abstractions of those primitives – as keys into the hash tables corresponding to each abstraction level.

The specialized knowledge structures stored in the hash tables are called *content vectors*. Each content vector summarizes the Fact Chronology of a single case by specifying the Fact Primitives, or abstractions of those primitives, found in the chronology and a count of how many times each primitive (or abstraction) appears. The concept of a content vector was first suggested by Forbus and colleagues [1994], but in SIROCCO it has been extended to represent fact abstractions, as well as the specific facts of a case.

The content vectors of all of the source cases in SIROCCO's case base are created and stored in the aforementioned hash tables prior to the retrieval and analysis of new target cases[19]. When a new case is provided to SIROCCO for analysis, the program immediately takes the ETL case format and generates four content vectors, one for each of the standard abstraction levels, to represent the new target case. This translation process is depicted in step 1 of Figure 3-13, and the resulting content vectors for the first two abstraction levels (i.e., Fact Primitive and Fact

Group) of Case 90-5-1's facts are shown in Figure 3-15. For simplicity, the content vectors for the Sibling and Root Group are omitted. Refer to Figure 3-6 to see the Fact Chronology of Case 90-5-1.

Case 90-5-1

Fact-Primitive-CV:	**Fact-Group-CV**:
(May-be-Hazardous-to-Safety 1)	(Deal-with-Potential-Dangers-or-Hazards 1)
(Owns 1)	(Own-Something 1)
(Resides-in 1)	(Specify-Location-of-Residence 1)
(Files-a-Lawsuit-or-Arbitration-Action-Against 1)	(Initiate-Legal-or-Arbitration-Proceedings 1)
(Is-Legally-Represented-by 1)	(Has-Legal-Representation 1)
(Hires-the-Services-of 1)	(Work-as-an-Employed-or-Contract-Professional-Engineer 1)
(Inspects 1)	(Perform-Engineering-Analysis-Review-or-Testing-Work 1)
(Discovers-That 1)	(Know-or-Believe-Something 2)
(Knows 1)	(Disclose-Information 2) ***
(Informs-That 2) ***	(Order-Subordinate-to-Perform-Task 1)
(Instructs-to 1)	

Figure 3-15: The Fact Primitive and Fact Group Content Vectors for Case 90-5-1

Note that almost all of the Fact Primitives in the Fact-Primitive-CV have values of 1, because each primitive appears only once, with the exception of "Informs-That," which appears twice, in steps 10 and 12 in Figure 3-6. Also note that the Questioned Fact of the target case has been denoted by three asterisks ("***").

The four content vectors representing the target case help SIROCCO to retrieve all of the source cases that share at least some fact similarity with the target. Each element of each content vector is used as a hash key to retrieve source cases sharing that same element. For instance, see Figure 3-16. On the left side of the figure are the first two elements of Case 90-5-1's Fact-Primitive-CV and the first two elements of its Fact-Group-CV. These elements serve as hash keys into the Fact-Primitive and Fact-Group hash tables, respectively, shown on the right side of the figure. At each hash table location, there is a list of the cases that contain that Fact Primitive or abstraction. Each element in this list also specifies the fact count for that Fact Primitive or fact abstraction within that case. For example, the Fact-Primitive hash table in Figure 3-16 shows that "May-be-Hazardous-to-Safety" is found once in Case 84-5-1 and once in Case 99-1-1. By using every element of the target case's content vectors as a hash key, the program is thus able to collect, and score, all of the cases that share at least some fact similarity with the target.

Notice how the multiple tables allow the program to retrieve and score abstract matches. For instance, the fact abstraction "Deal-with-Potential-Dangers-or-Hazards" is stored in the Fact-

[19] It is actually *fact situations*, not cases, that have associated content vectors, since Fact Chronologies are part of fact situations (see Figure 3-4). Multiple cases may share the same fact situations and therefore may share the same content vectors. Since this is a largely irrelevant distinction for this description, I will simplify refer to cases only.

Group-CV of Figure 3-16, because Case 90-5-1 contains the primitive "May-be-Hazardous-to-Safety" which is a Fact Primitive of this abstraction category. When this fact abstraction is hashed into the Fact-Group hash table, on the right, it retrieves not only the cases that contain an exact match to the primitive (i.e., 84-5-1 and 99-1-1, discussed above) but also two cases, 63-4-1 and Case 72-3-1, that match only at the fact abstraction level. These cases are found at this location because they apparently contain a different Fact Primitive in the "Deal-with-Potential-Dangers-or-Hazards" category.

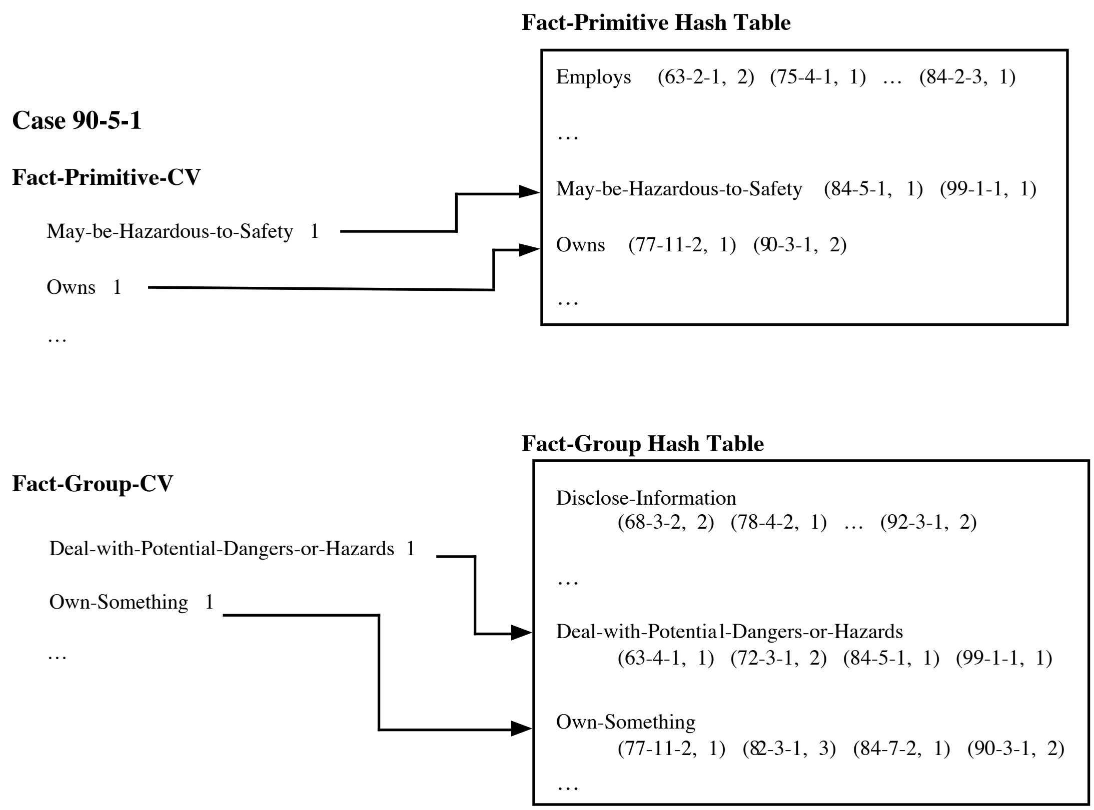


Figure 3-16: Example of Target Content Vectors Retrieving Cases Using the Hash Tables

As shown in step 2(a) of Figure 3-13, a *dot product* calculation is then applied (incrementally) to determine the degree of content vector overlap between the target and each of the retrieved source cases. The dot product is calculated as follows:

$$Dot\text{-}Product\ (SC, TC) = \sum cv_{TC} \cdot cv_{SC}$$

Where *SC* is the source case
TC is a target case
cv_{TC} is a count of element *x* in the target content vector
cv_{SC} is a count of element *x* in the source content vector
element *x* is either a Fact Primitive or a fact abstraction

The content vectors representing the first two abstraction levels for each of the example source cases are shown in Figure 3-17. All three of the example cases are retrieved in step 2(a) at each abstraction level, because in each case elements are shared between the target and the source cases. For instance, Case 76-4-1 is retrieved at the Fact Primitive level because it shares the primitives "Hires-the-Services-Of," "Discovers-That," "Instructs-to," and "Informs-That" with the target case.

Case 76-4-1

Fact-Primitive-CV:
(Hires-the-Services-Of 1)
(Reviews-and-Analyzes 1)
(Discovers-That 3)
(Informs-That 2) ***
(Terminates-the-Services-of 1)
(Instructs-to 1)
(Pays-For 1)
(Calls-a-Hearing-Regarding 1)
(Claims-That 1)

Fact-Group-CV:
(Work-as-an-Employed-or-Contract-Professional-Engineer 1)
(Perform-Engineering-Design-or-Analysis-Work 1)
(Perform-Engineering-Analysis-Review-or-Testing-Work 1)
(Know-or-Believe-Something 3)
(Disclose-Information 3) ***
(Terminate-Services-by-Client 1)
(Order-Subordinate-to-Perform-Task 1)
(Give/Receive-Remuneration 1)
(Purchase-or-Pay-for-Something 1)
(Initiate-Legal-or-Arbitration-Proceedings 1)

Case 99-1-1

Fact-Primitive-CV:
(Has-Experience-in 1)
(May-be-Hazardous-to-Safety 1)
(Hires-the-Services-of 1)
(Inspects 2)
(Discovers-That 2)
(Informs-That 2)
(Terminates-the-Services-of 1)
(Instructs-to 1)
(Pays-for 1)
(Employs 1)
(Criticizes 1) ***

Fact-Group-CV:
(Specialize-or-Provide-Experience-… 1)
(Deal-with-Potential-Dangers-or-Hazards 1)
(Work-as-an-Employed-or-Contract-Professional-Engineer 1)
(Perform-Engineering-Analysis-Review-or-Testing-Work 2)
(Know-or-Believe-Something 2)
(Disclose-Information 2)
(Terminate-Services-by-Client 1)
(Order-Subordinate-to-Perform-Task 1)
(Give/Receive-Remuneration 1)
(Purchase-or-Pay-for-Something 1)
(Criticize-or-Accuse 1) ***

Case 84-5-1

Fact-Primitive-CV:
(Hires-the-Services-Of 1)
(Is-a-Work-Segment-of-the-Engineering-Project 1)
(May-be-Hazardous-to-Safety 1)
(Knows 1)
(Asks-For 1)
(Refuses-the-Request-by-to 1)
(Provides-Engineering-Services-on 1) ***

Fact-Group-CV:
(Work-as-an-Employed-or-Contract-Professional-Engineer 1)
(Relate-Engineering-Projects-to-One-Another 1)
(Deal-With-Potential-Dangers-or-Hazards 1)
(Know-or-Believe-Something 1)
(Request-Something 1)
(Refuse-Request 1)
(Perform-Miscellaneous-Engineering-Work 1) ***

Figure 3-17: The Fact Primitive and Fact Group Content Vectors for Cases 76-4-1, 99-1-1, and 84-5-1

The Fact Primitive overlap between 90-5-1 and 76-4-1 produces a dot product of 9 (1 * 1 for each of the matching primitives "Hires-the-Services-Of" and "Instructs-to," plus 1 * 3 for "Discovers-That," plus 2 * 2 for "Informs-That"). Not surprisingly, since these two cases are conceptually quite similar, this score ranks as one of the best in the list of retrieved cases. However, the score for 99-1-1, the case that shares much event similarity with the target but ultimately deals with a different issue, is ranked even higher, at the top of the list for Fact Primitive matching and near the top for Fact Group matching. A partial listing of the top dot product scores for the first two abstraction levels after step 2(a) is[20]:

Fact Primitive Dot Products:

1.	**Case 99-1-1: Criticism of an Inexperienced Engineer**	**11**
	Case 88-6-1: Whistleblowing City Engineer	11
2.	Case 89-7-1: Duty to Report Safety Violations	10
3.	**Case 76-4-1: Public Welfare - Knowledge of Damaging Information**	**9**
…		
8.	**Case 84-5-1: Engineer's Recommendation Overruled**	**3**
…		

Fact Group Dot Products:

1.	Case 91-9-1: Misrepresentation of Education	17
	Case 77-5-1: Use of Another's Project Study	17
2.	**Case 76-4-1: Public Welfare - Knowledge of Damaging Information**	**16**
	Case 92-6-2: Public Welfare - Hazardous Waste	16
3.	**Case 99-1-1: Criticism of an Inexperienced Engineer**	**14**
…		
14.	**Case 84-5-1: Engineer's Recommendation Overruled**	**4**
…		

In step 2(b) of Figure 3-13, SIROCCO identifies and adjusts downward dot products in which fact abstractions have been "double-counted." Since the Action/Event Hierarchy is a network and not a tree, it is possible that more than one fact abstraction, and hence more than one element in a content vector, will represent the same base Fact Primitive. For instance, consider the Fact Primitive "reviews-and-analyzes" from Figure 3-11 and also shown as a content vector element of Case 76-4-1 in Figure 3-17. Any case using this primitive will have both "Perform-Engineering-Design-or-Analysis-Work" and "Perform-Engineering-Analysis-Review-or-Testing-Work" elements in its Fact Group content vector, since both are parents of "reviews-and-analyzes." Matching such a content vector to a target case containing the same two fact abstractions leads to double-counting and thus the need for a downward adjustment[21]. Adjustments are required only above the Fact Primitive level and, in our example, none of the source cases require changes.

Because cases with long Fact Chronologies tend to be favored by the dot product calculation, simply because they contain more elements that can be matched, it is necessary to provide a normalization factor. This is what SIROCCO does in step 2(c) of Figure 3-13 when it divides the dot product by the total count of the current level's source content vector, which is roughly

---

[20] Although sorted case rankings are used here for illustrative purposes, the cases are actually not sorted until the final step of Stage 1.

equivalent to the length of the Fact Chronology of that case[22]. In the example trace, this calculation slightly reorders the list, as shown below. Notice, in particular, that Case 89-7-1 moves to the top of the Fact Primitive list, jumping over Case 99-1-1, and also moves above 99-1-1 in the Fact Group list. This occurs because Case 89-7-1's Fact Chronology is shorter than Case 99-1-1's.

Fact Primitive Dot Products:

1.	Case 89-7-1: Duty to Report Safety Violations	1	(10 / 10)
2.	**Case 99-1-1: Criticism of an Inexperienced Engineer**	**.786**	**(11 / 14)**
	Case 88-6-1: Whistleblowing City Engineer	.786	(11/14)
3.	**Case 76-4-1: Public Welfare - Knowledge of Damaging Information**	**.750**	**(8/12)**
…			
13.	**Case 84-5-1: Engineer's Recommendation Overruled**	**.429**	**(3 / 7)**
…			

Fact Group Dot Products:

1.	Case 91-9-1: Duty to Report Safety Violations	1	(17/14)
2.	Case 92-6-2: Public Welfare - Hazardous Waste	1.14	(8/7)
	Case 76-4-1: Public Welfare - Knowledge of Damaging Information	**1.14**	**(8/7)**
…			
6.	**Case 99-1-1: Criticism of an Inexperienced Engineer**	**.933**	**(14 / 15)**
…			
16.	**Case 84-5-1: Engineer's Recommendation Overruled**	**.571**	**(4 / 7)**
…			

After the adjusted, normalized dot products are calculated for all of the source cases at each predefined abstraction level, a weighted sum is used to tally a combined dot product across all levels for each case (see step 2(d) of Figure 3-13). A user-specified parameter, *CV-Weight-List*, which consists of four weights corresponding to each of the predefined abstraction levels[23], is central to this calculation. To calculate the contribution of the dot product at a particular fact level, the weight for that level is multiplied by the dot product divided by the maximum dot product at that level. Dividing by the maximum dot product effectively normalizes the contribution of the dot product at that level.

Informal experimentation with SIROCCO indicated that a *CV-Weight-List* of (Fact-Primitive = 1.0; Fact-Group = 0.5; Sibling-Group = 0.25; Fact-Root = 0.125) yields reasonably accurate results. These values specify that each abstraction level wields 50% more weight in the combined dot product calculation than its successor level. The results of applying this parameter setting in

---

[21] In general, note that such an adjustment algorithm could quickly become intractable. However, because multiple parents are relatively rare in the Action/Event Hierarchy – there are less than 10 Fact Primitives with multiple parents and a number of these are rarely used in the foundational and trial cases – in practice this routine takes very little time.
[22] Actually, SIROCCO allows the user to choose a normalization factor from among the following: (1) the sum of the source and target counts, (2) the shortest of the source and target counts, (3) the target content vector count, and (4) the source content vector count. In practice, I have consistently found the last of these, the source content vector count, to be the best factor and thus it is used in the example and in all experiments reported in the dissertation.
[23] SIROCCO normalizes this list so that the relative values add up to 1.0.

the example are shown below. At this point in the algorithm, the example source case that appeared to be most similar to the target, i.e., Case 76-4-1, is, in fact, rated very high, higher than the other two example cases. Notice that Case 99-1-1, the case with many similarities but which ultimately turns on a completely different ethical issue, is also highly rated. Finally, the third example source case, 84-5-1, which was actually cited by the board, lags far behind.

Combined Dot Products:

1.	Case 89-7-1: Duty to Report Safety Violations	.973
2.	**Case 76-4-1: Public Welfare - Knowledge of Damaging Information**	**.848**
3.	Case 92-6-2: Public Welfare - Hazardous Waste	.787
4.	Case 88-6-1: Whistleblowing City Engineer	.772
5.	**Case 99-1-1: Criticism of an Inexperienced Engineer**	**.767**
…		
18.	**Case 84-5-1: Engineer's Recommendation Overruled**	**.494**
…		

Questioned Facts, another key knowledge source not yet brought to bear on the problem, will make a critical difference in this ranking. As has been discussed, Questioned Facts are important because they are essentially the focal point of each case, anchoring SIROCCO's retrieval and analysis method. All of the output produced by the program should, in some sense, be relative to the issues raised by the Questioned Facts. SIROCCO leverages this knowledge in Stage 1 by increasing the dot products of any source cases that share a Questioned Fact with the target case, at any of the four abstraction levels (see step 3 of Figure 3-13). Each source case can realize this benefit at only one level, in particular at the most-specific level in which the match occurs (see the condition "no questioned fact update yet" in step 3). The user-specified parameter *QF-Weight-List* provides a multiplier for each of the four abstraction levels. Unlike the *CV-Weight-List*, *QF-Weight-List* is not normalized; the values provided by the user are directly applied as weights. The dot product for each source case with a Questioned Fact matching the target is increased by the maximum dot product (from the combined dot product calculation of step 2(d)) multiplied by the specified weight for that level.

For the example, a *QF-Weight-List* of (Fact-Primitive = 1.0; Fact-Group = 0.5; Sibling-Group = 0.25; Fact-Root = 0.125) is applied. This means a Questioned Fact match at the most-detailed level will improve the score by an amount equal to the highest combined dot product (i.e., max-dot-product * 1.0); a match at the next level will improve the score 50% of the highest combined dot product, and so on.

At this point, we can begin to see how SIROCCO's similarity assessment appropriately evaluates the example source cases. For Case 76-4-1, the Questioned Fact computation is good news; the score for that case doubled, as its Questioned Fact ("informs-that") matches the target's

Questioned Fact exactly, i.e., at the Fact Primitive level (see the Questioned Facts for these two cases, highlighted by "***" in Figure 3-15 and Figure 3-17). However, perhaps more importantly, this calculation *devalues* Case 99-1-1, the deceptively similar case that is not directly relevant to Case 90-5-1. Because 99-1-1's Questioned Fact does not match 90-5-1's Questioned Fact at any of the predefined abstraction levels, its score does not change, meaning that it effectively falls down the ranked list. Case 84-5-1's Questioned Fact also does not match 90-5-1's, so it, too, falls back in the ranked list. It isn't until later in SIROCCO's assessment that 84-5-1is recognized as a potentially relevant citation.

Dot Products with Questioned Facts Applied:

1.	Case 89-7-1: Duty to Report Safety Violations	1.946
2.	**Case 76-4-1: Public Welfare - Knowledge of Damaging Information**	**1.821**
3.	Case 92-6-2: Public Welfare - Hazardous Waste	1.759
4.	Case 88-6-1: Whistleblowing City Engineer	1.745
5.	Case 91-9-1: Misrepresentation of Education	1.737
6.	Case 63-11-1: Recommendation of Personnel	1.649
…		
17.	**Case 99-1-1: Criticism of an Inexperienced Engineer**	**.767**
…		
27.	**Case 84-5-1: Engineer's Recommendation Overruled**	**.494**
…		

Critical Facts are the final type of knowledge used by SIROCCO in Stage 1. Critical Facts are those Facts in source cases that are part of one or more *Code* or *Case Instantiations*. In other words, these are the Facts that the board employed as part of a code or case operationalization, as interpreted by the case enterer. SIROCCO utilizes this knowledge by increasing the dot products of any source cases that have a Critical Fact that matches a Fact of the target case, at any of the four abstraction levels (step 4 of Figure 3-13). As with the Questioned Fact weighting, each case can accrue benefit only at the most specific level at which the match occurs. The user-specified *CF-Weight-List* is applied in similar fashion as was the *QF-Weight-List*, and the increase to the dot product of a matching source case is equal to the maximum dot product multiplied by the weight corresponding to the abstraction level of the match.

Because Critical Facts appeared to have a slightly less-central role than Questioned Facts, the standard *CF-Weight-List*, set to (Fact-Primitive = 0.333; Fact-Group = 0.111; Sibling-Group = 0.036; Fact-Root = o.012), reflects a lesser contribution to the dot product calculation. In the example run, the Critical Facts calculation affected the scores but not the relative position of any of the top-rated cases. The following list contains the final scores for Stage 1:

Dot Products with Critical Facts Applied:

1.	Case 89-7-1: Duty to Report Safety Violations	2.270
2.	**Case 76-4-1: Public Welfare - Knowledge of Damaging Information**	**2.145**
3.	Case 92-6-2: Public Welfare - Hazardous Waste	2.083
4.	Case 88-6-1: Whistleblowing City Engineer	2.069
5.	Case 91-9-1: Misrepresentation of Education	2.061
6.	Case 63-11-1: Recommendation of Personnel	1.973
…		
15.	**Case 99-1-1: Criticism of an Inexperienced Engineer**	**1.067**
…		
28.	**Case 84-5-1: Engineer's Recommendation Overruled**	**.494**
…		

Stage 1 concludes with SIROCCO sorting the source cases by descending dot product. The top *N* source cases are then passed to Stage 2 for processing. An *N* value of 6 appeared to provide, in general, the most-accurate results without flooding Stage 2 with excessive (and expensive!) computation, so this value is used in the example as well. Notice that this means that Stage 1 has, in fact, managed to include the most relevant source example, Case 76-4-1, while, at the same time, eliminating the far less relevant Case 99-1-1. On the other hand, Case 84-5-1, a case actually cited by the board as an analogous precedent, is also eliminated. However, we will see Case 84-5-1 later in the processing, as SIROCCO's operationalization techniques will eventually recover it.

3.1.4. Stage 2: Structural Mapping

The goal of Stage 2 is to find structural mappings from each of the top-rated source cases to the target case. The *structure* of a case is a graph that is defined by (1) the Actors, Objects, Fact-Phrases, and Fact-Primitives of individual Facts and (2) the temporal relationships between Facts represented by the Time Qualifiers and the propagation of those qualifiers. Structural mapping between two cases is not attempted across the entire case representation of the cases but, rather, it is focused on subsets of the source Facts: those represented by the *Code Instantiations* and *Case Instantiations* of the source case. This approach achieves two objectives. First, it focuses SIROCCO's reasoning on those Facts that are most relevant to a source case, and in particular to codes and cases cited in that source case, since the Instantiations are composed of the Critical and Questioned Facts of a case. Second, it leads to a significant computational benefit, as the process of graph mapping is known to be intractable for general complex graphs [Bunke and Messmer, 1993].

SIROCCO's Stage 2 algorithm is summarized in Figure 3-18. The algorithm is composed of two major functions, with Collect-and-Sort-Instantiations calling Search-For-the-Best-Structural-Mapping. As in the description of Stage 1, the input parameters to Stage 2 are displayed in italics throughout the figure. As an example of how Stage 2 operates, consider some of the representation details of one of the example source cases, 76-4-1. Figure 3-19 shows the Fact Chronology for Case 76-4-1, while Figure 3-20 depicts the analysis representation for this case.

The goal of Collect-and-Sort-Instantiations is to produce a list of Instantiations that are mapped to the target case and sorted by degree of match, according to structural characteristics. This function operates by iterating over all of the relevant Instantiations that are associated with the top *N* cases retrieved in Stage 1. For each relevant Instantiation, Collect-and-Sort-Instantiations calls Search-For-the-Best-Structural-Mapping to perform a graph mapping of that Instantiation to the target.

Relevant Instantiations are those that either support or conflict with the conclusion reached by the board. In Figure 3-20, the rows containing Codes II.1.a.[24], I.1., III.2.b., III.1., I.4., III.1.b., and III.4. appear in the first two tables because they each represent supporting or conflicting *Code Instantiations*. On the other hand, the row containing Case 67-10-1 is in the third table because it is not considered a relevant *Case Instantiation* for the purposes of graph mapping; it neither supports nor conflicts with the board's conclusion.

Search-For-the-Best-Structural-Mapping executes a variant of A* search in its attempt to map each relevant Instantiation to the target case. The goal of the search is to map each of the Facts of the source Instantiation to a corresponding Fact in the target case while maintaining a one-to-one and consistent mapping between the Actors and Objects of the source and the target. Each node in the search space (with the exception of the initial node, see below) represents (1) a proposed mapping of a pair of Facts, one from the source Instantiation and one from the target, (2) all of the Fact mappings that preceded this node (i.e., all of the successful Fact mappings from ancestor nodes), (3) a one-to-one, consistent set of Actors and Objects entailed by the Fact mappings of (1) and (2), and (4) consistent temporal relations between mapped Facts of the source and target.

[24] Because 76-4-1 was decided before the current code of ethics was established, the codes cited in its analysis representation tables are older, outdated codes. The codes shown in boldface and parentheses below the original code citation are the newer versions of these codes. Note that multiple old codes can map to the same new code (i.e., old Codes 2 and 2(a) of Figure 3-20 both map to new Code II.1.a.), and multiple new codes can map to the same old code (i.e., new Codes I.1. and II.1.a. map to old Code 2(a)). SIROCCO handles the first case by performing separate graph mappings for each of the multiple new citations and then choosing the best match score to represent that *Code Instantiation*. The second case is handled by defining multiple *Code Instantiations* for a single row of the table, each one with identical information.

SIROCCO Stage 2: Structural mapping

INPUT: *Target-Case*, *Top-Stage-1-Source-Cases, Temporal-Match-Flag*

OUTPUT: A list of mapped *Code Instantiations* and *Case Instantiations*, sorted according to ascending match scores. Also, the *Top-Stage-1-Source-Cases* are passed on to Analysis.

Collect-and-Sort-Instantiations

```
For each Source-Case in Top-Stage-1-Source-Cases
   For each Source-Case-Instantiation in Source-Case
            (i.e., For each Supporting-Code, Supporting-Case, Conflicting-Code, Conflicting-Case)
      If Critical-Facts and Questioned-Facts of Source-Case-Instantiation are unique Then
         Best-Search-Result =
            Search-For-The-Best-Structural-Mapping
               (Source-Case-Instantiation, Target-Case, Temporal-Match-Flag)
         Push Best-Search-Result onto Code-and-Case-Instantiations
      Else
         Push search result from previous mapping onto Code-and-Case-Instantiations
Sort Code-and-Case-Instantiations by ascending match score and return them.
```

Search-For-The-Best-Structural-Mapping (Source-Case-Instantiation, *Target-Case*, *Temporal-Match-Flag*)

```
Facts-In-Instantiation = Collect Questioned-Facts and Critical-Facts of Source-Case-Instantiation
Solution-Depth = Length of Facts-In-Instantiation
Search-Node-1 = New Node :Object-Mappings (Target Questioned Actor, Source Questioned Actor)
Search-Node-List = List with Search-Node-1 as sole element
Until (Search-Node-List is empty) or (Goal-Node-P (First (Search-Node-List), Solution-Depth))
   Current-Best-Search-Node = First (Search-Node-List)
   Insert New-Child-Search-Node
            (Current-Best-Search-Node, EMPTY-MAP, Solution-Depth, Temporal-Match-Flag)
      into Search-Node-List
   For each Source→Target in Source-Target-Mapping-List
      Insert New-Child-Search-Node
            (Current-Best-Search-Node, Source→Target, Solution-Depth, Temporal-Match-Flag)
      into Search-Node-List
Return First (Search-Node-List)
```

Goal-Node-P (Search-Node, Solution-Depth)

```
If (Depth-Of (Search-Node) = Solution-Depth) Then Return True Else Return False
```

New-Child-Search-Node (Parent-Node, Source→Target, Solution-Depth, *Temporal-Match-Flag*)

```
If (Source→Target preserves one-to-one mapping and
   Source→Target preserves or extends existing Actor and object mappings and
   Source→Target preserves temporal relations (if Temporal-Match-Flag is on) ) Then
   Current-Depth = Depth-Of (Parent-Node) + 1
   Local-Mismatch-Cost = Get abstraction level value of Source→Target, adjust for pos-neg if necessary.
   Return New Node
      :Fact-Primitive-Mappings All Fact Primitive mappings from Source→Target on this path
      :Object-Mappings         All Actor and object mappings from Source→Target on this path
      :Depth                   Current-Depth
      :Path-Mismatch-Cost      Path-Mismatch-Cost-of (Parent-Node) + Local-Mismatch-Cost
      :Match-Score             (Path-Mismatch-Cost / Current-Depth) +         f (n) = g (n) + h' (n)
                               (Path-Mismatch-Cost / Solution-Depth)
```

Figure 3-18: SIROCCO's Stage 2 Algorithm

1.	XYZ Corporation **<hires the services of>** Engineer Doe **<for>** (Engineer Doe **<reviews and analyzes>** Discharge).	Pre-existing fact
2.	Engineer Doe **<reviews and analyzes>** Discharge.	After the start of 1
3.	Engineer Doe **<discovers that>** (Discharge **<fails standards and may be hazardous to safety>**).	After the start of 2
4.	Engineer Doe **<informs>** XYZ Corporation **<that>** (Discharge **<fails standards and may be hazardous to safety>**).	Immediately after the conclusion of 3
5.	XYZ Corporation **<terminates the services of>** Engineer Doe.	After the conclusion of 4, Ends 1
6.	XYZ Corporation **<instructs>** Engineer Doe **<to>** (Engineer Doe **<does not write paper/article>** (Discharge **<fails standards and may be hazardous to safety>**)).	Occurs concurrently with 5
7.	XYZ Corporation **<pays>** Engineer Doe <for> (Engineer Doe **<reviews and analyzes>** Discharge).	Occurs concurrently with 5
8.	Control Authority **<calls a hearing regarding>** Discharge.	After the conclusion of 5
9.	XYZ Corporation **<claims that>** (Discharge **<does not fail standards and is not hazardous to safety>**).	After the start of 8
10.	Engineer Doe **<discovers that>** (Control Authority **<calls a hearing regarding>** Discharge) & (XYZ Corporation **<claims that>** (Discharge **<does not fail standards and is not hazardous to safety>**)).	After the conclusion of 9
11.	Engineer Doe **<does not inform>** Control Authority **<that>** (Discharge **<fails standards and may be hazardous to safety>**). *[Questioned fact]*	After the start of 10

Figure 3-19: The Fact Chronology of Case 76-4-1

The initial node of the search space is a mapping of the questioned Actor of the source to the questioned Actor of the target. New nodes are generated from an existing node by selecting an unmapped Fact from the source and mapping it to each of the unmapped Facts in the target in which the corresponding Fact Primitives match either exactly or abstractly. A new node is generated for each mapping between the source Fact and an unmapped target Fact (e.g., if F is an unmatched source Fact that has Fact Primitive mappings to target Facts F1 and F2, two new nodes are generated, one corresponding to F->F1 and one corresponding to F->F2). In order for each new node to be valid, the Actors and Objects between the source and target must map in a one-to-one, consistent manner with prior mappings and all temporal relations must be consistent. An "empty" node is also generated at each level to represent a failure of mapping this particular Fact but to allow mapping on this path for subsequent Facts. The goal node is reached when the current depth equals a pre-defined solution depth (equal to the number of Facts to match in the source) and either the current node has the lowest mismatch score of all open nodes, as defined by the A* cost function, or the list of nodes is empty.

Questioned Fact(s): Fact 11

Questioned Actor or Actors: Engineer Doe

The Board's Conclusion: Unethical

The board cited the following evidence in support of their conclusion:

Code	Code Status	How Cited	Grouped With	Over rides	Why Relevant?	Why Violated, Not Violated, Changed, or Not Applicable?
2 **(II.1.a.)**	Violated	Explicitly discussed	2(a)	1,7	^ Engineer's judgment is overruled in a particular professional circumstance. [6] Overruling the Engineer's judgment may lead to the endangerment of the safety, health, property or welfare of the public [3, 9] ^	^ Engineer does not notify the proper authority [11] ^
2(a) **(I.1., II.1.a.)**	Violated	Explicitly discussed	2	1,7	^ Engineer is involved in a professional situation in which the public welfare is at stake [3, 9] ^	^ Engineer's action does not hold paramount public welfare [11] ^
2(c) **(III.2.b.)**	Violated	Explicitly discussed	None	None	^ Engineer has a client [1] The client insists that the Engineer completes, signs or seals plans and/or specifications [6] … [1, 2, 3, Inference …]^	^ Engineer does not notify the proper authorities [11] ^

The board cited the following evidence that conflicts with their conclusion:

Code	Code Status	How Cited	Grouped With	Over rides	Why Relevant?	Why Violated, Not Violated, Changed, or Not Applicable?
1 **(III.1., I.4.)**	Not violated	Explicitly discussed	1(c)	None	^ Engineer has a client [1]^	^ Engineer acts as a faithful agent or trustee … [12] ^
1(c) **(III.1.b.)**	Not violated	Explicitly discussed	1	None	^ Engineer has a client [1] Engineer believes the client's project will not be successful [3] ^	^ Engineer advises … the project will not succeed [4] ^
7 **(III.4.)**	Not Violated	Explicitly discussed	None	None	^ Engineer obtains confidential information concerning the business affairs … of a former client [2, 3] ^	^ Engineer does not disclose the confidential information [11] ^

The board cited the following background information that neither directly supports nor directly conflicts with their conclusion:

Case	Citation Type	How Cited	Grouped with	Q #	Why Relevant?	Why Distinguished or Analogous?
67-10	Relevant, But Not Controlling	Explicitly discussed	None	1	^%Engineer is involved in a professional situation in which the public welfare is at stake %[3, 9] ^	^ NA ^

Figure 3-20: The Analysis Representation of Case 76-4-1

Evaluating each node in the A* search space involves the combination of two measures: (1) the quality of the partial solution up to the current node and (2) an estimate of the cost of achieving a solution from the current node. This combination of measures is called the A* cost function and is implemented in SIROCCO as follows:

$$f(n) = g(n) + h'(n)$$

Where n is a node at depth d in the search tree
$g(n)$ is a measure of the degree of mismatched Facts at d
$h'(n)$ is the best possible score to complete the mapping

More specifically, $g(n)$ is equal to the "mismatch cost" at n divided by d. The mismatch cost is a summation of the degree of mismatch at each node up to and including n. The mismatch costs at different levels of abstraction are: 0 for an exact Fact Primitive match, 0.4 for a match at the Fact Group level, 0.6 at the Sibling Group level, 0.9 at the Fact Root level, and 1.0 for a "failed" match. Thus, for a search path at a depth of d = 2 in which a Fact Primitive match and a Fact Group match have been proposed, the mismatch cost is 0.4 (0 + 0.4) and $g(n)$ = 0.2 (0.4 / 2). Based on a parameter setting provided by the user, an adjustment of 0.1 may also be added to mismatches between positive and negative Facts.

The $h'(n)$ function is calculated by dividing the current mismatch cost (i.e., the mismatch cost up to node n) by the fixed solution depth. The solution depth is always fixed to be the number of Facts in the Instantiation (even if a Fact doesn't match, an "empty" match node is created). Thus, $h'(n)$ provides the mismatch cost that would be attained by achieving an exact match (i.e., adding 0) at each node from n until the goal node is reached. For instance, using the example from the previous paragraph and assuming a fixed depth of 4 (i.e., the current Instantiation has 4 Facts), $h'(n)$ = 0.1 (0.4 / 4). Because $h'(n)$ is the most-optimistic possible completion of the mapping, it satisfies the *admissibility* condition [Ginsberg, 1993, p. 78] and guarantees that the algorithm will never return a suboptimal goal node. In particular, SIROCCO always returns the minimum $f(n)$ found at the fixed solution depth.

SIROCCO's search is a variant of A* because the solution depth is always fixed to the number of Facts in the Instantiation (see Solution-Depth in the algorithm of Figure 3-18), rather than a variable, minimum depth, as in standard A*. The solution depth is fixed because the "empty" node generated at each level acts as a catch-all; even if no successful mapping occurs at that level the current path may be continued with this level counted as a "failed" match (i.e., a 1.0 mismatch cost).

Now let us examine a specific example of SIROCCO's search. Figure 3-21 depicts a search tree for the *Code Instantiation* of III.4. ("Do not Disclose Confidential Information Without Consent") of 76-4-1 (see Figure 3-20) mapped to the target case, 90-5-1. This *Code Instantiation* represents the engineer's actions relating to the protection of his client's confidentiality. In particular, the fact that the engineer performed a review (i.e., step 2, "reviews-and-analyzes" in Figure 3-19), discovered a potential safety hazard (i.e., step 3, "discovers-that"), and then did not report the hazard to the authorities (i.e., step 11, "does-not-inform-that"), indicates that he acted to protect confidentiality. These actions, linked to *Code Instantiation* III.4., are ultimately mapped to the target and are represented by the successful search path in the figure (i.e., circled items 1, 2, 4, and 5).

In the figure, search nodes are represented by rectangles. The text inside each node describes that node as: (1) the new mapping of a source Fact to a target Fact, (2) the Fact Primitives that are mapped from source to target by this new mapping, and (3) the previous node mappings that also hold at this node. Actor and Object mappings between source and target are not shown, due to space constraints, but they are also contained in each node. The circled numbers indicate the order in which nodes are expanded, and the number in italics below each node, next to the name of the node, is the match score for that node.

The search in this example did not take long to focus on the optimal path. At the first tier the "does-not-inform-that" Fact Primitive of the source case maps to the identical primitive of the target, at Node 3. Although this is the best path because it matches the Questioned Fact of the source with the Questioned Fact of the target, the search briefly focuses on another possibility, the mapping of "does-not-inform-that" to "informs-that" at Node 2, when the second tier nodes under Node 3 yield higher cost function values than Node 2. However, since this branch does not yield the lowest-cost mapping, the search returns to the path beneath Node 3 and this ultimately leads to the best solution.

Now let us trace the search process in somewhat more detail. The search begins with a calculation of the solution depth. Because there are three Critical Facts in the Instantiation, and the Questioned Fact is also a Critical Fact, the solution depth is 3. (Notice that the search tree of Figure 3-21 has three tiers.) SIROCCO generates an initial search node containing only a mapping between the Questioned Actors of the two cases (i.e., Engineer Doe of 76-4-1 and Engineer A of 90-5-1) and places that node in a search list. An initial node constructed as such assures that the primary protagonists are placed in correspondence in the graph mapping between the cases.

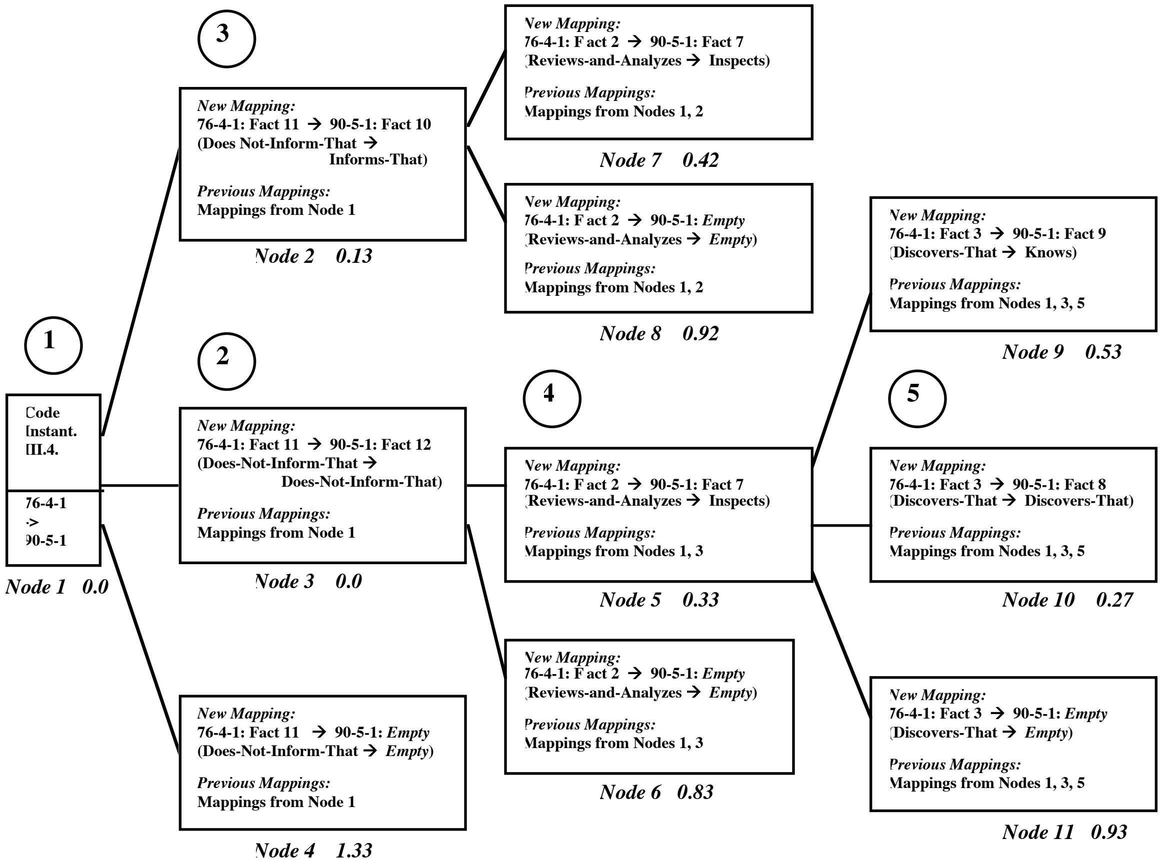


Figure 3-21: The Search Tree for 76-4-1's *Code Instantiation* of III.4. and Case 90-5-1

The main loop of the search terminates when either the search list becomes empty, in which case the search failed to find a mapping, or a goal node is encountered. A goal node is simply a node at the front of the sorted search list that is at a depth equal to the solution depth. The node at the beginning of the sorted search list is always the most promising node and is the one expanded on each iteration.

A node is expanded by generating children nodes that represent each of the possible source-to-target fact mappings from the next unmapped Questioned or Critical Fact. The Source-Target-Mapping-List is ordered by Fact number (i.e., from the chronology) except that Questioned Facts are always placed at the front of the list. This assures that if a Fact in the target can map equally well to multiple Facts of the source, the mapping to the Questioned Fact is attempted first.

In the search example, three nodes are generated at the initial search ply. Two of the nodes correspond to mappings of the source case's Questioned Fact (i.e., Fact 11 in the Fact Chronology of Figure 3-19) to equivalent target Fact Primitives (i.e., Facts 10 and 12 in the Fact Chronology of Figure 3-6). The third node corresponds to the *empty mapping*. An empty mapping is provided at each ply since a Fact later in the path begun by that node could potentially provide a better overall mapping. In other words, the algorithm leaves open the possibility of later expanding down a path that includes this node.

A schematic of the proposed mappings represented by Nodes 2 and 3 of Figure 3-21 is shown in Figure 3-22. Fact Primitives are indicated by ovals, while Actors, Objects, and Internal Fact Phrases are indicated by rectangles. On the left is Fact 11 of source Case 76-4-1. On the right are two Facts from the Fact Chronology of target Case 90-5-1. The proposed mapping from Fact 11 to Fact 10, represented by Node 2, matches the "does-not-inform-that" Fact Primitive of Case 76-4-1 to the "informs-that" Fact Primitive of Case 90-5-1, indicated by the lightly shaded arrow. The corresponding Actors and Objects of the two Facts are also mapped to one another. The proposed mapping from Fact 11 to Fact 12, the one that is ultimately selected by the search, is represented by Node 3 and matches the "does not-inform-that" Fact Primitives of the two cases. Again, the corresponding Actors and Objects of the two Facts are mapped to one another.

Both of the mappings in Figure 3-22 satisfy the four conditions of a valid mapping. A valid mapping and the corresponding expansion of a new node implies that the following four conditions hold on that node:

1. The Fact Primitive of the source case maps to the Fact Primitive of the target case at one of the predefined fact abstraction levels.

2. The new mapping preserves one-to-one mapping between the source case and target case. That is, each source Fact Primitive maps to only one target Fact Primitive and vice versa along the path to the new node. Likewise, each Actor, Object, and Internal Fact Phrase maps to only one Actor, Object, or Internal Fact Phrase, and vice versa.

3. The new mapping preserves or extends existing Actor, Object, and Internal Fact Phrase mappings. That is, Actor, Object, and Internal Fact Phrase mappings that existed on the path to this node are preserved, and the node may introduce new such mappings. Note, however, that by default mappings between Fact Primitives above the Fact Primitive level preserve the existing mappings and do not introduce any new mappings[25].

[25] Notice that this default allows imprecision in the mapping process. For instance, two Facts could map to one another at the Fact Group level with Actor and Object mappings that are inconsistent with the prior set of mappings. The reason for permitting this is simple. For anything other than an exact match, there is no general method for positionally matching the arguments of matching Facts. Because abstract Fact matches anyhow provide less weight to the mapping

4. If temporal matching is turned on, the temporal relationships between the Facts in the source are mapped to the corresponding relations between the Facts in the target. Temporal relations are considered consistent across the cases if the Allen relations of every pair of source Facts intersect with the Allen relations of the corresponding pair of target Facts.

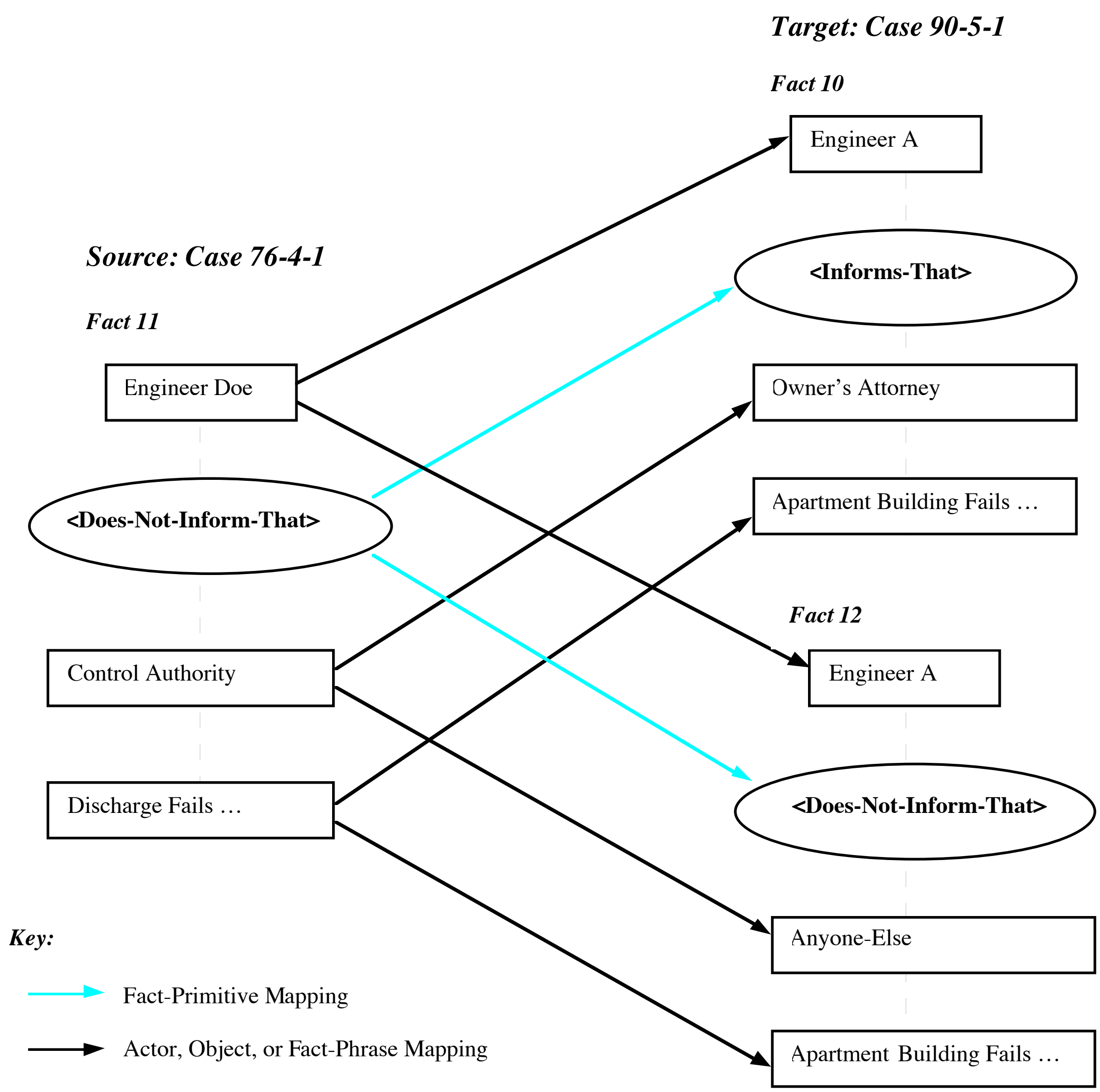


Figure 3-22: Mapping of Facts at the First Ply of the Search Tree

---

score and because these abstract mappings are quite often correct, due to effects of context, it was decided to accept this imprecision and permit such mappings.

Condition 1 is met because the match occurs at the Fact Primitive level for the mapping represented by both Node 2 and Node 3. Because none of the Actors, Objects, or Fact Primitives have yet been mapped, condition 2, one-to-one mapping, is also met by both nodes. The Actor mapping defined at Node 1 (i.e., Engineer Doe mapped to Engineer A) is preserved by both of these mappings, so condition 3 is met. Finally, because this is the first Fact mapping of the search, there are no temporal relations to match; thus condition 4 is trivially met.

The three nodes in the first ply are evaluated and sorted as depicted in the following table. Node 3 is the best node (i.e., it has the lowest match score) because it represents an exact mapping (i.e., "does-not-inform-that" to "does-not-inform-that") from Fact 11 of the source to Fact 12 of the target. The local mismatch cost of an exact mapping is 0.0 and, because this is the first ply, the path mismatch cost is also 0.0. This yields an overall match score of 0.0. Node 2 is not quite as good as Node 3, with a match score of 0.13, because it represents an exact Fact Primitive mapping that is adjusted by a positive-negative difference between the target and source Facts (i.e., "does-not-inform-that" to "informs-that"). Finally, Node 4, the empty mapping, has the lowest match score at 1.33.

Node	**Local Mismatch Cost**	**Path Mismatch Cost**	**(Path-Mismatch-Cost / Current-Depth) + (Path-Mismatch-Cost / Solution-Depth)**	**Match Score**
Node 3	0.0	0.0	(0.0 / 1) + (0.0 /3) =	**0.0**
Node 2	0.1	0.1	(0.1 / 1) + (0.1 / 3) =	**0.13**
Node 4	1.0	1.0	(1.0 / 1) + (1.0 / 3) =	**1.33**

The algorithm continues by expanding Node 3, the best node on the first ply. Two nodes, Node 5 and Node 6, are generated. Node 5 corresponds to the only possible mapping from the next Critical Fact, Fact 2 of source Case 76-4-1, to the target. Note that this matching is inexact. In particular, "reviews-and-analyzes" of Fact 2 maps to "inspects" of Fact 7 of the target. The four mapping conditions are again met, so this is in fact a valid node. In particular, Engineer Doe maps to Engineer A, which is consistent with the mapping from Node 1, and a new object mapping, "Discharge" to "Apartment Building" is introduced. Since neither of these objects has already been mapped, there is a consistent, one-to-one mapping between source and target.

Since there are now two nodes on this potential solution path, it is necessary to check condition 4, i.e., preserving of consistent temporal relations across the source and target. In particular, SIROCCO can now test whether the two mapped Facts in the source, 2 and 11, have an analogous temporal relation to the two mapped Facts in the target, 7 and 12. A Fact in the source is said to *correspond to* a Fact in the target if the source Fact maps to the target Fact. A pair of Facts in the source is *temporally analogous* to a pair of Facts in the target if the following condition holds:

$$ta\ (F_{S1}, F_{S2}, F_{T1}, F_{T2}) = (F_{S1} \lozenge F_{S2})\ \Omega\ (F_{T1} \lozenge F_{T2}) \neq \emptyset$$

Where F_{S1}, F_{S2} are Facts in the source case
F_{T1}, F_{T2} are Facts in the target case
F_{S1} corresponds to F_{T1}
F_{S2} corresponds to F_{T2}
◊ is the temporal relationship between two facts, in terms of the Allen temporal relations
Ω intersection

In other words, given a pair of Facts in the source, F_{S1} and F_{S2}, and a pair of corresponding Facts in the target, F_{T1} and F_{T2}, these pairs are temporally analogous if the temporal relationship between F_{S1} and F_{S2} intersects with the temporal relationship between F_{T1} and F_{T2}.

A *set* of Facts in the source is temporally analogous to a set of Facts in the target if the following holds:

$$TA\ (F_S, F_T) = ta\ (F_{Sx}, F_{Sy}, F_{Tx}, F_{Ty}) \text{ for all x and y}$$

Where F_S is a set of Facts in the source case
F_T is the set of *corresponding* Facts in the target case

In other words, given a set of Facts in the source, F_S, and a set of corresponding Facts in the target, F_T, these sets are temporally analogous to one another if every pair of facts in F_S is temporally analogous to the corresponding pair of facts in F_T.

Because temporal relationships between *all* Facts in a chronology are typically not provided by the case enterer, it is necessary to compute these relationships when they are not defined. SIROCCO does this by calling a time propagation and management system, TIMELOGIC [Koomen, 1989]. TIMELOGIC determines the temporal relationship (i.e., the ◊ relation) between any pair of Facts in a chronology through a process of forward-chaining over the Allen relations.

Returning to the example, the goal is to determine whether *ta* holds between the pairs of corresponding source and target facts represented by Node 5. In particular, assuming F_{S2} = 76-4-1, Fact 2; F_{S11} = 76-4-1, Fact 11; F_{T7} = 90-5-1, Fact 7; and F_{T12} = 90-5-1, Fact 12; *ta* takes the following value:

$$ta\ (F_{S2}, F_{S11}, F_{T7}, F_{T12}) = (\text{B C Fi M O})\ \Omega\ (\text{B C Fi M O}) = (\text{B C Fi M O})$$

Neither (F_{S2} ◊ F_{S11}) or (F_{T7} ◊ F_{T12}) is explicitly defined in the chronology. Thus, TIMELOGIC propagates the temporal relations provided by the case enterer and deduces these values. Because *ta* is non-empty, the temporal condition holds and, thus, Node 5 is a valid node.

The sorted list of nodes after the expansion of Node 3 is shown in the following table. Node 5's local mismatch cost is 0.4, since the source and target match at the Fact Group level, and this yields a match score of 0.33. Node 6, the empty mapping generated at this ply, has a match score of 0.83, a bit better than the previous empty mapping (i.e., Node 4), because of the match at the parent node.

Node	Local Mismatch Cost	Path Mismatch Cost	(Path-Mismatch-Cost / Current-Depth) + (Path-Mismatch-Cost / Solution-Depth)	Match Score
Node 2	-	0.1	(0.1 / 1) + (0.1 / 3) =	**0.13**
Node 5	0.4	0.4	(0.4 / 2) + (0.4 /3) =	**0.33**
Node 6	1.0	1.0	(1.0 / 2) + (1.0 /3) =	**0.83**
Node 4	-	1.0	(1.0 / 1) + (1.0 / 3) =	**1.33**

Next, SIROCCO expands Node 2, since it is now the node with the top match score, and a new ply is created that contains the mapping of "reviews-and-analyzes" to "inspects" (i.e., Node 7 in Figure 3-21) and a new empty mapping node (i.e., Node 8). The match scores at this ply are slightly worse than the previous ply, since Node 2, the parent node, had a slightly worse match score than Node 3, the parent of the previous ply. The new list of sorted nodes is as follows.

Node	Local Mismatch Cost	Path Mismatch Cost	(Path-Mismatch-Cost / Current-Depth) + (Path-Mismatch-Cost / Solution-Depth)	Match Score
Node 5	-	0.4	(0.4 / 2) + (0.4 /3) =	**0.33**
Node 7	0.4	0.5	(0.5 / 2) + (0.5 / 3) =	**0.42**
Node 6	-	1.0	(1.0 / 2) + (1.0 /3) =	**0.83**
Node 8	1.0	1.1	(1.1 / 2) + (1.1 / 3) =	**0.92**
Node 4	-	1.0	(1.0 / 1) + (1.0 / 3) =	**1.33**

The search reverts to the previous ply, as Node 5 is now the most fruitful node. Expanding this node yields three new nodes: one corresponding to an exact match of the primitive "discovers-that" in both the source and target (i.e., Node 10), one corresponding to a match at the Fact Group level between "discovers-that" and "knows" (i.e., Node 9), and one corresponding to the empty match (i.e., Node 11). All of the new nodes satisfy the four conditions of a valid match. For instance, Node 10 satisfies the temporal condition (*TA*), because the three possible *ta* functions at this node yield nonempty values. The previously discussed *ta*, computed at Node 5, still holds here and, in addition, the following functions hold:

$$ta\ (F_{S3}, F_{S11}, F_{T8}, F_{T12}) = (B)\ \Omega\ (B) = (B)$$

$$ta\ (F_{S3}, F_{S2}, F_{T8}, F_{T7}) = (A\ D\ F\ Mi\ Oi)\ \Omega\ (D\ F\ S) = (F\ D)$$

At this stage a solution has been reached. Node 10 is the highest rated node and, because it is at a search depth equal to the solution depth of 3, it is identified as a goal node by the Goal-Node-P function of the Stage 2 algorithm (Figure 3-18). The final match scores are as follows:

Node	Local Mismatch Cost	Path Mismatch Cost	(Path-Mismatch-Cost / Current-Depth) + (Path-Mismatch-Cost / Solution-Depth)	Match Score
Node 10	0.0	0.4	(0.4 / 3) + (0.4 /3) =	**0.27**
Node 7	-	0.5	(0.5 / 2) + (0.5 / 3) =	**0.42**
Node 9	0.4	0.8	(0.8 / 3) + (0.8 / 3) =	**0.53**
Node 6	-	1.0	(1.0 / 2) + (1.0 / 3) =	**0.83**
Node 8	-	1.1	(1.1 / 2) + (1.1 / 3) =	**0.92**
Node 11	1.0	1.4	(1.4 / 3) + (1.4 / 3) =	**0.93**
Node 4	-	1.0	(1.0 / 1) + (1.0 / 3) =	**1.33**

Because 0.27 actually measures mismatch and the maximum possible mismatch is 2.0, the *match percentage* of an Instantiation is calculated as (2.0 – match score) / 2.0. Thus, the match percentage for 76-4-1's *Code Instantiation* of III.4. is 1.73 / 2.0 = 86.7%. The match percentage, rather than the match score, is conveyed to the user when SIROCCO explains its reasoning.

The following table summarizes the mapped solution. Note that this is the mapping explained by SIROCCO in Figure 1-3. The first pair of matching Facts match at the Fact Group level, while the other two pairs of matching Facts match at the Fact Primitive level. The Actors, Objects, and Internal Fact Phrases are mapped consistently across all Fact correspondences (e.g., Engineer Doe maps to Engineer A, Control Authority maps to Anyone Else), and the steps are in the same chronological order across the two cases.

Mapping Level	Source Facts (Case 76-4-1)	Mapped Target Facts (Case 90-5-1)
Fact Group	2. Engineer Doe **<reviews and analyzes>** Discharge	7. Engineer A **<inspects>** Apartment Building
Fact Primitive	3. Engineer Doe <discovers that> (Discharge <fails standards and may be hazardous to safety>).	8. Engineer A **<discovers that>** (Apartment Building **<fails standards and may be hazardous to safety>**).
Fact Primitive	11. Engineer Doe <does not inform> Control Authority <that> (Discharge <fails standards and may be hazardous to safety>)	12. Engineer A **<does not inform>** Anyone Else **<that>** (Apartment Building **<fails standards and may be hazardous to safety>**).

At this point, the value of *Code* and *Case Instantiations* should be abundantly clear. First, the focus on a small set of Facts, i.e., the Questioned and Critical Facts of an Instantiation, allows the program to constrain its search to a relatively small search space, as shown by Figure 3-21. Typically, two to six Facts are linked to an Instantiation. In addition, the Instantiations allow the program to focus on those Facts that are most relevant to the cited code or case. If a source case's Instantiation provides a good match to the Facts of a target case, there is a good chance the associated code or case is also relevant to the target case. This is one of the primary underpinnings of the Analyzer, discussed in the next section.

After Stage 2 has found the best mapping for each of its Instantiations, the Instantiations are sorted by ascending match score. That list, as well as the original top-rated *N* surface cases, is passed on to the Analyzer for the final selection of codes, cases, and additional suggestions.

A couple important efficiency techniques used in the Stage 2 algorithm of Figure 3-18 bear mentioning. First, a simple form of case-based reasoning was used to avoid performing the same search multiple times. Because the Instantiations associated with a source case are often linked with the same Critical Facts as other Instantiations of that source case, it is frequently possible to reuse the results of a previous search. This reuse process is shown in the steps of Collect-and-Sort-Instantiations of Figure 3-18. Whenever the algorithm recognizes that an identical set of source steps have already been mapped to the target (i.e., Critical Facts are not unique), the previous mapping is used instead of starting a new search. Reusing past structural mappings can provide considerable computational benefit, since SIROCCO's A* search is the most expensive aspect of its architecture. Notice, however, that example Case 76-4-1 did not provide this benefit, as each of the *Code Instantiations* related to this case are linked to a unique set of Facts (see the *Code Instantiations* represented in Figure 3-20).

Second, SIROCCO achieves a significant computational saving during the search itself by reusing the fact abstraction matching information computed during Stage 1. Recall that Facts are matched at various levels of abstraction in Stage 1 in order to calculate the weighted dot product. Since identical information is required to map Fact Primitives to one another during search (i.e., The source-to-target mappings at different levels of the Action/Event Hierarchy are used to generate nodes in the search.) that information is dynamically cached in a hash table during Stage 1 and is accessible to the Stage 2 algorithm.

3.1.5. The Analyzer

The Analyzer phase of SIROCCO assesses the results of Stages 1 and 2 and produces the final output of the program, i.e., lists of (1) suggested codes, (2) suggested cases, and (3) additional relevant information. It achieves this by attempting to apply a series of heuristics that are designed to identify the information most likely to be relevant to the user in assessing the current problem situation. A specialized set of heuristics is used to generate each of the three lists produced by the Analyzer.

The Analyzer algorithm is presented, in summary form, in Figure 3-23. The algorithm is composed of three major steps, corresponding to each of the lists of information ultimately produced by the program. The first two steps, which generate the lists of potentially relevant codes and cases, are composed of analogous substeps: Select, Sort, Filter, and Display. These first two steps process the *Code* and *Case Instantiations* collected during Stage 2 and the top surface-matching cases from Stage 1 to produce a list of suggested codes and a list of suggested cases. The final step of the algorithm applies heuristics to the suggested codes and cases of steps 1 and 2 to generate any additional suggestions indicated by operationalization knowledge not involved in earlier stages of the program.

To illustrate the operation of the Analyzer, the processing of example case, 90-5-1, is once again examined and discussed. Figure 3-24 shows an excerpt of the output produced by SIROCCO after it analyzes the example target case. Notice that this output overlaps with that shown in Figures 1-2 and 1-3, except that the excerpt shown here focuses on the heuristics used to select one suggested code and two suggested cases. In this section, the application of these heuristics and how they contribute to SIROCCO's suggestions are explained.

SIROCCO Analyzer

INPUT: *Target-Case*, *Code-and-Case-Instantiations, Top-Stage-1-Source-Cases, Analyzer-Parameters*

OUTPUT: Display the codes, cases, and additional suggestions that are possibly relevant to the *Target-Case*. If requested, provide explanations for selected codes and cases.

1. *Apply Code-Selection Heuristics and Display the Possibly Relevant Codes*

1(a). ***Select the Candidate-Codes by Applying Heuristics***
For each Code-Instantiation in *Code-and-Case-Instantiations*
    For each Code-Selection-Heuristic in Code-Selection-Heuristics (except “Frequent Occ…’)
        If Code-Selection-Heuristic applies to Code-Instantiation Then
            Push Code-Selection-Heuristic details onto Candidate-Codes [Code-Instantiation]
For each Candidate-Code in Candidate-Codes
    If “Frequent Occurrences in Top Cases” heuristic applies to Candidate-Code Then
        Push “Frequent Occurrences in Top Cases” details onto Candidate-Codes [Candidate-Code]

1(b). ***Sort the Candidate-Codes*** by
(1) descending # of applied heuristics,
(2) descending # of appearances in citations of *Top-Stage-1-Source-Cases*, and
(3) ascending match scores in *Code-and-Case-Instantiations*

1(c). ***Filter the Candidate-Codes***
For each Candidate-Code in Candidate-Codes
    If Candidate-Code statistically collocates with all higher rated Candidate-Codes Then
        Push Candidate-Code onto Possibly-Relevant-Codes

1(d). ***Display the Possibly Relevant Codes.***
For each Possibly-Relevant-Code in Possibly-Relevant-Codes
    Display Possibly-Relevant-Code
    If Heuristics-Exp-Flag is on Then, Display explanation of the applied heuristics
    If Mapping-Exp-Flag is on Then, Display explanation of the structural mapping results.

2. *Apply Case-Selection Heuristics and Display the Possibly Relevant Cases*

2(a). ***Select Candidate-Cases by Applying Heuristics***
For each Case-Instantiation in *Code-and-Case-Instantiations*
    For each Case-Selection-Heuristic in Case-Selection-Heuristics
        If Case-Selection-Heuristic applies to Case-Instantiation Then
            Push Case-Selection-Heuristic details onto Case-Candidates [Case-Instantiation]

2(b). ***Sort the Candidates-Cases*** by
(1) descending # of applied heuristics,
(2) ascending match scores in *Code-and-Case-Instantiations*

2(c). ***Filter the Candidate-Cases***
For each Candidate-Case in Candidate-Cases
    If (Candidate-Case has sufficient citation overlap with Possibly-Relevant-Codes) and
       (Candidate-Case cites at least one highly rated code from Possibly-Relevant-Codes) Then
        Push Candidate-Case onto Possibly-Relevant-Cases

2(d). ***Display the Possibly Relevant Cases.***
For each Possibly-Relevant-Case in Possibly-Relevant-Cases
    Display Possibly-Relevant-Case
    If Heuristics-Exp-Flag is on Then, Display explanation of the applied heuristics
    If Mapping-Exp-Flag is on Then, Display explanation of the structural mapping results.

3. *Apply Additional Selection Heuristics and Display the Resulting Information*

For each Code-or-Case-Citation in Possibly-Relevant-Codes and Possibly-Relevant-Cases
    Apply heuristics to find (1) Positive-Negative Fact Primitive matches, (2) *Define the Superior Code* operationalizations, (3) *Apply a Hypothetical to a Code* operationalizations, (4) *Rewrite Code* operationalizations, (5) *Define or Elaborate a General Issue or Principle* operationalizations

Figure 3-23: SIROCCO’s Analyzer Algorithm

```
*************************************************************************
***  SIROCCO has the following suggestions for evaluating
***  '90-5-1: Failure To Report Information Affecting Public Safety'
*************************************************************************

*** Possibly Relevant Codes:
III-4: Do not Disclose Confidential Information Without Consent
    Heuristics Explanation:
    ----------------------------------------------------------------------------------
      o Cited by 4 of the 6 best surface matching cases.
      o 86.7% match to 3 critical facts in case 76-4-1.
      o 86.7% match and ques. fact match (Source and Target: FACT-PRIMITIVE) in case 76-4-1.
      o > 50.0% match in multiple cases: 89-7-1, 76-4-1.
      o Grouped with code I-4 in case 76-4-1; Good match to ques. facts.
      ...
 ...

*** Possibly Relevant  Cases:
76-4-1: Public Welfare - Knowledge of Information Damaging to Client's Interest
    Heuristics Explanation:
    ----------------------------------------------------------------------------------
      o 100.0% match to 3 critical facts in code I-4.
      o 100.0% match and ques. fact match (Source and Target: FACT-PRIMITIVE) in code I-4.
      o 86.7% match to 3 critical facts in code III-4.
      o 86.7% match and ques. fact match (Source and Target: FACT-PRIMITIVE) in code III-4.
      ...

89-7-1: Duty To Report Safety Violations
    Heuristics Explanation:
    ----------------------------------------------------------------------------------
      ...
      o 80.0% match to 5 critical facts (citation to 84-5-1).
      o 80.0% match and ques. fact match (Source and Target: FACT-PRIMITIVE) (citation to 84-5-1).

84-5-1: Engineer's Recommendation For Full-Time, On-Site Project Representative
    Heuristics Explanation:
    ----------------------------------------------------------------------------------
      o 80.0% match to 5 critical facts (citation of 89-7-1).
      o 80.0% match and ques. fact match (Source and Target: FACT-PRIMITIVE) (citation of 89-7-1).

*** Additional Suggestions:
o The codes II-1-A ('Primary Obligation is to Protect Public (Notify Authority if
  Judgment is Overruled).') and I-1 ('Safety, Health, and Welfare of Public is Paramount')
  may override codes III-4 ('Do not Disclose Confidential Information Without Consent'),
  I-4 ('Act as a Faithful Agent or Trustee'), and III-1 ('Be Guided by Highest Standards
  of Integrity') in this case. See case 76-4-1 for an example of this type of code
  conflict and resolution.

...
```

Figure 3-24: An Excerpt of SIROCCO’s Output for Case 90-5-1

The general idea of both steps 1 and 2 is to try to apply all of the heuristics to each of the Instantiations in order to collect a group of possibly relevant citations (i.e., codes and cases). Each applied heuristic is treated as a single piece of evidence that the associated “candidate” citation may be relevant. The codes (or cases) are sorted by the number of heuristics that were successfully applied to each, as well as by additional information used to break ties. The codes (or cases) are then filtered using contextual, domain-specific knowledge. For instance, a code is dropped from the candidate's list if it is not frequently cited together with a higher-rated code, according to historical information. In the final substep, the candidate codes (or cases) remaining after filtering are displayed, along with explanation information, if it has been requested by the user. Because the citations are sorted by number of applicable heuristics, the order in which they are displayed roughly corresponds to SIROCCO's relative degree of belief in their relevance.

The Analyzer is highly parameterized. Each heuristic can be individually turned on or off and has various threshold levels that are user specified. In addition, the filter mechanisms used in steps 1 and 2 have parameterized thresholds. All of the settings used in the example, and subsequently in the experiments discussed in Chapter 4, were empirically established, through experimentation with the foundational cases.

Now let us consider how the Analyzer generates the partial output of Figure 3-24. First, consider the list of possibly relevant codes. Code III.4., the single code shown in the figure out of a total of 10 actually suggested, is suggested because it has a total of eight heuristics that apply to it, five of which are displayed. Code III.4.'s location as second on the list indicates that the program has strong evidence of its relevance. Only Code I.4. ("Act as a Faithful Agent or Trustee"), with ten applicable heuristics, ranks higher. (To see the entire list of possibly relevant codes, refer to Figure 1-2.)

All of SIROCCO's *code-selection heuristics,* along with a brief explanation of each, are shown in Figure 3-25. The items in italics beneath the explanation of some of the heuristics are the parameters and parameter values used by that heuristic. The assigned value for each parameter is the empirically established default used by SIROCCO.

The algorithm attempts to apply all of the code-selection heuristics to the input data it receives from Stage 2. For instance, the heuristic "Frequent Occurrences in Top Cases" is successfully applied to Code III.4. The rationale behind this heuristic is that frequent citations provide accumulated evidence of relevance and, in this particular situation, four of the top-rated six cases from Stage 1 cite Code III.4. Notice that this heuristic specifically employs an operationalization technique, *Reuse an Operationalization*. In particular, it accesses the top-rated cases to determine whether a sufficient number of these cases operationalized (i.e., cited) the code in question, a form of reuse[26].

---

[26] Notice also that the first heuristic is the only code or case selection heuristic that doesn't directly or indirectly involve an Instantiation evaluated by Stage 2. Rather, the heuristic is applied directly to a *code* and uses the list of top-rated

Frequent Occurrences in Top Cases	Satisfied by a code that is cited by a high percentage of the top-rated *N* cases from Stage 1. *[Occurrences-Pct-Thresh = 0.33]*
Good Match to Critical Facts of *Code Instantiation*	Satisfied by a *Code Instantiation* from Stage 2 that has a low match score and at least a minimum number of Critical Facts. *[CF-Match-Score-Thresh = 0.5; CF-Minimum-Facts-Thresh = 2]*
Good Match to Questioned Facts of *Code Instantiation*	Satisfied by a *Code Instantiation* from Stage 2 that has a low match score and a match of Questioned Facts at or below a specified abstraction level.. *[QF-Match-Score-Thresh = 1.0; QF-Abstraction-Thresh = 3]*
Multiple Occurrences of Good Match	Satisfied when multiple *Code Instantiations* of the same code from Stage 2 have low match scores. *[Instantiations-Thresh = 2; Multiple-Match-Score-Thresh = 1.0]*
Grouped with Good Match of *Code Instantiation*	Satisfied by a code that is grouped with a good matching *Code Instantiation* from Stage 2.

Figure 3-25: The Code-Selection Heuristics

The second and third items shown beneath Code III.4. in Figure 3-24 are due to successful application of the heuristics "Good Match to Critical Facts of *Code Instantiation*" and "Good Match to Questioned Facts of *Code Instantiation*." These heuristics are intended to reward *Code Instantiations* that have reasonably good match scores and that also match well to Critical and Questioned Facts, respectively. Recall that Case 76-4-1's *Code Instantiation* of III.4., discussed in the previous section, was mapped to the target case with a match score of 0.27 (equivalent to a match percentage of 86.7%). This match percentage is above the threshold for both heuristics, and the Critical and Questioned Fact match criteria are also met[27]. Thus, both of these heuristics are successfully applied to candidate Code III.4.

The fourth heuristic applied to Code III.4. is "Multiple Occurrences of Good Match." The rationale behind this heuristic is that when multiple cases contain a *Code Instantiation* to the same code, each with a good match score, there is evidence that the corresponding code is relevant. In particular, the heuristic gives credit to a code that is cited by more than one of the top-rated cases, and for each of the citations the corresponding Instantiation is above a specified match percentage threshold. For instance, quality matches to Case 76-4-1's and Case 89-7-1's *Code Instantiation* of III.4., lead to the application of this heuristic. Notice that this heuristic,

---

cases from Stage 1 to decide if the heuristic applies. This detail is shown in step 1(a) of Figure 3-23. This is important because, as we shall see in Chapter 4, this heuristic was employed by two of the competitor methods in the experiment, namely NON-OP-SIROCCO and EXTENDED-MG. This would not have been possible if the heuristic relied on Instantiations, something that is unique to SIROCCO.

unlike the prior two, is based solely on the quality of the match score, irrespective of whether the match includes the Questioned Fact or a certain number of Critical Facts.

The final heuristic supporting the selection of Code III.4. is an application of "Grouped with Good Match of *Code Instantiation.*" The idea is that Code III.4. accrues evidence of its relevancy due to its association with another code that is potentially relevant. This heuristic represents a direct application of the *Group Codes* operationalization technique. The heuristic is applied in this instance because another of 76-4-1's *Code Instantiations*, one applied to Code I.4., matched well to the target, and Code I.4. is grouped with Code III.4. in the analysis representation of Case 90-5-1.

After the heuristics have been applied to all of the Code Instantiations, and the Analyzer has sorted the resultant list of codes, a filter is applied to delete codes from the candidates list that do not collocate well with higher-rated codes. The Analyzer uses historical data collected from all of the cases decided between 1958 and 1992, a total of 475 cases, to determine collocation. Essentially, the filter checks the percentage of times code 1 is cited when code 2 is cited and vice versa. If both values are below a user-specified threshold (default = 0.02), then the lower-rated code is dropped from the candidates list. In the analysis of Case 90-5-1, one code, Code III.3. ("Avoid conduct that will discredit the profession or deceive the public") is filtered from the list at this step, because it does not collocate well with a number of higher-rated codes. (Note that this action is not visible in the output.) The board, in fact, does not cite this code, so this is arguably a correct move by SIROCCO.

After displaying and explaining (if requested) the selected codes, the Analyzer applies a similar sequence of steps to the *Case Instantiations* (step 2 of Figure 3-23). The *code-selection heuristics* are listed in Figure 3-26. Perhaps the most critical heuristic – certainly the one that is applied most frequently – is "Case Cites Matching Code." This heuristic relies on the codes that were selected in step 1 of Figure 3-23. That is, for every *Code Instantiation* that is well matched, the Analyzer adds some heuristic evidence that the citing case is also relevant. In SIROCCO's output depicted in Figure 3-24, this heuristic is responsible for all four of the pieces of evidence shown beneath Case 76-4-1. These satisfied heuristics refer to the *Code Instantiations* of I.4. and III.4. that have already been discussed.

The second and third heuristics in Figure 3-26 are analogous to the similarly named code-selection heuristics. As per their code counterparts, these heuristics are intended to reward *Case*

---

[27] Evaluation of a Questioned Fact mapping is based on a predefined list that orders all of the match possibilities by "goodness" of match. For instance, an exact match between the Questioned Facts of the source and the target yields the best match, a match between the Questioned Facts at the Fact Group level yields the next best match, and so on.

Instantiations that have reasonably good match scores and that also match well to Critical and Questioned Facts, respectively.

Case Cites Matching Code	Satisfied by a case that cites a *Code Instantiation* from Stage 2 that is a good match.
Good Match Critical Facts of *Case Instantiation*	Satisfied by a *Case Instantiation* from Stage 2 that has a low match score and at least a minimum number of Critical Facts. *[CF-Match-Score-Thresh = 0.5; CF-Minimum-Facts-Thresh = 2]*
Good Match Questioned Facts of *Case Instantiation*	Satisfied by a *Case Instantiation* from Stage 2 that has a low match score and a match of Questioned Facts at or below a specified abstraction level. *[QF-Match-Score-Thresh = 1.0; QF-Abstraction-Thresh = 3]*
Good Match Critical Facts of Citing Case	Satisfied by a case from Stage 1 that cites a *Case Instantiation* with a low match score and at least a minimum number of Critical Facts. *[CF-Match-Score-Thresh = 0.5; CF-Minimum-Facts-Thresh = 2]*
Good Match Questioned Facts of Citing Case	Satisfied by a case from Stage 1 that cites a *Case Instantiation* with a low match score and a match of Questioned Facts at or below a specified abstraction level.. *[QF-Match-Score-Thresh = 1.0; QF-Abstraction-Thresh = 3]*
Grouped with Good Match of *Case Instantiation* or Citing Case	Satisfied by a case that is grouped with either (1) a good matching *Case Instantiation* from Stage 2 or (2) a case that cites a good matching *Case Instantiation* from Stage 2.

Figure 3-26: The Case-Selection Heuristics

If a case cites another case, and the cited case is found to be relevant to a new fact situation, there is some reason to believe that the citing case may also be relevant. The heuristics "Good Match Critical Facts of Citing Case" and "Good Match Questioned Facts of Citing Case" represent this notion. In particular, if one of the top-rated cases from Stage 1 contains a well-matched *Case Instantiation*, both the citing case and the Instantiated case accumulate evidence of relevance. This is shown in the output of Figure 3-24 under the selections of Cases 84-5-1 and 89-7-1. The fact that *Case Instantiation* 84-5-1 of Case 89-7-1 is well matched, provides evidence of Case 84-5-1's relevance to the present case. At the same time, Case 89-7-1 heuristically is assigned evidence of its relevance for the same reason.

Recall that Case 84-5-1 involved the overruling of an engineer's judgment by a client in a situation in which safety was at issue. The ethical review board actually cites 84-5-1 in its analysis of Case 90-5-1, but it was not highly rated by Stage 1 of SIROCCO and thus dropped from the analysis until this point. The fact that it is ultimately suggested by the program illustrates that SIROCCO is not totally dependent on the results of its surface retrieval stage. It also shows how SIROCCO is flexible in its use of operationalization knowledge. In particular, the program can bring relevant knowledge to bear at any stage of the process.

The filtering process for cases (i.e., step 2(c) of Figure 3-23) involves, in part, the use of a similarity measure called the *citation overlap*. The citation overlap is based on the principle that when two cases cite the same code or codes from the same category, there is a strong indication that the cases are relevant to one another. Likewise, when one case directly cites another, or when two cases share a citation to a third, there is a strong indication of relevance. Aside from its usage here, this measure is integral to the SIROCCO experiments, and it is discussed in detail and formally defined in Chapter 4.

The Analyzer uses the citation overlap measure to determine when candidate cases have sufficient overlap with the set of possibly relevant codes, suggested in step 1. If a case has sufficient overlap with the set of possibly relevant codes – above a user-specified threshold – and the case cites at least one code from the possibly relevant codes that is highly rated, the Analyzer proposes it as a possibly relevant case. A case that does not sufficiently overlap with the set of possibly relevant codes or that does not cite at least one highly rated code is filtered from the suggested cases list.

In step 2 (d), the Analyzer displays the suggested cases, together with explanations of each individual case (if requested by a user-specified parameter).

The final step of the Analyzer, step 3 in Figure 3-23, is designed to provide qualitative suggestions beyond the possibly relevant codes and cases. This step of the process explicitly brings to bear four of the operationalization techniques, *Define the Superior Code*, *Apply a Hypothetical to a Code*, *Rewrite a Code*, and *Define or Elaborate a General Issue or Principle*. For each of the possibly relevant codes and cases, an attempt is made at this stage to find information, in the form of operationalizations applied in past situations, that may be relevant in the present circumstances. The *additional suggestions heuristics* used for this task are shown in Figure 3-27.

For instance, consider the additional suggestion made by the Analyzer at the bottom of Figure 3-24. Basically, the program suggests that two codes dealing with public safety (II.1.a. and I.1.) may override, in the circumstances of target case 90-5-1, a code dealing with confidentiality (III.4.), a code dealing with loyalty to one's employer (I.4.), and a code dealing with standards of integrity (III.1.). This suggestion is made because Case 76-4-1, a case cited as possibly relevant to 90-5-1, is an example of such a conflict (see the "Overrides" column in Figure 3-20 to verify this), and all of the same codes are suggested by SIROCCO in the present case.

It is instructive to observe that the ethical review board actually did employ a *Define the Superior Code* operationalization in their analysis of Case 90-5-1. To see this, refer to the analysis representation of case 90-5-1 in Figure 3-10. Although the specific code conflict they denote (i.e., Code II.1.a. overrides Code II.1.c., "Do not reveal confidential information without consent") is different, the issue is essentially the same: that of deciding between an obligation to public safety and an obligation to a client's confidentiality.

Suggest Positive-Negative Fact Primitive Matches	Whenever a Fact of a possibly relevant source case matches a Fact of the target case in positive-negative fashion, suggest that this mismatch may be important to the analysis of the case.
Suggest *Define the Superior Code* Operationalizations	Whenever a *Code Instantiation* that is a good match overrides or is overridden by another code that is also considered possibly relevant, suggest that a code conflict may exist in this case.
Suggest *Apply Hypothetical to a Code* Operationalizations	Whenever a *Code Instantiation* that is a good match has a hypothetical applied to it, suggest that this hypothetical may be relevant to the present case.
Suggest *Rewrite a Code* Operationalizations	Whenever a possibly relevant past case cites a changed code, suggest that the change made to the code may be relevant to the current case.
Suggest *Define or Elaborate a General Issue or Principle* Operationalizations	Whenever a possibly relevant past case defines or elaborates a general issue, suggest this definition or elaboration as possibly relevant to the present case.

Figure 3-27: The Additional Suggestions Heuristics

3.2. How Does SIROCCO Use the Operationalization Techniques?

Now that the complete architecture of SIROCCO has been described, it is possible to discuss, in more concrete terms, how and to what degree the computational model implements the operationalization techniques. Figure 3-28 provides a summary of how code operationalizations are implemented in SIROCCO, while Figure 3-29 provides a summary of how case operationalizations are implemented.

Each of the operationalization techniques first introduced in Figure 1-1 is listed in one of the two figures along with: (1) a brief description of the technique's representation within SIROCCO, (2) an explanation of how SIROCCO reasons using that technique within each stage of the program, and (3) an indication of whether the technique contributes to the generation of suggested codes and cases (designated by a "C" in the final column) or whether the technique is focused solely on providing additional, qualitative suggestions (designated by an "S" in the final column).

The Instantiation techniques are the most central and important operationalization techniques employed by SIROCCO. Not only are Instantiations utilized by all stages of the program, but a number of the other techniques *rely* on Instantiations in order to perform their task. For instance, *Define the Superior Code*, used to generate the additional suggestion shown in Figure 3-24, would not have been invoked if the *Code Instantiation* of Code II.1.a. had not first been established. The central role of Instantiations in SIROCCO squares with the observed reasoning of the review board. The board typically relies on cited codes and cases as "anchors" for performing further analysis. In other words, it is typical for the board to "link" the facts of a case to potentially relevant codes and past cases before applying other techniques to those citations, such as posing hypotheticals or elaborating general issues.

Code Operationalizations			
Operationalization Technique	**Representation in SIROCCO**	**Reasoning in SIROCCO**	
Code Instantiation	Defined by the linking of Critical and Questioned Facts to a code citation in the "Why Relevant?" and "Why Violated…?" columns of the analysis representation. A *Code Instantiation* represents the interpretation of a code and the linking of that code to the facts of a case.	**Stage 1.** The weighted dot product of a source case is improved if Questioned Facts and Critical Facts match the target. **Stage 2.** Focuses Stage 2's structural mapping search. SIROCCO attempts only to map the Critical and Questioned Facts of Instantiations to the target case. **Analyzer**. The Analyzer's code-selection heuristics strongly favor codes that have numerous strongly matched *Code Instantiations*.	**C**
Apply a Hypothetical to a Code	Defined by the use of a Hypo as a supporting element of a code citation in the "Why Relevant?" or "Why Violated…?" columns of the analysis representation. A quote (or quotes) from the board's analysis supplies the hypothesized facts.	**Analyzer**. The Analyzer proposes that a hypothetical may be relevant whenever a strongly matched *Code Instantiation* has a Hypo as a supporting element. The hypothetical is proposed as possibly relevant in the "Additional Suggestions" section of SIROCCO's output.	**S**
Rewrite a Code	Defined by the use of the "Changed" value in the "Code Status" column of a code citation in an analysis representation. This status denotes that the code has either recently changed or that a change is suggested in the present case.	**Analyzer**. The Analyzer proposes that a code rewrite may be relevant whenever one is associated with a possibly relevant case. The code rewrite is proposed as possibly relevant in the "Additional Suggestions" section of SIROCCO's output.	**S**
Define the Superior Code	Defined by the inclusion of other codes in the "Overrides" column of a code citation in an analysis representation. Supplying codes in this column denotes that the present code takes precedence over the others in the current context.	**Analyzer**. The Analyzer proposes that one code may override others whenever a strongly matched *Code Instantiation* exhibits such an override and all of the same codes are possibly relevant in the present context. The code conflict is proposed in the "Additional Suggestions" section of SIROCCO's output.	**S**
Group Codes	Defined by the inclusion of other codes in the "Grouped With" column of a code citation in an analysis representation. Supplying codes in this column signifies that this code is grouped with the other codes for force of argument in the current context.	**Analyzer**. The Analyzer's code-selection heuristic "Grouped with Good Match of *Code Instantiation*" selects codes as possibly relevant when they are grouped with strongly matched *Code Instantiations*.	**C**

Figure 3-28: How the Code Operationalizations are Implemented in SIROCCO

Case Operationalizations			
Operationalization Technique	**Representation in SIROCCO**	**Reasoning in SIROCCO**	
Case Instantiation	Defined by the linking of Critical and Questioned Facts to a case citation in the "Why Relevant?" and "Why Violated…?" columns of the analysis representation.	**Stage 1.** The weighted dot product of a case is improved if Questioned Facts and Critical Facts match. **Stage 2.** Focuses Stage 2's structural mapping search. SIROCCO attempts only to map the Critical and Questioned Facts of Instantiations to the target case. **Analyzer**. The Analyzer's case-selection heuristics strongly favor cases that have numerous strongly matched *Case Instantiations*. In addition, the cases that cite strongly matched *Case Instantiations* are also strongly favored.	**C**
Define or Elaborate a General Issue or Principle	Defined by case citations supplied in the "Background Information" category with a citation type of "Relevant, But Not Controlling" in an analysis representation Such case citations typically introduce or define general scenarios, issues, or principles.	**Analyzer**. The Analyzer proposes that a general scenario, issue, or principle may be relevant whenever a possibly relevant case cites a case that employs this operationalization. The general scenario, issue, or principle is proposed in the "Additional Suggestions" section of SIROCCO's output.	**S**
Reuse an Operationalization	Not explicitly defined in SIROCCO's representation. Implicitly defined whenever a case is cited, since virtually every cited case provides operationalizations that are reused by the computational model.	**Stage 1, Stage 2, Analyzer**. This technique is implicitly employed in SIROCCO's overall strategy of selecting codes, cases, and additional suggestions. Because SIROCCO takes a case-based approach, all of its suggestions rely on operationalizations used in past cases.	**C**
Group Cases	Defined by the inclusion of other cases in the "Grouped With" column of a case citation in an analysis representation. Supplying cases in this column signifies that this case is grouped with the other cases for force of argument in the current context.	**Analyzer**. The Analyzer's case-selection heuristic "Grouped with Good Match of *Case Instantiation*" selects cases as possibly relevant when they are grouped with strongly matched *Case Instantiations*.	**C**

Figure 3-29: How the Case Operationalizations are Implemented in SIROCCO

Reuse an Operationalization is the most general and all-encompassing technique. It relies on the application of operationalizations in past cases and the analogical transfer of those applications to new cases. Because SIROCCO takes a case-based approach, *Reuse an Operationalization* is arguably the core implementation idea of the architecture. It is also fair to say that in connection with this technique the computational model diverges from the review board in behavior. While SIROCCO relies completely on its case base to identify suggestions to provide to the user – in other words, it is a pure case-based reasoner – the board is capable of reasoning from first principles when, for instance, it interprets codes or applies hypotheticals.

A specialized application of *Reuse an Operationalization* is important in connection with the experiments discussed in Chapter 4. Two of SIROCCO's competitor methods, namely NON-OP-SIROCCO and EXTENDED-MG, effectively use a form of *Reuse an Operationalization* to generate suggested codes. In particular, both of these methods employ the first heuristic shown in Figure 3-25, "Frequent Occurrences in Top Cases," to generate code suggestions. Thus, it can be said that both methods use at least a weak form of the operationalization techniques. Because EXTENDED-MG outperforms the system it is based upon, MG, this is significant: it provides additional evidence that the operationalization techniques do make a difference in supporting accurate suggestions.

Five of the nine operationalization techniques support SIROCCO in suggesting possibly relevant codes and cases. Those five techniques are *Code Instantiation*, *Group Codes*, *Case Instantiation*, *Reuse an Operationalization*, and *Group Cases* (indicated by a "C" in the final column of Figures 3-27 and 3-28). These techniques ultimately have the goal of suggesting, indicating, or pointing to particular codes and cases. The other four techniques are used exclusively for providing additional, qualitative suggestions (denoted by a "S" in the final column). In other words, the latter four techniques imply that some *explanation* or *discussion* of the codes and cases is required. Consider, for instance, that *Apply a Hypothetical to a Code* requires language to "fill in" unstated facts. This distinction between the two sets of operationalizations is critical because it is only the former techniques that are objectively verifiable through a direct comparison of SIROCCO's output to that of the review board[28].

Fortunately, the five verifiable operationalization techniques include the techniques that are most central to operationalization, the *Code* and *Case Instantiations*. Thus, testing this core group of techniques is highly indicative of how the operationalization techniques fare as a whole. The experiments reported in Chapter 4 directly test the accuracy of this critical subset of the operationalization techniques.

3.3. How Does SIROCCO Use Temporal Knowledge?

As discussed in Section 3.1.4, SIROCCO uses temporal relations as one component of the graph matching process of Stage 2. In particular, during node expansion of the Stage 2 search algorithm, SIROCCO checks that the temporal relations between the Critical and Questioned Facts of the source case are consistent with the (proposed) corresponding Facts of the target case. If the relations are consistent, SIROCCO allows node expansion and investigation of the particular mapping represented by that node. If the relations are inconsistent, SIROCCO disallows the expansion of this node.

[28] Strictly speaking, the other operationalizations *could* be compared to the explanations provided by the board. I contend, however, that such a process would drastically impact objectivity because of the need for extensive language interpretation.

In order to illustrate how temporal relations can benefit SIROCCO in its task, this section presents a brief example in which the time relationships between Facts play an integral role in the suggestions produced by the program[29]. Consider the two cases shown in Figure 3-30.

Facts of Case 91-6-1:

Facts: Engineer A contracts to serve as a consultant to a federal environmental agency for the development of an overall hazardous waste remedial strategy. Under the contract with the federal agency, Engineer A agrees to provide basic consulting services along with an understanding that the federal agency may request additional services at a later date. Nothing is contained in the contract between Engineer A and the agency concerning other work for other clients. Two years following completion of basic services to the federal agency, Engineer A is retained to provide environmental consulting services by a major industrial corporation which has been deemed by the federal agency to be responsible in a dispute over the clean-up of a hazardous waste site. Following the execution of its contract with the corporation, Engineer A is contacted by the federal environmental agency and is asked to provide consulting services to the agency per Engineer A's original understanding with the agency in connection with the specific hazardous waste site of the major industrial corporation which is now a client of Engineer A. Engineer A informs the federal agency that the performance of such services would constitute a conflict of interest and declines to perform the services requested.

Question:

Was it unethical for Engineer A to agree to perform services to the industrial corporation under the facts without the prior consent of the federal agency?

Facts of Case 62-7-1:

Engineer Z, an engineering consultant, has been retained by a County Metropolitan Commission to perform all necessary engineering and advisory services. The Commission does not have an engineering staff, so Engineer Z acts as the staff for the Commission in the preparation of sewerage and water studies, the establishment and financing of sanitary districts, and reviews and approves plans submitted by other engineers.

Engineer Z has also been retained by a private company to perform the engineering design for a development of several thousand housing units. Involved are extensive contract negotiations between the Commission and the developer for the construction and financing of sanitary and water facilities for the development. As consultant to the Commission, Engineer Z will under the circumstances have a key role in the negotiations.

Question:

Does the dual role of Engineer Z constitute a conflict of interest?

Figure 3-30: An Example Target Case, 91-6-1, and an Example Source Case, 62-7-1

Both of these cases deal with a *conflict of interest*, a common issue faced by many professional engineers. The first case, 91-6-1, is SIROCCO's target case for the example. In this case, Engineer A provides and completes engineering consulting services for a federal agency regarding the development of a hazardous waste strategy. There is an "understanding" between Engineer A and the agency that Engineer A may be required to perform additional services in the future. However, there is no stipulation in the contract regarding Engineer A's work for other clients. Two years later, Engineer A performs and completes consulting services for a corporation that has been cited by the federal agency for violation of hazardous waste policy.

[29] The example in this section was actually generated by SIROCCO; it is not hypothetical.

Subsequently, the federal agency contacts Engineer A, cites the original understanding between the engineer and the agency, and asks that Engineer A provide consulting services to the agency in connection with the hazardous waste of the corporation. Engineer A declines the agency's request, citing a conflict of interest. The question raised by this scenario is whether it was ethical for Engineer A to have provided services for the private corporation without having first received approval of the federal agency.

In Case 62-7-1, Engineer Z is retained by a governmental body to provide all engineering and advisory services in connection with sewerage and water studies. This service includes the review and approval of plans submitted by other engineers. At the same time, Engineer Z is also retained by a private development company to perform design services for a new housing development. The sanitary and water facilities of the new development ultimately must be approved by Engineer Z's other employer, the governmental body, and Engineer Z himself. Thus, Engineer Z is placed in the position of both designing (for the private company) and approving (for the governmental body) the new facilities.

A conflict of interest appears to be present in both cases. In fact, the review board found the actions of both Engineer A and Engineer Z to be unethical, citing several conflict of interest codes. SIROCCO is also capable of identifying the presence of a conflict of interest in Case 91-6-1, in part by retrieving and mapping the target case to source Case 62-7-1. However, because of a critical difference between 91-6-1 and 62-7-1 – a difference based on mismatched temporal relations between the cases – SIROCCO is also capable of correctly rejecting the following code cited by the board in their analysis of Case 62-7-1:

> "*Code III.1.c*. Engineers shall not accept outside employment to the detriment of their regular work or interest. Before accepting any outside employment they will notify their employers."

The intent of this code is to avoid a special type of conflict of interest, the kind that arises when an engineer has an on-going job or assignment and *at the same time* accepts other work that may be detrimental to the regular assignment. Notice how the two cases differ significantly on this particular issue. In Case 91-6-1, Engineer A's employment with the corporation begins long after his contract with the agency concludes. Although there is an informal understanding that the agency *may* need his services at a later date, a long period of time passes without Engineer A providing any services for the agency. Thus, the board concluded that engineer A's work for the agency was not "regular work or interest" and was thus outside the scope of Code III.1.c. On the other hand, the board does cite Code III.1.c. in their analysis of Case 62-7-1. In this case, Engineer Z has a "regular" assignment for a governmental body and, *at the same time*, accepts a conflicting assignment with a private company. Engineer Z's assignment with the company is detrimental to his role with the governmental body, because will be asked to approve what he himself has designed.

By citing Code III.1.c. in their analysis of Case 62-7-1, but not in their analysis of 91-6-1, the review board acknowledges at least some difference between the two cases. That is, although 62-7-1 and 91-6-1 clearly raise conflict of interest issues, the specific nature of the conflict is somewhat different between the two cases.

Now let us examine how SIROCCO is able to identify this difference between the cases. First, consider a version of the program that *does not* take temporal considerations into account. Shown below is a structural mapping between *Code Instantiation* III.1.c. of source Case 62-7-1 and target Case 91-6-1 that does not impose temporal constraints.

Mapping Level	**Source Facts (Case 62-7-1)**	**Mapped Target Facts (Case 91-6-1)**
Fact Group	2. Engineer Z **<is employed by>** County Metropolitan Commission **<as>** Engineer & Advisor.	1. Federal Environmental Agency **<hires the services of>** Engineer A **<for>** Hazardous Waste.
Fact Primitive	3. Engineer Z **<is hired to provide services for>** Developer D.	6. Major Industrial Corporation **<hires the services of>** Engineer A **<for>** Hazardous Waste.
Fact Group	4. Engineer Z **<in his capacity as>** Advisor **<takes the action>** (Engineer Z **<negotiates with>** Developer D **<for>** Housing Development H).	8. Engineer A **<provides engineering services on>** Hazardous Waste **<for>** Major Industrial Corporation.

In this structural mapping the key mappings between the Facts of the source and the target are the first two. These are the Facts that indicate that each engineer worked for two separate clients. If temporal considerations are disregarded, i.e., if the fourth condition of a valid mapping, discussed in Section 3.1.4, is dropped, the above table represents a valid mapping from the source to the target. Because of the resulting favorable match score of this *Code Instantiation*, a version of SIROCCO that does not take temporal considerations into account would ultimately, but incorrectly, propose III.1.c. as a possibly relevant code.

Now, consider how the enforcement of temporal relationships changes the structural mapping. Below is a mapping of the same Instantiation, except that it is generated by a version of SIROCCO that *does* take temporal considerations into account.

Mapping Level	Source Facts (Case 62-7-1)	Mapped Target Facts (Case 91-6-1)
Fact Primitive	3. Engineer Z <is hired to provide services for> Developer D.	6. Major Industrial Corporation <hires the services of> Engineer A <for> Hazardous Waste.
Fact Group	4. Engineer Z <in his capacity as> Advisor <takes the action> (Engineer Z <negotiates with> Developer D <for> Housing Development H).	8. Engineer A <provides engineering services on> Hazardous Waste <for> Major Industrial Corporation.

Notice that the mapping between Fact 2 of the source and Fact 1 of the target, the Facts corresponding to the engineers' first employment, is no longer part of the structural mapping. This is so because the temporal relations between Facts 2 and 3 of the source do not overlap with the temporal relationships between corresponding Facts 1 and 6 of the target. In particular, we have

$$ta\ (F_{S2}, F_{S3}, F_{T1}, F_{T6}) = (E)\ \Omega\ (B) = \emptyset$$

In other words, Facts 2 and 3 of the source occur at the same time[30], while Facts 1 and 6 of the target occur in non-overlapping sequence. Because these pairs of Facts are not temporally analogous with one another, the node representing a structural mapping of all of these Facts is rejected as invalid. Ultimately, the "lesser" mapping, shown in the table immediately above, is identified as the best possible mapping by a version of SIROCCO that enforces temporal constraints. Because this mapping is assigned a relatively poor match score, the Instantiation is ultimately rejected and Code III.1.c. is not proposed as a possibly relevant code. This is the correct action, and note that it is only taken by the version of SIROCCO that enforces temporal constraints.

This example illustrates how SIROCCO is able fruitfully to use temporal relationships to help in its assessment of new cases and in proposing possibly relevant citations. Because of a number of such examples uncovered during my analysis of the NSPE BER cases (see Section 2.2.3), combined with some early results of the computational model, I came to the conclusion that temporal relationships are important to the assessment of engineering ethics cases. In particular, I proposed the secondary thesis of this dissertation, that SIROCCO's temporal knowledge makes a difference in the accuracy of its predictions.

[30] In this example, I use the Time Qualifiers actually supplied by an independent case enterer. Arguably, Facts 2 and 3 do not occur at precisely the same time. More likely, Fact 3 occurs during Fact 2. Notice, however, that this would not affect the example, as the calculation of *ta* would still yield the empty set for such a time specification.

As a side note, it is worthwhile mentioning that this example also illustrates the wide range of engineering ethics cases that can be addressed by SIROCCO. Although conflicts of interest are not one of the Selected Topics for which SIROCCO was specifically designed to address, in this instance the program capably handled a case involving that issue. In other words, this is an example of SIROCCO addressing an issue outside its primary scope of knowledge.

3.4. Explanation in SIROCCO

One of the key aspects of SIROCCO, particularly in comparison to the alternative methods with which the program is empirically compared in this dissertation, is its capability to explain its reasoning. Explanation is particularly important when one considers the engineering ethics domain and the ultimate intended use of the tool. In engineering ethics, and perhaps in any professional ethics domain, it is important for the decision maker to be able to carefully deliberate on his or her decisions. This involves, at a minimum, identifying which facts are relevant in light of applicable principles, resolving conceptual issues by defining terms of the principles and their application to case facts, and engaging in moral reasoning (e.g., use past cases for "line-drawing" comparisons, employ creative middle-way solutions) [Harris *et al.*, 1999].

To support a human engaged in these analytical steps, a computational model must provide more than lists of relevant codes and cases. For instance, it is important that the human user understand which of the Facts of a target case are morally relevant and which codes (i.e., principles) these relevant Facts relate to. This is precisely the type of information provided in SIROCCO's explanations of its structural mappings (see, for instance, Figure 1-3). Displaying the steps of the target that match to possibly relevant codes allows the user to discern the facts that might be considered "critical" in the context of the new case. More specifically, the *Code* and *Case Instantiations* of past cases allow the human interlocutor to focus on and assess relevant facts and pay less attention to those facts that may not be relevant.

SIROCCO's structural mapping explanations also help the user with conceptual issues. Although the program is not capable of directly defining open-textured terms, such as those discussed in the Boisjoly example of Chapter 1, its retrieval and structural mapping capability effectively allows it to retrieve *exemplars* of such definitions in the context of past cases. In some sense, an Instantiation's extensional representation – that is, the links from a code or case to the Facts of past cases – provides an exemplar definition of the terms that are relevant to the code or case. By comparing and contrasting the Facts of the current target to exemplars of past Instantiations, it is possible for the user to deduce, at least to a rudimentary degree, the board's intended definitions of various terms.

SIROCCO's "additional suggestions" can also be helpful in supporting the user's moral deliberation. For instance, the suggestion that a conflict between codes might exist in Case 90-5-1 (see the bottom of Figure 1-2 and Figure 3-24) provides the user with potentially important information for understanding the moral issues involved in a scenario. Without such explanation, the user might tend to look at the codes in isolation. Or, even if they recognize that there is a conflict between codes, they may not know of an example of its resolution, something that SIROCCO's explanation provides by indicating the past case or cases that exhibited such a conflict.

Ultimately, the user of SIROCCO is responsible for performing the actual moral reasoning and coming to a conclusion about new cases. However, SIROCCO, equipped with a rudimentary explanation capability, provides valuable support in focusing attention on the relevant factual and conceptual issues of an engineering ethics case.

4. The SIROCCO Experiments

The previous chapter described the architecture of SIROCCO, showing how the program implements and reuses the operationalization techniques that were observed in the NSPE BER's analyses of engineering ethics cases. As discussed above, a core subset of these techniques, in particular the *Code Instantiations* and *Case Instantiations*, is very important. Also, the use of temporal knowledge within SIROCCO was described, formalized, and demonstrated. The question now is whether the proposed architecture and implemented functionality of SIROCCO can be shown empirically to verify the theses proposed in this dissertation.

This chapter presents a series of experiments performed with SIROCCO. The experiments were designed to test: (1) the *primary thesis*, that is, whether the core set of operationalization techniques lead to accurate suggestions of relevant codes and past cases, (2) the *secondary thesis*, that is, whether SIROCCO's temporal knowledge similarly supports accurate suggestions, and (3) *computational characteristics* of the program, for instance, trade-offs between accuracy and efficiency and the scalability of the program.

4.1. The Experimental Design

As with any formal evaluation, the SIROCCO experiments required a series of design choices, summarized in Figure 4-1. As discussed below, making these choices provided a general framework for all of the experiments.

4.1.1. The Top Level Experimental Approach

The most fundamental aspect of the experimental design is the use of a benchmark function to score SIROCCO, ablated versions of SIROCCO, and the competitor methods. In this approach the output of a method[31] is compared to a hand-coded representation of the actual NSPE BER analyses. The hand-coded representation is the analysis representation of a case, discussed in Section 3.1.2. The goal of the benchmark is to quantify the degree with which the method simulates the analysis produced by the board. To the degree that a method's output agrees with the review board's opinions, that method is deemed successful.

[31] Henceforth, the term "method" is used in a general sense to include SIROCCO and all of the computational methods with which it is compared.

Top-Level Experimental Design Approach:	Benchmarking, followed by external system comparisons and ablation comparisons. *Benchmarking* – Evaluation of a single method by comparing the output of that method with the hand-coded analyses of the ethical review board (i.e., the NSPE BER) for the same cases. *External System Comparison* – Comparison of SIROCCO with other computational methods by benchmarking each and then comparing the benchmark results. *Ablation Comparison* – Comparison of a fully featured version of SIROCCO with versions of the program from which functionality is excised. Again, the benchmark is used as the basis of comparison.
Comparison Points:	Specific items of comparison, for purposes of calculating the benchmark, are the suggested codes and cases of SIROCCO (or one of the other methods) compared with the codes and cases cited by the ethical review board for the same cases.
Experiment Metric:	The *F-Measure*, an information retrieval metric that combines *precision* and *recall*, is used to quantify the benchmarks. Informally, the F-Measure computes a form of overlap of the codes and cases between methods, and that overlap can be either exact or inexact. Exact overlap occurs when the identical code or case is found in both methods. Inexact code overlap is computed using the *Code Hierarchy* as a basis. Inexact case overlap is calculated using a *citation overlap metric*.
Parameter Settings:	SIROCCO and each of the other computational methods has its own unique set of parameters. The experiment parameter settings for SIROCCO were established using informal, but extensive, pre-tests. The experiment parameter settings for each of the other methods were decided by choosing the best run of pre-tests in which parameters were varied over a large number of runs (i.e., for each method 100 runs of the 184 foundational cases using the benchmark comparison).
Test Cases:	A *trial set* of 58 trial cases were chosen randomly from two categories of cases decided between 1993 and 1998 by the ethical review board. The first category was the Selected Topics and the second category was the Non-Selected Topics. A majority of the cases (44) were chosen from the Selected Topics category. Additional test results were obtained from the *foundational set* of cases, the 184 cases decided between 1958 and 1992 used to develop, debug, and pre-test SIROCCO.
Statistical Model:	Because of the non-Gaussian nature of the F-Measures obtained for every benchmark, a *nonparameteric bootstrap procedure* was used to compare pairs of methods.

Figure 4-1: Summary of the Experimental Design of the SIROCCO Experiments

Using the benchmark as a basis, two general tactics were used to test SIROCCO's capabilities and the hypotheses of this dissertation. First, several other computational methods were identified as baselines, or competitors, to SIROCCO and those other methods were implemented or obtained to process the NSPE BER cases. Second, the parameters of SIROCCO were used to create ablated versions of the program.

The competitor methods were first presented in Figure 1-4 and described in Section 1.3.4. The first two competitor methods, RANDOM and INFORMED-RANDOM, essentially represent worst-case baselines. RANDOM uses purely random selection of codes and cases. INFORMED-RANDOM selects randomly from a pre-computed distribution of cited codes and cases that mirrors the frequency with which codes and cases were cited by the review board in decisions between 1958 and 1992. In addition, INFORMED-RANDOM limits its selected citations to only the 20 most-frequently cited codes and the 40 most-frequently cited cases.

Finding a computational method or methods that are more competitive with SIROCCO than the random approaches was not easy, since engineering ethics is a novel AI domain that provides no directly comparable computational model. However, because engineering ethics is a heavily linguistic domain – as we have seen, its cases and analyses are in complex natural language – *full-text retrieval*, a computational approach that calculates similarity between a target and a source document based on similar language or similar concepts, seemed a likely area in which to find a strong competitor to SIROCCO. The use of such an approach is well understood and established in the information retrieval field [Frakes and Baeza-Yates, 1992; Korfhage, 1997]. A further advantage of full-text retrieval is that it requires no knowledge representation, thus making it relatively straightforward to apply. In recent years, much attention has been given to improving upon this technology within the case-based reasoning literature [Burke *et al.*, 1997; Lenz, 1998; Brüninghaus and Ashley, 1998; Brüninghaus and Ashley, 1999]. If SIROCCO can be shown to outperform such a system, there is evidence that a representation-intensive approach to case-based reasoning, such as that implemented in SIROCCO, is still viable.

MG [Witten *et al.*, 1999], a public-domain full-text retrieval system based on the vector space model, was selected as one competitor. An extension of that method, EXTENDED-MG, which I developed, was also included. MG retrieves possibly relevant cases and codes using a pure full-text-retrieval approach, in particular, by applying vector comparisons between the target text, represented as a n-dimensional term vector, and the text of all of the cases and codes, also represented as n-dimensional vectors. In a term vector, the terms or concepts are assigned weights proportional to the frequency of the term or concept in the document and inversely proportional to the frequency of the concept in the corpus. To calculate similarity, the direction, or more specifically the angle of direction, of the target vector and source vectors are compared. The smaller the angle, the more similar the cases. EXTENDED-MG also uses this approach in order to retrieve cases, but codes are selected from the retrieved cases based on the SIROCCO heuristic "Frequent Occurrences in Top Cases" (see Figure 3-25). Recall that this heuristic implements an aspect of the operationalization technique *Reuse an Operationalization*; thus, EXTENDED-MG employs operationalization knowledge, albeit only weakly, to help in its retrieval.

Two ablated versions of SIROCCO were used in the experiments. To test the primary thesis, a version of SIROCCO that only very weakly employs operationalization techniques, NON-OP SIROCCO, was included in the first experiment along with the competitor methods described above. NON-OP SIROCCO uses Stage 1 and the dot product calculation to retrieve and score cases, but it does not use Questioned Facts, Critical Facts, structural mapping, or most of the heuristics of SIROCCO's Analyzer to perform its task. To select codes, NON-OP SIROCCO, as EXTENDED-MG does, uses the heuristic "Frequent Occurrences in Top Cases." To test the secondary thesis, an ablated version of SIROCCO, sans the temporal knowledge, was tested. This version of the program, called NON-TEMP SIROCCO, provides the full functionality of SIROCCO with the exception that it doesn't check the consistency of temporal relations across matched cases, i.e., condition 4 of a valid mapping discussed in Section 3.1.4.

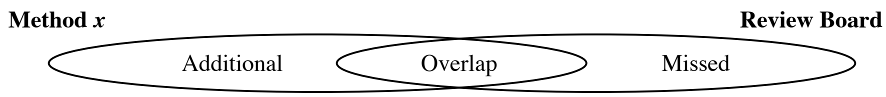
4.1.2. The Comparison Points

The comparison points for the experiment were the code and case citations of the methods and the review board. Each of the methods discussed in Section 4.1.1, as well as SIROCCO, accepts a *target case* and produces two output lists: (1) a list of possibly relevant codes and (2) a list of possibly relevant cases. These lists, in turn, were compared to the corresponding citation lists produced by the review board, as hand-coded by the case enterers, for the same cases.

Although SIROCCO also produces "additional suggestions" that could theoretically be compared to the opinions and analyses of the review board, the suggestions are in natural language. Thus, they would need to be interpreted and somehow compared to the language of the board's analyses, unlike the suggested codes and cases that can be directly compared. Such an interpretive process is highly subjective, so this aspect of SIROCCO was not evaluated in the experiments reported in this dissertation. It is worth mentioning, however, that SIROCCO is the only method that produces such suggestions. Thus, SIROCCO produces potentially helpful information not provided by any of its competitors.

4.1.3. The Experiment Metric

To calculate overlap between a method's solution and the board's solution, an information retrieval metric known as the *F-Measure* [van Rijsbergen, 1979, p. 173-176; Lewis *et al.*, 1996] which combines precision and recall, was used. Informally, the F-Measure was used to compute a form of overlap of the codes and cases between two methods. A Venn diagram depicting the overlap and the equations for precision, recall, and the F-Measure are shown in Figure 4-2.



Method *x* = The code and case suggestions produced by SIROCCO or one of the other methods.
Review Board = The code and case citations of the ethics review board.
Overlap = The code and case citations shared by both Method *x* and the review board.
Additional = The extra code and case suggestions made by Method *x* but not cited by the review board.
Missed = The set of code and case suggestions lacking in Method *x* but cited by the review board.

Precision (*P*) = Overlap / (Overlap + Additional) **Recall (*R*)** = Overlap / (Overlap + Missed)

F-Measure[32] (Method *x*, Review Board) = $\frac{(\beta^2 + 1)PR}{\beta^2 P + R}$

Figure 4-2: Calculation of the F-Measure in the SIROCCO Experiments

Two F-Measures were calculated for each method for each case, one representing combined exact matches of codes and cases between the method's solution and the board's, and one representing combined inexact matches of codes and cases. Codes and cases were combined to focus and simplify the comparisons between methods. The combined exact F-Measure was calculated by comparing codes and cases separately but treating them as being in one group for purposes of calculating precision and recall. In other words, the corresponding numerators and denominators for the precision and recall calculations of the code and case comparisons were added together. For instance, if Method *x* suggested Code A, Code B, and Case X, and the review board cited Code A and Case X for the same case, then, assuming ß = 1.0, the F-Measure calculation would be:

P (Method x, Review Board) = (1 + 1) / (2 + 1) = 0.67
R (Method x, Review Board) = (1 + 1) / (1 + 1) = 1.0
F-Measure (Method x, Review Board) = (2 * 0.67 * 1.0) / ((1 * 0.67) + 1.0) = 0.8

The combined inexact F-Measure required different treatment of the codes and cases. Codes were considered matched if they resided in the same category of the Code Hierarchy. Exact code matches were first identified, tallied, and removed. The remaining codes in the "Additional" and "Missed" categories (see the Venn diagram of Figure 4-2) were then pairwise compared to check for matches at higher abstraction levels of the Code Hierarchy. Any abstractly matching pairs were considered "equal" for the purposes of the inexact F-Measure calculation. An abstractly

[32] ß < 1.0 gives greater weight to precision; ß = 1.0 gives equal weight to recall and precision; ß > 1.0 gives greater weight to recall. ß = 1.0 was used for all experiments reported in this dissertation.

matched pair is any pair that shares the same top-level abstraction category. There are a total of 22 such categories.

Calculating the case-matching component of the combined inexact F-Measure was more complicated. Cases were not grouped by a predefined knowledge structure such as the Code Hierarchy, and so there was no static means of computing equivalence or relevance between pairs of cases. Yet it was also true that if the board did not cite a case, say Case A, this did not necessarily imply that Case A was irrelevant. Rather, it might indicate that the board was unable, or perhaps did not have the time, to find such a case for their analysis. Because there are significantly more cases to choose from than codes, it is far less likely that the board will cite a given case than a given code. Alternatively, the board might have simply believed they had made themselves clear without citing the case. The board cited, on average, significantly fewer cases than codes per case. For the 475 cases decided between 1958 and 1992, the board averaged 2.24 code citations per case and 1.44 case citations per case. It was not unusual, in fact, for the board to cite no cases in their analyses – this happened in 40% (73 out of 184) of the foundational case analyses – while only very few of the board's case analyses had no code citations.

Anyhow, it was necessary to find a fair way of assigning credit to each method for finding cases that seemed to be (or are) relevant, even if they did not show up in the board's opinions. In other words, credit was required to be given for a relevant case suggestion, even if there was no corresponding case cited by the target case. With respect to Figure 4-2, this notion was quantified as follows: For each case suggestion made by Method *x*, credit was given for an inexact match to either (a) the target case or (b) one of the cases the board cited in its analysis of the target. There might be multiple matches to the target, but each cited case could match only once. For the purposes of the F-Measure calculation, therefore, each inexact match increased "Overlap" by 1 and decreased "Additional" and "Missed" by 1. When "Missed" reached 0, it was no longer decremented. Notice that in the relatively common circumstance of the board citing no cases, the calculation amounted to simply incrementing "Overlap" by 1 for each inexact match. If a target case cited at least one case, the "Overlap" for the recall calculation would have a maximum value equal to the number of cases cited by the target.

The altered version of the F-Measure metric used to handle inexact matching – particularly that of case citations – is a reasonable and fair metric with which to benchmark the computational methods. While the precise definitions of precision and recall were slightly altered, the spirit and intent of the metrics remain intact. In particular, the altered versions of the F-Measure, precision, and recall capture the "correctness" of a method's selections with respect to both the board's citations and additional knowledge sources (i.e., the Code Hierarchy and the citation overlap metric).

To calculate the combined inexact F-Measure, the corresponding numerators and denominators of code and case precision and recall were added, as with the exact calculation. As an example, suppose Method *x* suggested Code A, Code B, Case X, and Case Y, while the review board, for the same case, cited Code C and Case Z. If Code A and Code C inexactly matched, Case X inexactly matched the target case, but Case Y did not inexactly match either Case Z or the target, then the F-Measure calculation would be:

P (Method x, Review Board) = (1 + 1) / (2 + 2) = 0.5
R (Method x, Review Board) = (1 + 1) / (1 + 1) = 1.0
F-Measure (Method x, Review Board) = (2 * 0.5 * 1.0) / ((1 * 0.5) + 1.0) = 0.67

Implementing Inexact Case Matching

The above calculation depends, of course, on a means for identifying inexact matches between cases. In the NSPE BER corpus, it was clear that *citation overlap* – a measure of *code citation overlap* combined with *case citation overlap* – is the most objective and feasible measure of similarity. When two cases cite the same code, or codes from the same category (i.e., *code citation overlap*), there is a strong indication that the cases are similar, or at least relevant, to one another. Likewise, when one case directly cites another, or when two cases share a citation to a third case (i.e., *case citation overlap*), there is again a strong indication that the cases are similar or relevant to one another. There is also evidence from the literature that citation overlap is a viable means of comparison. For instance, it has been used as a basis for comparing and ranking AI research journals [Cheng *et al*., 1996].

As part of the calculation of inexact matching between cases, the F-Measure was again used, but in a different capacity, as the metric to calculate the code citation overlap between the cases. Intuitively, the F-measure was a good choice because it measures the degree with which "issues" are shared between cases. Shared codes represent, roughly speaking, common issues between cases, while unshared codes represent issues relevant to one case but not the other. The more issues two cases share, proportionally speaking, the more likely they are to be relevant to one another. The F-measure effectively computed this through a form of intersection that also imposed a penalty for lack of intersection (i.e., missing relevant codes, including irrelevant codes). Matching, or overlap, between two codes was calculated by taking 1 divided by the level at which they share an ancestor in the Code Hierarchy. ß was set to 1.0 to give equal weight to precision and recall in the F-Measure calculation.

With respect to case citation overlap, a shortest path algorithm was applied between cases. Viewing the case base as a network, with cases as nodes and case citations as edges, the shortest path between Case 1 and Case 2 was defined as the minimum number of edges between Case 1 and Case 2. Case citation overlap was computed as follows:

$$\textit{Case Citation Overlap}\ (\text{Case1, Case2}) = \frac{1}{ShortestPath(Case1, Case2)} \text{ or 0.0, if there is no path}$$

Thus, for instance, a direct citation between two cases resulted in case citation overlap = 1.0 (1/1), and a case citation shared by two cases (i.e., a case node existing between the two cases) resulted in case citation overlap = 0.5 (1/2).

The final step in computing citation overlap is combining code citation and case citation overlap. Citation overlap was defined as the weighted sum of the constituent overlap functions:

$$\textit{Citation Overlap}\ (\text{Case1, Case2}) = \frac{(\alpha - 1)CodeCitationOverlap(Case1, Case2) + CaseCitationOverlap(Case1, Case2)}{\alpha}$$

For the experiments, α was fixed at 5.0, favoring the code citation overlap by a 4-1 ratio over the case citation overlap. In other words, while the presence of case citation overlap is a strong indicator of relevance, the absence of case citation overlap is not a strong indicator of irrelevance.

While it is impossible objectively and precisely to define the citation overlap threshold, a value ≥ 0.35 was taken to be high enough to consider two cases relevant to one another. This threshold was determined based on a sampling of approximately 30 pairs of foundational cases.

Some Comments Regarding the Suitability of the F-Measure

The board's analyses provide a useful benchmark for comparing the output of the various methods. The NSPE BER is an established, authoritative committee, and its published opinions appear to be highly regarded within the engineering profession. Thus the board's opinions provide, arguably, the best available and objective benchmark. And although the F-Measure may not be a perfect tool to evaluate SIROCCO's performance against a benchmark, it certainly approximates overlap. Further, it is not clear that there is a better tool to use for the types of objective experiments reported in this dissertation. Perhaps more importantly, the metric is used to calculate the benchmark for all of the methods compared in the experiments and only evaluates the features common to all methods (e.g., there is no advantage to longer explanations). Thus, any slight advantage or disadvantage that may result from the altered metrics applies uniformly across all methods.

However, the use of the F-Measure, precision, and recall to evaluate SIROCCO's performance against some quantifiable benchmarks in this experiment does not come without disadvantages. As pointed out by Edwina Rissland and colleagues, the "traditional IR assumption of an unequivocal master answer key is a weighty one" [1997, p. 17]. Because of inherent inconsistencies (e.g., changes in the composition of the board over time lead to different opinions and citations), the board's analyses are somewhat uneven: some are excellent and highly analytical, some are less thoughtful and somewhat shallow. The altered F-Measure applied to inexact matches partially addresses this by supplementing the board's opinions with additional knowledge, but it does not solve the problem altogether. Another issue is the relative sparseness of citation lists. As previously mentioned, the board averages less than 3 code citations and less than 2 case citations per case; many of the board's analyses have only a single citation. This means, of course, that very small numbers are used in the precision and recall calculations, leading to widely ranging values. For instance, if the board cites only two codes and a method suggests neither of them, both precision and recall are 0. However, if the method suggests just one code and that code matches one of the two board citations, the precision score rises dramatically to 1.0, while the recall score increases to 0.5.

4.1.4. Parameter Settings

SIROCCO and all of the other methods have parameter settings that control program functionality. The parameter settings of SIROCCO used in the experiments were decided based on extensive, but informal, pre-tests. Because SIROCCO has a large number of parameters (over 30), some of which accept a wide range of values, it was impossible to exhaustively pre-test the program in search of optimal settings. Instead, small ranges of values for each parameter were identified as the most likely to be successful and various tests were run with these settings.

Most of SIROCCO's parameters and the particular settings used in the experiment were discussed in Chapter 3. The key settings are shown in Figure 4-3. Notice that the flag for temporal matching, *Temporal-Match-P*, was turned on for all experiments except for Experiment #3, the one specifically designed to test the absence of temporal matching.

The flag to check for positive-to-negative Fact Primitive matching, *Positive-to-Negative-Match-P*, was turned off[33]. SIROCCO's Analyzer module has a large number of parameters, not shown in Figure 4-3, corresponding to turning each of the heuristics on and off and providing thresholds for each. For the experiments, all of the heuristics were turned on and the threshold settings used were those shown in Figures 3-24, 3-25, and 3-26.

Method	Parameters	Experiment Setting	Explanation of Parameter
SIROCCO	*N*	6	The number of top-rated source cases to retrieve.
	CV-Weight-List	(1.0, 0.5, 0.25, 0.125)	Combine content vectors at 4 abstraction levels according to these weights.
	QF-Weight-List	(1.0, 0.5, 0.25, 0.125)	Apply questioned fact weight according to the 4 abstraction levels.
	CF-Weight-List	(0.333, 0.111, 0.036, 0.012)	Apply critical fact weight according to the 4 abstraction levels.
	Temporal-Match-P	T (Except for Experiment #3)	Turn on/off temporal reasoning.
	Positive-to-Negative-Match-P	NIL	Turn on/off positive-to-negative fact matching.
	Date-Filter-P	T	Specify whether only earlier-decided cases may be cited.

Figure 4-3: The Key Parameters and Experiment Settings of SIROCCO

A date filter flag, *Date-Filter-P*, is used to specify when SIROCCO may cite only earlier-decided cases for the target versus citing any case, earlier or later. In all of the experiments, the date filter was turned on for SIROCCO, as well as for all of the other methods, meaning that only earlier cases could be cited. Turning the date filter on provided more realistic output and also allowed each method to be more equitably compared to the benchmark. In other words, since the board obviously could not cite future cases, neither should the methods.

In fact, the key reason why all of the trial cases were chosen from a later time period (i.e., 1993-1998) than the foundational cases (i.e., 1958-1992) was to set up this more realistic evaluation and to assure that a reasonably large pool of cases could be cited by each and every trial case. The evaluation was more realistic since a case from, say 1993, could not cite a case in the future from, say 1994. And the large pool of cases was assured because only later cases, i.e., those that can cite a large number of earlier cases, were used in the tests.

[33] This parameter, discussed briefly in section 3.1.4 and illustrated in the search tree of Figure 3-21, controls whether SIROCCO checks for positive-to-negative matches of Fact Primitives and accordingly reduces the match score for such matches. A positive-to-negative match occurs when the base form of a Fact Primitive (e.g., "employs") is matched with a negative form of that primitive (e.g., "does not employ"). The parameter-tuning pre-tests showed that positive-to-negative matching provides only marginal improvement in SIROCCO's accuracy. This finding, combined with the fact that checking for positive-to-negative matches is computationally expensive, since it is applied at every new search node, led to the decision to turn the parameter off for the experiments.

To assure that each of the methods compared to SIROCCO in the experiments was run with optimal parameter settings, extensive parameter-tuning pre-tests were executed. Each method was run 100 times on all 184 foundational cases, with different settings used for each execution run. To determine the best run for each method, the mean F-Measure of exact matches was added to the mean F-Measure of inexact matches for each run. The highest sum of F-Measures over the 100 runs was taken to be the best for that method and the corresponding parameter settings were then used in all of the experiments. Figure 4-4 provides a summary of the five competitor method's parameters and settings as determined by the parameter-tuning pre-tests.

The date filter flag, *Date-Filter-p*, is not shown in Figure 4-4 as it took the same value for all methods. The flag was turned on (i.e., set to "T"), meaning that only earlier cases could be cited, as discussed above.

The MG program has a few additional parameters, not shown in Figure 4-4, mostly dealing with print features [Witten *et al.*, 1999, p. 464]. Although vector space models, of which MG is an example, can often be parameterized for performance along a number of dimensions, such as the use of different combining functions, term weight calculations, and relative term frequency functions, the public-domain version of the software tested in these experiments came with pre-set, fixed defaults which provide good performance on average. This means that optimizing the internal algorithm for this particular problem was not possible. On the other hand, research has shown that it is extremely difficult to identify particular settings that provide the best overall performance [Zobel and Moffat, 1998]. Optimizing along one dimension often means suboptimizing along another.

In the experiments, MG's vector space model is applied separately to the database of textual cases and to the database of textual codes, the former for case retrieval, the latter for code retrieval. EXTENDED-MG uses MG to perform case retrieval but then applies the heuristic technique "Frequent Occurrences in Top Cases" to the retrieved cases in order to retrieve codes.

In MG's vector space representation, the weight of each term is inversely proportional to its frequency in the entire corpus. In other words, terms that appear frequently are valued less in the match process, while terms that appear seldom are highly valued in the match process. MG automatically computes this when it compiles the corpus.

In the parameter-tuning pre-test, three corpora were tested, each covering the entire set of foundational cases but containing different textual information per case. One corpus contained only the textual description and question raised of each case (FACTS); one contained the textual description, question, and textual code citations (FACTS-CIT); and one contained the textual description, question, citations, and the full analysis provided by the review board (ALL). The pre-tests established that the last of these textual databases, ALL, produced the most accurate results for MG.

Method	Parameters	Experiment Setting	Explanation of Parameter
RANDOM	*Max-Code-Citations-Per-Case*	8	Suggest a random number of codes, between 1 and *Max-Code-Citations-Per-Case*, per target case.
	Max-Case-Citations-Per-Case	1	Suggest a random number of cases, between 1 and *Max-Case-Citations-Per-Case*, per target case.
INFORMED-RANDOM	*Max-Code-Citations-Per-Case*	10	Suggest a random number of codes, between 1 and *Max-Code-Citations-Per-Case*, per target case.
	Max-Case-Citations-Per-Case	9	Suggest a random number of cases, between 1 and *Max-Case-Citations-Per-Case*, per target case.
	Number-of-Top-Codes	20	Suggest codes only from the most frequently cited *Number-of-Top-Codes*.
	Number-of-Top-Cases	40	Suggest cases only from the most frequently cited *Number-of-Top-Cases*.
MG (Managing Gigabytes)	*MG-Case-Base-Type*	ALL	Search textual database of facts only (FACTS), facts and citations (FACTS-CIT), or facts, citations, and analysis (ALL).
	Max-Cases-to-Consider	10	The maximum number of case citations suggested per target case. The exact number of suggested cases is based on *MG-Case-Threshold*.
	MG-Case-Threshold	0.5	Suggest all cases with scores within *MG-Case-Threshold* percent of the top-rated case.
	Max-Codes-to-Consider	8	The maximum number of code citations suggested per target case. The exact number of suggested codes is based on *MG-Code-Threshold*.
	MG-Code-Threshold	0.5	Suggest all codes with scores within *MG-Code-Threshold* percent of the top-rated code.
EXTENDED-MG	*MG-Case-Base-Type*	ALL	Search textual database of facts only (FACTS), facts and citations (FACTS-CIT), or facts, citations, and analysis (ALL).
	Max-Cases-to-Consider	2	The maximum number of case citations suggested per target case. The exact number of suggested cases is based on *MG-Case-Threshold*.
	MG-Case-Threshold	0.25	Suggest all cases with scores within *MG-Case-Threshold* percent of the top-rated case.
	Max-Cases-to-Consider-for-Codes	10	Suggest codes only from those cited in the top-rated *Max-Cases-to-Consider-for-Codes*
	MG-Code-Threshold	0.25	Suggest all codes that are cited in at least *MG-Code-Threshold* percent of the top-rated *Max-Cases-to-Consider-for-Codes*.
NON-OP SIROCCO	*N*	10	The number of top-rated source cases to retrieve.
	CV-Weight-List	(1.0, 0.5, 0.25, 0.125)	Combine content vectors at 4 abstraction levels according to these weights. Same as SIROCCO.
	Non-Op-Case-Threshold	0.8	Suggest all cases in the top *N* cases with a Match Score ≥ *Non-Op-Case-Threshold*
	Non-Op-Code-Threshold	0.25	Suggest all codes that are cited in at least *Non-Op-Code-Threshold* percent of the top-rated *N* cases.

Figure 4-4: The Parameters and Experiment Settings of the Five Competitor Methods

4.1.5. The Test Cases

SIROCCO and the other methods were evaluated using a *trial set* of 58 cases[34]. To ensure objectivity – in particular, to avoid criticism that the experiment was biased because test cases were hand-coded by the experimenter – two independent case enterers were enlisted to hand-code all of the trial cases. The trial cases encoded by the case enterers were used in unedited form in the experiments.

Before the trial cases were selected, 20 cases from the grand total of 97 cases decided by the NSPE BER between 1993 and 1998 were eliminated. Those that were eliminated were either clearly not transcribable into EETL (because of ambiguous or vague language, scenarios that did not involve actions or events, or heavy reliance on Actors, Objects, or Fact Primitives not yet provided in the language) or they contained citations to codes that were encountered either never or very rarely within the analyses of the foundational cases

The trial cases were then chosen randomly from two subsets of the remaining 77 cases. Forty-four were chosen at random from 52 Selected Topics cases, and 14 were chosen at random from 25 Non-Selected Topics cases. Two of the 44 original Selected Topics cases were rejected by one case enterer as not transcribable and these were replaced by two other Selected Topics cases chosen at random. Four additional cases were transcribed but not included in the trial set because they did not cite any codes that were cited a minimal number of times in the foundational set of cases.

As explained in Chapter 1, the Selected Topics cases are those cases that cite at least one code related to one of the following subject areas: public safety, confidential information, duty to employer, credit for engineering work, proprietary interests, and honesty in reports and public statements. Examples of Non-Selected Topics are conflicts of interest, competence, and unfair competition. The main reason for using such a ratio was to select a significant majority of cases from SIROCCO's supposed area of expertise, since a similar majority of the foundational cases also come from this category, while at the same time selecting at least some cases outside of that area of expertise. Such a distribution of cases allowed for significant testing of the program on what it should "know" but also allowed for at least limited testing of the program outside of its main area of expertise.

Despite the deletions for cases that cannot be represented in EETL or that only cite seldom-used codes, the remaining trial set actually covers 60% of the entire, available population of cases

---

[34] Originally, there were 62 trial cases , but 4 were dropped from the experiment because the review board did not cite any supporting or conflicting code or case citations, meaning that the F-Measure would be guaranteed to be 0,

decided between 1993 and 1998 (58 / 97). All conclusions and generalizations drawn from the experiments of course exclude the two categories of cases that were deleted from the population.

4.1.6. The Statistical Model

The data generated by benchmarking each method against the review board's citations using the F-Measure turned out to be highly non-Gaussian. In other words, the data did not exhibit anything approaching the bell-shaped distribution required for the application of standard Gaussian statistics. The F-Measures of all data sets distributed widely between 0 and 1, instead of clustering around the mean, as in a well-behaved bell curve. This is not surprising in light of earlier comments (see Section 4.1.3.) which stated that the relatively sparse analyses of the board lead to highly variable and distributed precision and recall values.

Thus, it was necessary to apply nonparametric statistical methods in order to compare the benchmarks of SIROCCO with the other methods. In particular, the *nonparametric bootstrap procedure* [Efron and Tibshirani, 1993; Davison and Hinkley, 1997] was applied. This procedure requires only that the data observations be independent and identically distributed from the population to be studied. The data observations, i.e., the F-Measures, are independent because the value of one observation has no effect on or relationship with any of the other observations. It is also reasonable to assume that the data observations are identically distributed, i.e., the likelihood of seeing any particular F-Measure is the same from observation to observation and matches the population distribution.

The nonparametric bootstrap procedure is executed as follows. A data set comprises all of the F-Measure readings of one method for either the exact or the inexact comparisons. Thus, in the SIROCCO experiments, a data set contained 58 readings. For each data set, 58 readings were drawn at random. When a reading was drawn, it remained in the set, eligible to be drawn again; this process is known as "sample with replacement." The mean after drawing 58 elements was then calculated. This process of drawing 58 readings and calculating the mean is repeated X times, with X empirically set to 100,000 for these experiments[35]. Summary measures (95% Confidence Interval (CI), p-value) are then calculated from the 100,000 estimates of the mean. In these experiments, the summaries corresponded to the 2.5 and 97.5 percentiles[36], obtained by sorting the 100,000 estimated means and then taking the 2,500th and 97,500th values. The p-value

---

regardless of method applied. These 4 cases were included in Experiments #4, #5, and #6 to pad the size of the case base and are referred to as the "discarded trial cases."

[35] Testing of the SIROCCO data with the nonparametric bootstrap indicated that 100,000 repetitions was sufficiently large enough to achieve accurate and stable results.

[36] The 2.5 and 97.5 percentiles are traditionally used to calculate a 95% CI.

for a hypothesis of a mean difference = 0 is calculated by doubling the proportion[37] of the 100,000 mean differences between two methods that are less than or equal to zero. Method x is considered significantly better than Method y if the p-value for a mean difference = 0 is less than 0.05 (5%).

Any two methods can also be visually, but informally, compared using a paired differences plot. Since all of the methods operate on an identical sample set, such a plot is created simply by subtracting the corresponding F-Measures obtained for the two methods and accumulating the results in various “differences” buckets. If a concentration of differences with relatively high amplitudes is clearly shown on one side or the other of the 0 difference hash, then the method corresponding to that region is obviously superior for that data set.

The fact that the experiment ultimately yielded so many significant findings indicates that the sample size of 58 was sufficiently large to grant adequate statistical power. Statistical power measures the ability of a test to detect differences from the null hypothesis, and in 9 out of 10 of the comparisons in the first experiment, this was achieved. In other words, the conclusions reached through testing with the 58 trial cases indicate a real effect and not one due to statistical fluctuations inherent in small sample sizes.

The nonparametric bootstrap procedure is the primary quantitative tool for experiments #1, #2, and #3, in which the primary and secondary theses of the dissertation were tested. Experiments #4, #5, and #6, which tested accuracy versus efficiency and the ability for SIROCCO to scale up, are evaluated in a less-formal manner, through inspection and interpretation of various data plots.

4.2. Testing the Primary Thesis

Two experiments were run to test the primary thesis. These experiments are discussed and the results reported in this section. In the first experiment, SIROCCO’s accuracy over the trial cases, as benchmarked against the board’s citations and gauged by the F-Measure, was compared to the accuracy of the five other methods, i.e., EXTENDED-MG, MG, NON-OP SIROCCO, INFORMED-RANDOM, and RANDOM. In the second experiment, the accuracy of SIROCCO was also compared to EXTENDED-MG over the trial cases, but in this experiment both methods were given credit for additional code and case citations not made by the board, but deemed reasonable by two experienced ethical reasoners. For both experiments, accuracy of both exact and inexact citation matching is reported.

---

[37] Because this is a two-sided test i.e. the alternatives are equal and not equal.

4.2.1. Experiment #1: Comparison of SIROCCO to Other Methods

Experiment #1 Process

In Experiment #1, each of the six methods, including SIROCCO, was run against each of the 58 trial cases. For each run of a single trial case, each method had at its disposal a database of approximately, but no more than, 241 cases that could be used to help decide cases and codes to suggest. More specifically, each method had access to the 184 foundational cases, plus the 58 trial cases, minus the target case and each of its sibling cases[38]. The date filter was turned on for all trial runs, meaning that each method could cite only those cases decided earlier in time. Both earlier and later cases could, however, be used by each method as supporting data in selecting codes to suggest. This choice was made simply to provide each method with an identical pool of cases from which to make its code selections.

For each method run against each trial case, the exact and inexact F-Measures were calculated by comparing that method's output to the citations made by the review board for corresponding cases, as discussed in Section 4.1.3. The mean exact and inexact F-Measure over all trial cases per method were calculated, and the nonparameteric bootstrap procedure was applied pairwise to comparisons of SIROCCO with each of the five other methods.

Quantitative Results

As can be seen in Figure 4-5, SIROCCO attained the highest mean F-Measure for both exact (0.21) and inexact (0.46) matching over the 58 trial cases. The closest competitor to SIROCCO for exact matching was EXTENDED-MG, at 0.14, and the closest competitor for inexact matching was MG, at 0.38. NON-OP SIROCCO's mean F-Measure for exact matching was third best, while for inexact matching it was fourth best. Not surprisingly, both of the random selection methods, i.e., INFORMED-RANDOM and RANDOM, performed worse than any of the other methods, according to the mean F-Measure.

Although the mean F-Measure is a useful approximation of the relative accuracy of SIROCCO versus the other methods, it does not say whether the differences between SIROCCO and the other methods are statistically significant. The nonparametric bootstrap procedure

[38] Sibling cases share the same fact situation. Deleting the sibling cases of a target case corrects for the anomalous situation in which a target cites its own sibling or uses its sibling's analysis, which very likely overlaps with its own analysis. A case with one sibling would have access to 240 cases, one with two siblings would have access to 239, etc. None of the trial cases has more than 3 sibling cases.

provides this information. The results of the procedure applied to pairs of methods for exact matching is shown in Table 4-1,

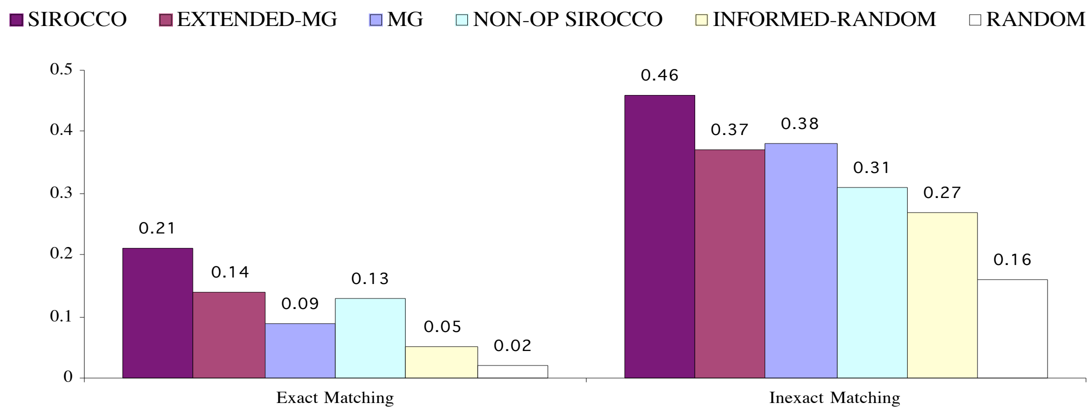


Figure 4-5: Mean F-Measures for all Methods over all of the Trial Cases

Table 4-1: Experiment #1, Exact Matching Comparisons Using the Nonparametric Bootstrap

Methods Compared	95% Confidence Interval for mean difference	p-value for mean difference=0
SIROCCO vs. EXTENDED-MG	(0.018, 0.121)	0.008
SIROCCO vs. MG	(0.074, 0.180)	< 0.001
SIROCCO vs. NON-OP SIROCCO	(0.039, 0.126)	< 0.001
SIROCCO vs. INFORMED-RANDOM	(0.116, 0.210)	< 0.001
SIROCCO vs. RANDOM	(0.139, 0.240)	< 0.001

The information in the table is interpreted as follows. Each row represents a pairwise comparison of two of the methods. The first column contains the compared methods. In our case, we are interested in how SIROCCO compares to each of the other methods, so there are five rows in the table. The second and third columns show the results of the nonparametric bootstrap procedure applied to that comparison. The first value in the second column represents the $2{,}500^{th}$ mean difference of Method 1 minus Method 2, while the second value in the second column represents the $97{,}500^{th}$ mean difference. Because the 100,000 differences for each method are sorted, these two values demarcate the 95% Confidence Interval for mean difference. Roughly speaking, this is a range estimate of the average difference in F-Measure between Method 1 and

Method 2 for exact matching. For instance, the average mean difference between SIROCCO and EXTENDED-MG, shown in row 1 of Table 4-1, is between +0.018 higher for SIROCCO and +0.121 higher for SIROCCO. The p-value is calculated by first subtracting the corresponding 100,000 means of the compared methods. Then the proportion of those calculations (x / 100,000) that are less than or equal to zero is multiplied by two. The resulting value is the probability of a mean difference = 0; in other words, the probability that Method 1 is no more accurate than Method 2, according to the F-Measure.

Table 4-1 shows that SIROCCO is clearly superior to all five of the other methods on the exact matching criteria. For each comparison, the interval for mean difference is positive, in favor of SIROCCO, and the probability of SIROCCO having a higher mean is greater than 99%. Taking 95% to be the threshold, as is customary, all five of the findings in this table are statistically significant.

Table 4-2 shows the results of the nonparametric bootstrap procedure applied to pairs of methods (i.e., SIROCCO versus each of the other methods) for inexact matching. In this analysis, SIROCCO is superior, in a statistically significant way, to MG, NON-OP SIROCCO, INFORMED-RANDOM, and RANDOM. The only exception is the comparison of SIROCCO to EXTENDED-MG, for which the probability of a mean difference = 0 is 94.3%, a marginally significant result.

Table 4-2: Experiment #1, Inexact Matching Comparisons Using the Nonparametric Bootstrap

Methods Compared	95% Confidence Interval for mean difference	p-value for mean difference=0
SIROCCO vs. EXTENDED-MG	(-0.003, 0.188)	0.057
SIROCCO vs. MG	(0.016, 0.149)	0.015
SIROCCO vs. NON-OP SIROCCO	(0.090, 0.224)	< 0.001
SIROCCO vs. INFORMED-RANDOM	(0.113, 0.266)	< 0.001
SIROCCO vs. RANDOM	(0.211, 0.386)	< 0.001

These results indicate that SIROCCO is clearly superior to MG, NON-OP SIROCCO, INFORMED-RANDOM, and RANDOM with respect to the trial cases. Further, the results strongly suggest that SIROCCO is also superior to EXTENDED-MG with respect to the trial cases. SIROCCO's superiority to EXTENDED-MG on exact matches is significant, and its superiority on inexact matches, while not quite significant, is very nearly so.

The comparisons between SIROCCO and the other methods can be visualized by inspecting Figure 4-6 and Figure 4-7. The bar charts in these figures correspond to the same method comparisons found in Tables 4-1 and 4-2 (i.e., SIROCCO vs. EXTENDED-MG, SIROCCO vs. MG, etc.), respectively, but the data that is plotted is closer to the raw data (i.e., the original F-Measure readings). In particular, each chart shows an accumulation of the differences in F-Measure readings between SIROCCO's performance on each trial case versus the other method's performance on the same case. The pairwise differences are accumulated in "buckets" and plotted. For instance, the first chart at the top of Figure 4-6, comparing SIROCCO with EXTENDED-MG, shows that for 16 trial cases there was no difference (i.e., difference = 0) between SIROCCO's F-Measure and EXTENDED-MG's F-Measure. Bars to the right of the 0 hash provide tallies of cases for which SIROCCO's F-Measure was greater than EXTENDED-MG's F-Measure. Bars to the left of the 0 hash indicate tallies of cases for which EXTENDED-MG's method yielded a higher F-Measure.

The charts provide intuitive visual support of the nonparametric bootstrap results. For instance, it is clear that a higher concentration and amplitude of bars is on the right side of the 0 hash, indicating SIROCCO's superiority, in virtually every chart of Figures 4-5 and 4-6. One visible exception is the first chart of Figure 4-7, the one representing the comparison of SIROCCO to EXTENDED-MG on inexact matching. Here, there is quite a dispersion of results; thus it is not visually obvious that SIROCCO's method outperforms EXTENDED-MG's. In fact, as has been discussed, this is the only comparison in which statistical significance was not achieved. Another chart in which it is somewhat difficult to see a clear distinction is the second one of Figure 4-7. In this chart, which compares SIROCCO and MG on inexact matching, there is a fair concentration of bars on both sides of the 0 hash. This is also consistent with the results shown in Table 4-2. SIROCCO is superior to MG in inexact matching with respect to the trial cases, but with a somewhat less decisive p-value than other comparisons.

The dispersion of results in the first chart of Figure 4-7 compared to the less-dispersed results of the second chart of that figure also explains why SIROCCO is less convincingly superior to EXTENDED-MG on inexact matching than it is to MG on inexact matching, even though EXTENDED-MG's mean F-Measure for inexact matching (0.37) is lower than MG's (0.37) over the trial cases (see Figure 4-5). A wide dispersion of results, particularly outliers at either end of the x-axis, correlate more strongly with higher p-values than do less disperse results. In other words, the variability and wide dispersion of results in the first chart of Figure 4-7 are strong visual indicators of the less than significant p-value found in row 1 of Table 4-2.

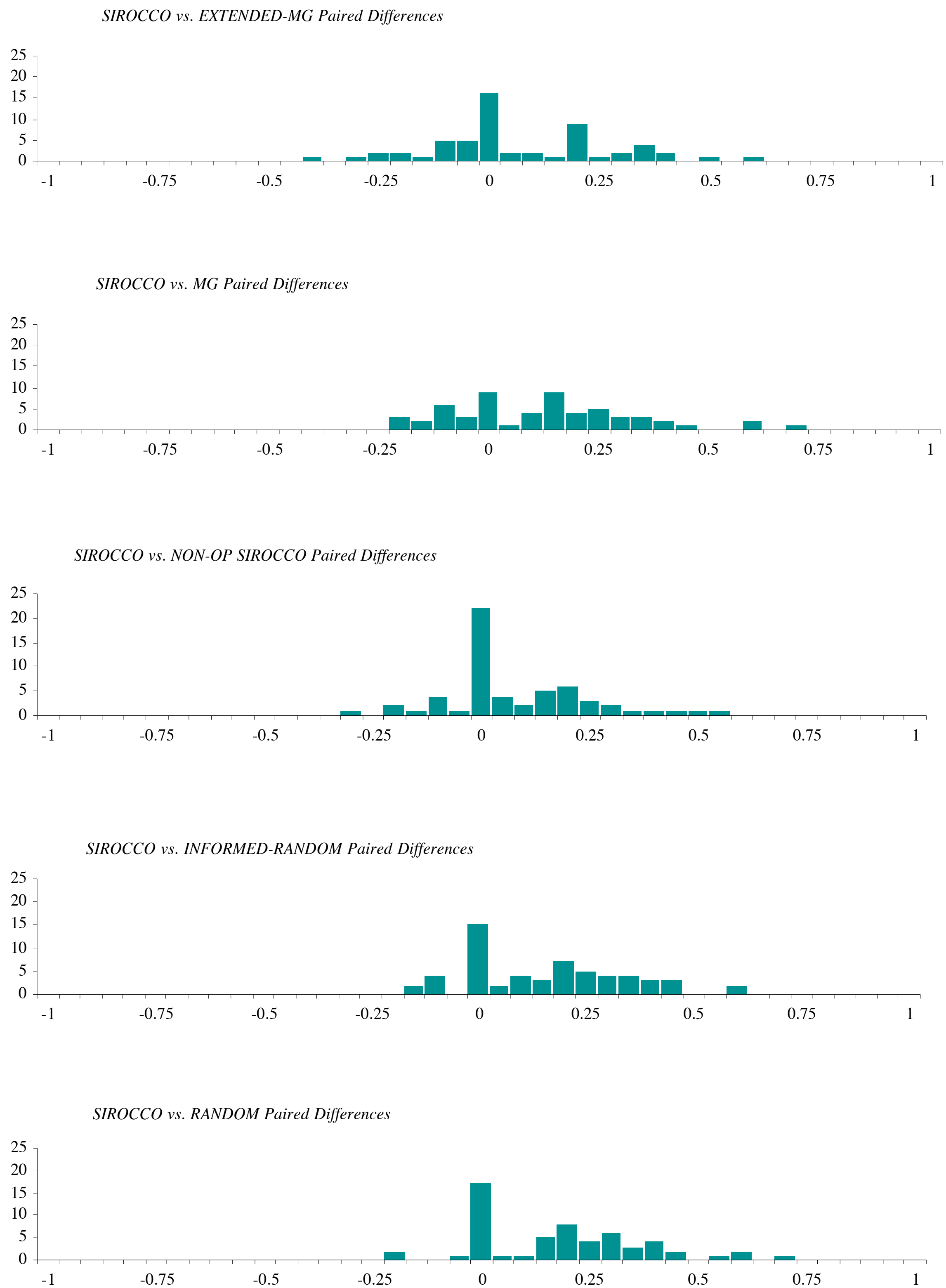


Figure 4-6: Experiment #1, Paired Differences of the F-Measures for Exact Citations

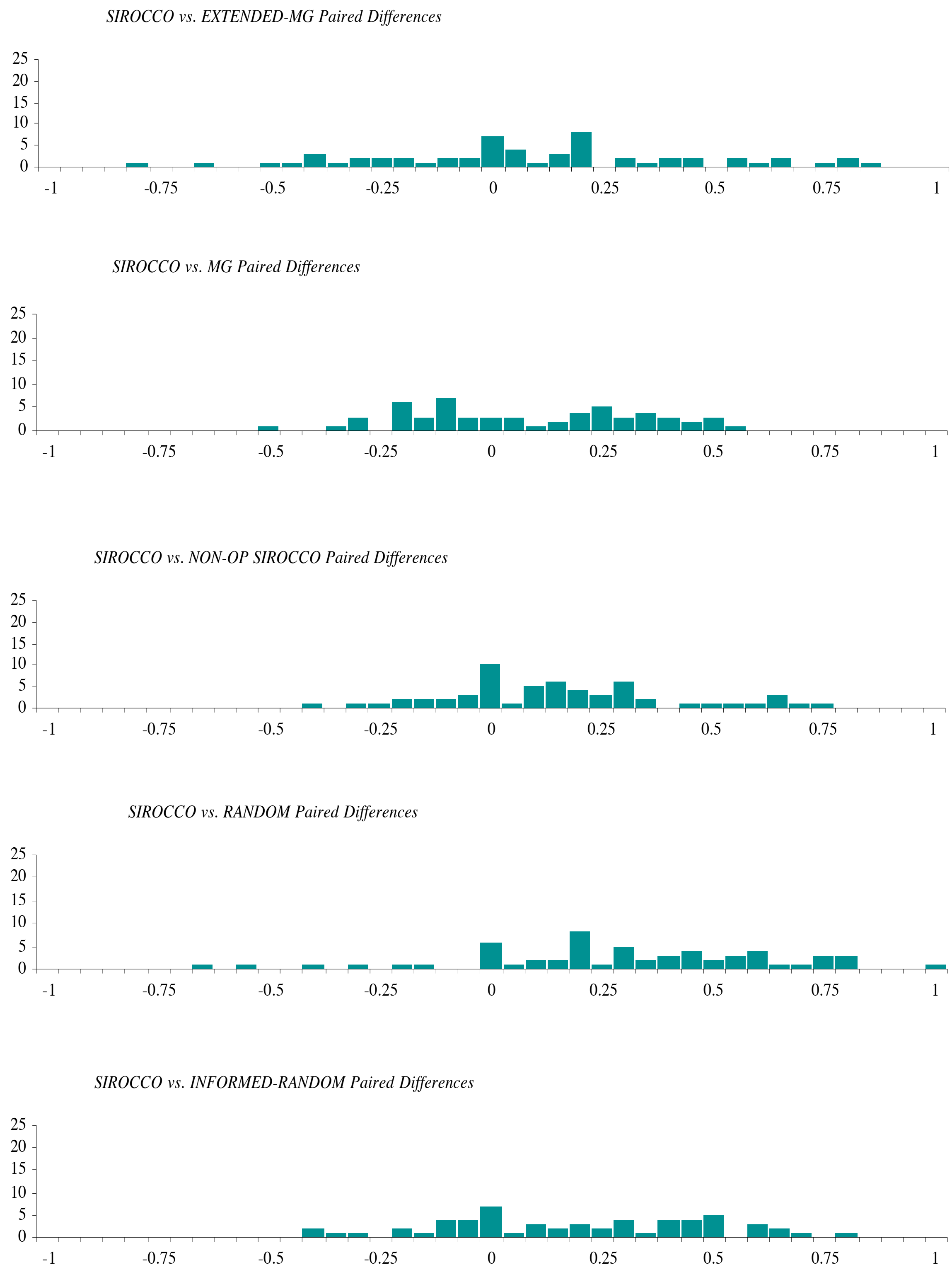


Figure 4-7: Experiment #1, Paired Differences of the F-Measures for Inexact Citations

Additional Results Related to Experiment #1

As a follow-up to Experiment #1, and as a way to informally verify the findings, a test was run in which all of the methods were given the 184 foundational cases, instead of the trial cases, as input. In this test, only the foundational cases (minus the target case and any sibling cases) were made available to each method as a database from which to identify relevant cases and codes. As in Experiment #1, F-Measures were calculated and the nonparametric bootstrap was applied to pairwise comparisons of SIROCCO to the other methods. The results mostly confirmed the statistical results of Experiment #1. In particular, SIROCCO performed significantly better than MG, NON-OP SIROCCO, INFORMED-RANDOM, and RANDOM on both exact and inexact matching, as in Experiment #1. However, it was not confirmed that SIROCCO was superior (i.e., p-value <= 0.05) to EXTENDED-MG with respect to the foundational cases for *either* exact or inexact matching, even though SIROCCO produced a better mean F-Measure for both data sets.

While this finding certainly does not support the claim that SIROCCO outperforms EXTENDED-MG, it is consistent with it. Experiment #1 is more of an objective indicator of the comparison between the methods, since cases were randomly selected in that experiment. On the other hand, the results of this follow-up test were still somewhat surprising and not obviously explainable. One possible explanation as to why EXTENDED-MG closed the gap in performance between itself and SIROCCO on the foundational cases is that many of these cases, which I hand-picked, may have certain linguistic characteristics that both encouraged me to select them and were somehow advantageous to EXTENDED-MG's algorithm.

Finally, the results of both Experiment #1 on the trial cases and the follow-up test with the foundational cases were informally evaluated with respect to performance on the Selected Topics cases and the Non-Selected Topics cases (see Table 1-1). The results are shown in Figure 4-8. The upper bar chart shows the mean exact and inexact F-Measures obtained for SIROCCO processing the trial cases. The statistics are further subdivided in this chart by the 44 Selected Topics cases versus the 14 Non-Selected Topics cases in the trial set. The lower chart displays analogous data that relates to SIROCCO's processing of the foundational cases (i.e., 135 Selected Topics cases and 49 Non-Selected Topics cases).

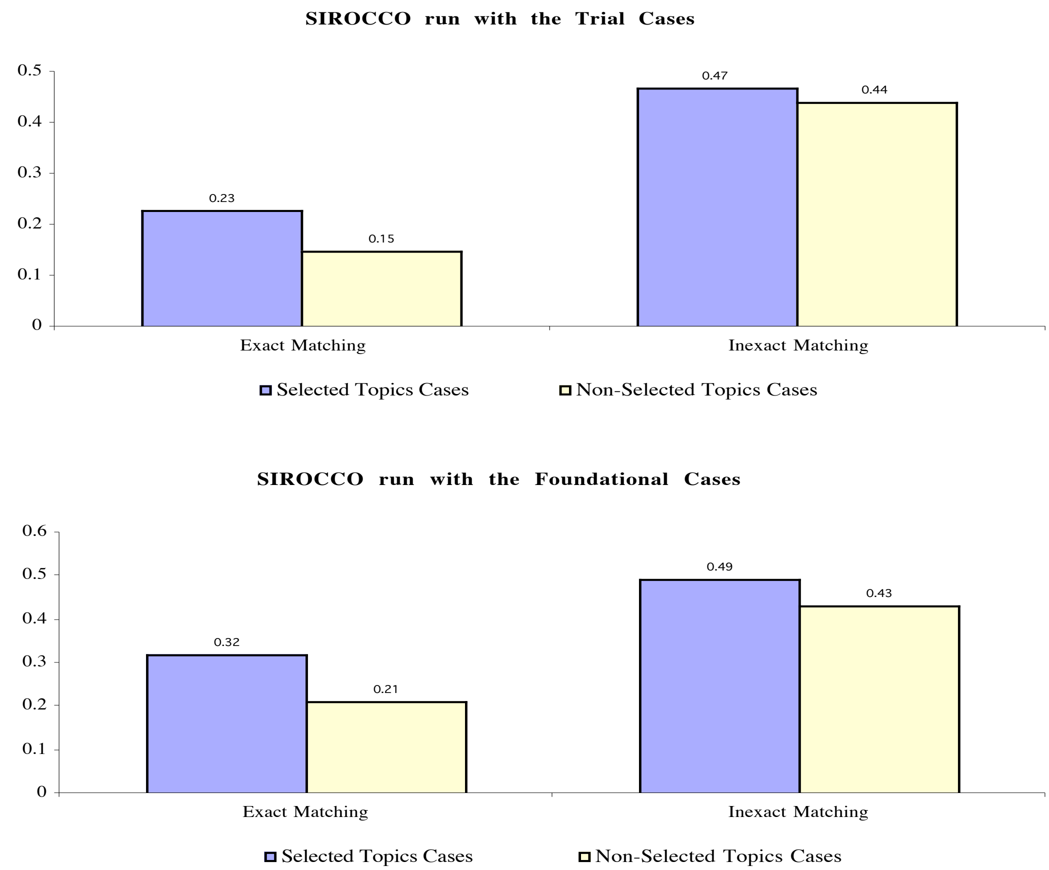


Figure 4-8: Mean F-Measures for SIROCCO Processing the Selected Topics Cases versus the Non-Selected Topics Cases

Because SIROCCO was developed using the Selected Topics cases as the primary focus, it is not surprising that the program is more accurate in processing cases in this category in both the trial and foundational sets and for both exact and inexact matching. However, the margin of improvement in a couple of the comparisons is certainly not overwhelming. For example, notice that the differences in SIROCCO's mean F-Measures in processing Selected Topics cases with respect to inexact matching, as compared to its mean F-measures in processing Non-Selected Topics cases, is relatively small for both the trial and foundational sets.

Of course, such comparisons are clearly informal, since there are far fewer Non-Selected Topics cases in both the trial and foundational sets. Nevertheless, the charts in Figure 4-8 provide at least some evidence that SIROCCO is capable of addressing a relatively wide range of engineering ethics cases. In particular, the charts provide evidence that the program does almost as well with cases outside of its primary area of expertise as it does with cases within that area of expertise.

4.2.2. Experiment #2: Expert Evaluation of Extra Citations

Experiment #2 Process

In Section 4.1.3 it was argued that the NSPE BER's case analyses provide the most-objective available benchmark for the SIROCCO experiments. On the other hand, using the review board's analyses as a gold standard is not without detractions. One could claim, for instance, that the NSPE BER's opinions do not always cite *all* of the relevant codes and cases. For instance, the board may not have had the time to collect all of the relevant codes and cases to support a particular argument or they may have believed they had enough supporting material. But the fact that certain codes and cases are not cited does not necessarily imply these materials are not relevant. Thus, in this experiment, the board's analyses were supplemented by the opinions of experienced ethical reasoners to help correct for this suspected incompleteness and to provide, arguably, a more precise comparison of SIROCCO with its nearest competitor in the first experiment, EXTENDED-MG.

The experienced ethical reasoners enlisted for the experiment were two graduate students with extensive backgrounds in ethics. One was a Ph.D. student in the philosophy of mathematics who has taught ethics courses, and the other was a Ph.D. student in bioethics.

Because the results of Experiment #1 indicated, quite decisively, that SIROCCO is superior to MG, NON-OP SIROCCO, INFORMED-RANDOM, and RANDOM in retrieval accuracy, all of these competitor methods were not evaluated in this experiment. On the other hand, because the comparisons between SIROCCO and EXTENDED-MG were less conclusive, EXTENDED-MG was included with SIROCCO in this experiment.

The data generated in Experiment #1 was used as the basis for the data evaluated in Experiment #2. Specifically, the code and case suggestions of SIROCCO and EXTENDED-MG for the 58 trial cases that did not match, either exactly or inexactly, with the board's citations for the same cases were collected and organized by case (called a *case group*). Up to two of the review board's code citations and up to two of their case citations were randomly added to each case group. This assured that the evaluators would not know which citations were made by the programs and which were made by the review board.

The evaluators were presented, on a web page, with the original case texts and questions for all of the 58 trial cases, together with the corresponding case groups of cited codes and cases. They were instructed to read each case and to evaluate whether each of the codes and cases in the corresponding case group could be considered reasonably relevant to the facts and could be used to support an argument answering the question raised in the scenario. The evaluator's responses were provided in an answer form also included on the web site.

More specifically, the evaluators were instructed to fill in the answer form as follows:

> "Provide an answer of 'R' for the citation if you believe it is reasonably relevant to the scenario. By reasonably relevant, I mean, would it be reasonable for an experienced ethical reasoner to reference the cited item in an argument answering the question raised by the case… Provide an answer of 'N' for the citation if it is not reasonably relevant to the scenario in the sense defined above."

Each evaluator was asked first to assess all of the citations on his/her own. After each had completed the initial evaluation, they were instructed to meet, discuss the differences in their respective answers, and to arrive, if possible, at a consensus for each differing citation. In other words, for every answer about which the evaluators individually disagreed (i.e., one evaluator provided an 'R' while the other provided an 'N'), they were told to either agree on either 'N' or 'R' for that response or to indicate with an 'NA' if they could not agree.

Three answer forms were collected from this process, one from each evaluator individually and one from the evaluators working together. The individual forms were used to calculate inter-rater reliability. The consensus form was used to recalculate the F-Measures for SIROCCO and EXTENDED-MG over the 58 trial cases. Specifically, for each code and case citation that was not originally cited by the review board, but for which the evaluators, by consensus, assigned an 'R,' that citation was added to the list of the review board citations for purposes of the F-Measure recalculation. This implies, of course, that SIROCCO and EXTENDED-MG would receive exact (and inexact) match credit for suggesting one of the citations rated as 'R' by the evaluators. Each 'NA' response was conservatively treated as an 'N' response for the purposes of the F-Measure recalculations, since its relevance was obviously controversial.

Quantitative Results

To confirm inter-rater reliability, the individual responses of the two evaluators were compared. They agreed on 77.4% (369 out of 477) of the responses. This percentage of agreement is typically considered fair-to-good[39].

A total of 75 of the additional code citations and 15 of the additional case citations were rated as 'R' by the evaluators in the consensus answer form. Of these additional citations, SIROCCO

---

[39] Here, "typically" refers to the use of inter-rater reliability in other academic disciplines, such as psychology. The use of this metric in case-based reasoning – and in AI more generally – is relatively rare.

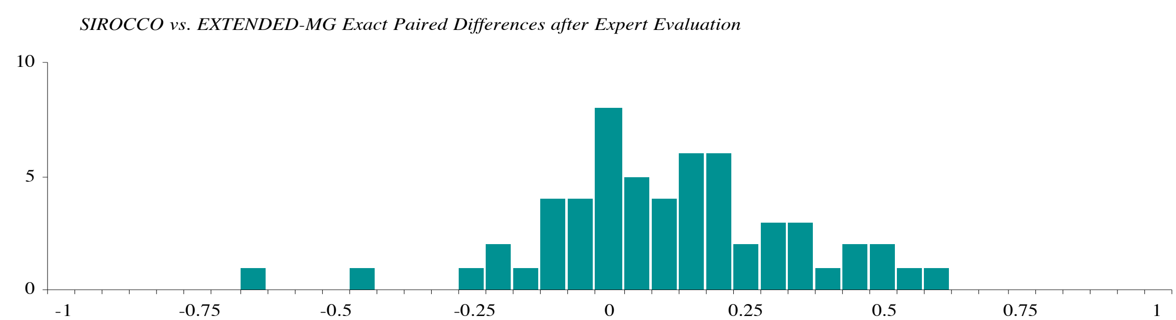
cited 63 of the codes and 7 of the cases, while EXTENDED-MG cited 34 of the codes and 9 of the cases.

For SIROCCO, the recalculated mean F-Measures were 0.359 for exact matching and 0.581 for inexact matching. For EXTENDED-MG, the recalculated mean F-Measures were 0.251 for exact matching and 0.456 for inexact matching.

Table 4-3: Experiment #2, Comparison of SIROCCO and EXTENDED-MG using Nonparametric Bootstrap after Expert Evaluation

Methods Compared	95% Confidence Interval for mean difference	p-value for mean difference=0
SIROCCO vs. EXTENDED-MG, Exact Matching	(0.048, 0.168)	< 0.001
SIROCCO vs. EXTENDED-MG, Inexact Matching	(0.030, 0.220)	0.010

Table 4-3 displays the results of the nonparametric bootstrap procedure applied to the recalculated F-Measures. Notice that there is now a significant difference between the accuracy of SIROCCO and EXTENDED-MG on both the exact and inexact match criteria. For both criteria, the confidence level of a difference (in favor of SIROCCO) is now at least 99%.



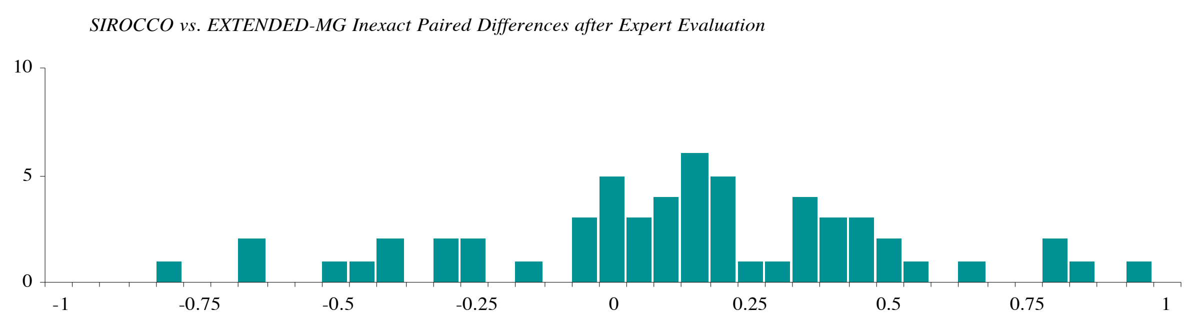


Figure 4-9: Experiment #2, Paired Differences of the F-Measures for Exact and Inexact Citations

The difference between SIROCCO's and EXTENDED-MG's performance after recalculating the F-Measures can also be easily visualized in the paired differences charts of Figure 4-9. Notice that a clear concentration of bars is on the right side of the 0 hash of both charts, indicating that SIROCCO is significantly better than EXTENDED-MG based on the recalculated measures.

4.2.3. Discussion

The combined results of Experiment #1 and Experiment #2 provide compelling evidence that SIROCCO is superior to EXTENDED-MG, MG, NON-OP SIROCCO, INFORMED-RANDOM, and RANDOM in accurately retrieving codes and cases. More importantly, these experiments provide very strong evidence in support of the primary thesis of this dissertation. In particular, the evidence suggests that a computational model, such as SIROCCO, that implements a critical subset of the operationalization techniques (i.e., *Code Instantiation*, *Group Codes*, *Case Instantiation*, *Group Cases*, and *Reuse an Operationalization*) can make accurate predictions of the principles and past cases that are likely to be important in the analysis of new cases.

Experiment #1 provides convincing statistical data to support the claim that SIROCCO outperforms MG, NON-OP SIROCCO, INFORMED-RANDOM, and RANDOM. However, SIROCCO's superiority over EXTENDED-MG is not decisively demonstrated in the first experiment, as SIROCCO's improvement on the exact matching criteria is significant, but its improvement on the inexact matching criteria is only marginally significant. However, the second experiment, in which the additional citations of SIROCCO and EXTENDED-MG were evaluated, does demonstrate, quite clearly, the superiority of SIROCCO. On both the exact and inexact matching criteria in the second experiment, SIROCCO performs significantly better than EXTENDED-MG. Outperforming EXTENDED-MG – as well as MG – strongly indicates that SIROCCO is a powerful retrieval method, as the full-text retrieval methods are arguably highly competitive alternatives for performing SIROCCO's task.

SIROCCO's significant improvement over NON-OP-SIROCCO goes right to the heart of the primary thesis, as it provides direct evidence of the benefit of using operationalization techniques (or, more precisely, a subset of the operationalization techniques) compared to not using the techniques. NON-OP SIROCCO actually depends on a weak version of one of the techniques (i.e., *Reuse an Operationalization*), so the difference between the two methods would likely have been even more decisive had the experiment compared SIROCCO to an ablated version of itself completely devoid of the operationalization techniques.

The fact that SIROCCO outperforms EXTENDED-MG which, in turn, tentatively outperforms MG[40] in Experiment #1 is also significant with respect to the primary thesis . Notice that the difference between MG and EXTENDED-MG involves the use of an operationalization technique in the latter method. That is, while MG performs its task using a pure full-text retrieval approach, EXTENDED-MG uses full-text retrieval augmented with *Reuse an Operationalization* (i.e., cases are retrieved by full-text retrieval, codes are selected based on frequency of occurrence in the retrieved cases). This again demonstrates the value of the operationalization techniques. While SIROCCO clearly outperforms the full-text retrieval system, it less clearly outperforms that same system when it employs an operationalization technique. However, the preponderance of operationalization techniques used by SIROCCO eventually proved, in Experiment #2, to perform more accurately than even the extended, operationalized version of the full-text retrieval system.

An Example of SIROCCO Providing a More Accurate and Complete Retrieval Than Either the NSPE BER or EXTENDED-MG

For some of the trial cases, SIROCCO's performance was more accurate and complete than either the review board or EXTENDED-MG due to its unique retrieval approach and case base. Such results are interesting because they indicate that SIROCCO has the capability to at least occasionally outperform even the board in retrieving relevant materials.

For instance, consider trial Case 96-8-1 shown in Figure 4-10. This case involves Engineer A who agrees to serve as a "peer reviewer" as part of an organized program designed to improve the "professional practice" of fellow engineers. As part of the program, Engineer A signs a confidentiality agreement that stipulates that she may not disclose the confidential information of peer-reviewed engineers and firms. Engineer A subsequently reviews the work of Engineer B's firm and discovers that some of the firm's work may violate safety codes and could place public safety in jeopardy. Engineer A attempts to resolve this issue by speaking with Engineer B but when no resolution is attained, she reports the potential violation to the appropriate authorities. The question raised by this case is whether it was ethical for Engineer A to violate the confidentiality agreement and report the potential safety violation.

---

[40] Results of the nonparametric bootstrap comparison of EXTENDED-MG and MG : On the exact matching criteria, EXTENDED-MG performed significantly better than MG. On the inexact matching criteria, the results were inconclusive.

Facts of Case 96-8-1: Peer Review - Confidentiality Agreements

Engineer A serves as a peer reviewer as part of an organized peer review program developed to assist engineers in improving their professional practice. When originally selected as a peer reviewer, Engineer A was asked to sign a "confidentiality agreement" whereby she agreed not to disclose confidential information involving peer-reviewed firms.

As part of a peer review visit, Engineer A visits Engineer B's firm. Following a review of the technical documentation in connection with a series of recent design projects involving Engineer B's firm, Engineer A discovers that Engineer B's work may be in violation of state and local safety code requirements and could endanger public health, safety, and welfare. Engineer A immediately discusses these issues with Engineer B in an effort to seek clarification and early resolution. Engineer A and Engineer B are unable to resolve the issue, and Engineer A subsequently informs the appropriate authorities about Engineer B's potential violation of the safety code.

Question:

Was it ethical for Engineer A to inform the authorities about Engineer B's potential violation of the safety code?

References:

Code II.1.e. – "Engineers having knowledge of any alleged violation of this Code shall report thereon to appropriate professional bodies and, when relevant, also to public authorities, and cooperate with the proper authorities in furnishing such information or assistance as may be required."

Code III.4. – "Engineers shall not disclose, without consent, confidential information concerning the business affairs or technical processes of any present or former client or employer, or public body on which they serve."

Figure 4-10: The Fact Situation, Question Raised, and Relevant Codes (according to the NSPE BER) in Case 96-8-1

According to the NSPE BER's cited references (see the bottom of Figure 4-10) and analysis (not shown), this case involves a conflict between Engineer A's obligation to report potential code violations (i.e., Code II.1.e.) and her obligation to maintain confidentiality (Code III.4.). The board concluded that Engineer A's obligation to report the possible code violations predominated in this circumstance and thus Engineer A's action was ethical. Although the facts clearly indicate that "public health, safety, and welfare" might have been at risk – and the board referred to this issue in their analysis, even citing public safety Case 76-4-1 (One of the example cases from Chapter 3, see Figure 3-3) – notice that they did not cite any codes related to the protection of the public.

During the Experiment #1 trial run of Case 96-8-1, SIROCCO was able to identify the relevance of Code III.4. (i.e., the confidentiality code) and also the relevance of Case 76-4-1. The program did not however cite Code II.1.e. (i.e., the code specifying an obligation to report code violations) as possibly relevant. On the other hand, SIROCCO was able to identify six codes that the evaluators in Experiment #2 deemed to be relevant but yet were not cited by the board. For instance, SIROCCO cited Codes I.1., II.1.a., and III.2.b., all of which relate to the issue of public health and welfare. As mentioned above, the board failed to cite any of the public safety codes, even though the facts indicated that they were relevant to the case. SIROCCO also cited an additional code dealing with confidentiality (i.e., Code II.1.c.) that was not cited by the board. Perhaps most interestingly, SIROCCO cited Code III.1. ("Engineers shall be guided in all their professional relations by the highest standards of integrity.") as relevant, while the board did not. The evaluators found this code to be relevant and one commented as follows: "Engineer A had the duty to report possible violations of the code, hence according to III.1 his actions were justified." In summary, SIROCCO cited two of the three codes and cases cited by the NSPE BER for Case 96-8-1, but the program also cited a significant number of codes that were deemed relevant by the evaluators yet were not cited by the board.

SIROCCO also provided a more accurate and comprehensive retrieval than EXTENDED-MG did for Case 96-8-1, according to the evaluators. While EXTENDED-MG managed to identify both of the board's cited codes (see Figure 4-10), it did not cite Case 76-4-1 nor did it cite two of the codes suggested by SIROCCO and sanctioned by the evaluators, i.e., Codes III.1. and III.2.b. In short, while EXTENDED-MG did a reasonably good job at identifying the codes relevant to Case 96-8-1, it missed an important case citation and was not as comprehensive as SIROCCO was in citing relevant codes.

4.3. Testing the Secondary Thesis

A single experiment was run to test the secondary thesis of the dissertation. In this experiment, SIROCCO's accuracy over the trial cases is compared to an ablated version of the program, called NON-TEMP SIROCCO, in which the temporal component of the program was turned off. More specifically, NON-TEMP SIROCCO represents a version of SIROCCO in which condition 4 of a valid structural mapping (see the discussion of Figure 3-22) is not enforced. In other words, in NON-TEMP SIROCCO structural mapping is based only on a consistent mapping of Actors, Objects, Fact-Phrases, and Fact-Primitives but not on temporal relations, as in SIROCCO.

The goal in comparing these two methods was to determine whether SIROCCO performs significantly better, in a statistical sense, than its counterpart lacking the temporal knowledge. Such a result would strongly support the hypothesis that SIROCCO's temporal knowledge assists it in making accurate predictions of the principles and past cases that are relevant to the analysis of new cases.

4.3.1. Experiment #3: SIROCCO with and without Temporal Reasoning

Experiment #3 Process

In Experiment #3, NON-TEMP SIROCCO, the ablated version of SIROCCO that does not apply temporal knowledge, was run in similar fashion as were SIROCCO and the five competitor methods in Experiment #1 (described in Section 4.2.1). That is, NON-TEMP SIROCCO was run against each of the 58 trial cases with the date filter turned on. As before, for each run of a single trial case, the program had access to the 184 foundational cases, plus the 58 trial cases, minus the target case and each of its sibling cases, for purposes of selecting citations. The only difference in this test run was that the *Temporal-Match-P* flag was turned off, meaning that NON-OP SIROCCO did not test mapping condition 4, discussed in Section 3.1.4. All other parameter settings of NON-TEMP SIROCCO in Experiment #3 were identical to those applied to SIROCCO in Experiment #1.

For each trial case, the exact and inexact F-Measures were calculated by comparing NON-TEMP SIROCCO's output to the citations made by the review board for corresponding cases. The mean exact and inexact F-Measure over all trial cases were calculated, and the nonparameteric bootstrap procedure was applied to these results, comparing them to the results of SIROCCO from Experiment #1.

Quantitative Results

The results of running NON-TEMP SIROCCO on all of the trial cases were as follows. The mean F-Measure for exact matching was 0.213 and the mean F-Measure for inexact matching was 0.461. Recall that the mean F-Measures for SIROCCO in Experiment #1 were 0.212 for exact matching and 0.462 for inexact matching.

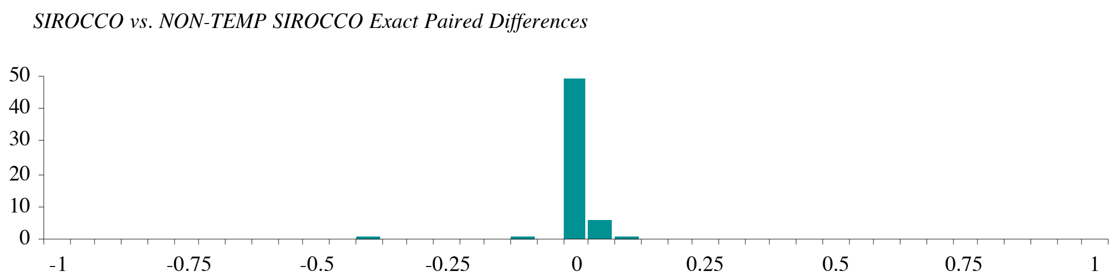
As shown in Table 4-4, the results of the nonparametric bootstrap indicate that SIROCCO was *not* significantly more accurate than NON-TEMP SIROCCO with respect to the trial cases.

Table 4-4: Experiment #3, Comparison of SIROCCO and NON-TEMP SIROCCO using Nonparametric Bootstrap

Methods Compared	95% Confidence Interval for mean difference	p-value for mean difference=0
SIROCCO vs. NON-TEMP SIROCCO, Exact Matching	(-0.017, 0.011)	0.909
SIROCCO vs. NON-TEMP SIROCCO, Inexact Matching	(-0.021, 0.025)	0.926

This result can be visualized by inspecting the paired differences charts of Figure 4-11. For both the exact and inexact matching criteria, a very high percentage of the F-Measures were identical, as witnessed by the large bar at the 0 hash mark of both charts. Only a small percentage of the cases led to different F-Measures and most of these are found close to the 0 hash mark, meaning that the absolute differences were slight.

A similar, informal test was also run with the foundational cases. In this test, the mean F-Measures for exact matching were 0.292 for NON-TEMP SIROCCO and 0.285 for SIROCCO; the mean F-Measures for inexact matching were 0.473 for NON-TEMP SIROCCO and 0.479 for SIROCCO. Although the p-value for a mean difference = 0 for exact matching was slightly more favorable on this data set (i.e., 0.148), it was still far less than significant. In addition, the p-value for inexact matching was not significant, as shown by the p-values in the table. Thus, this test tended to confirm the results of Experiment #3 – that SIROCCO's temporal knowledge did not improve retrieval accuracy.



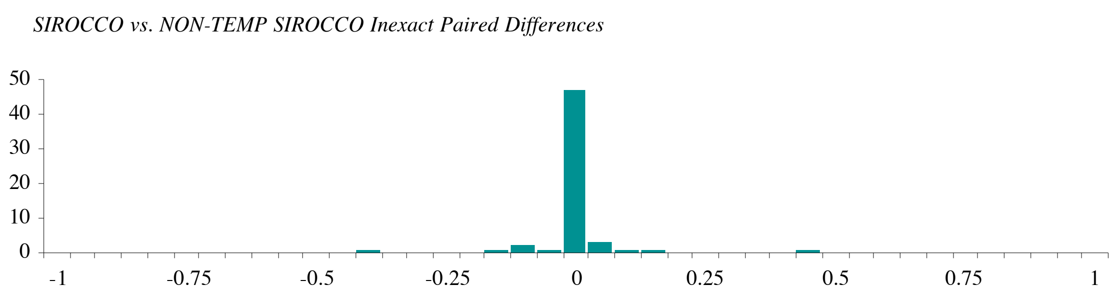


Figure 4-11: Experiment #3, Paired Differences of the F-Measures for Exact and Inexact Citations

4.3.2. Discussion

It is clear from the results of Experiment #3, as well as from the informal follow-up test with the foundational cases, that SIROCCO's application of temporal knowledge did not make a difference in the overall accuracy of code and case retrieval. In other words, the secondary thesis of this dissertation is not supported by the results of this experiment.

Why didn't SIROCCO's temporal knowledge make a statistical difference? As discussed and illustrated in Section 3.3 there are clearly specific instances in which SIROCCO's temporal knowledge can be used to identify important distinctions between cases and thereby lead to the correct selection of relevant codes and cases. However, Experiment #3 shows that while specific instances may benefit from the application of temporal knowledge, accuracy in general is not improved by SIROCCO's current implementation and use of temporal knowledge.

One possible reason for this is evident in the charts found in Figure 4-11. As discussed above, a significant majority of the trial cases exhibited no difference in the F-Measures attained by SIROCCO versus those attained by NON-TEMP SIROCCO. This suggests that a large number of the trial cases simply did not involve temporal considerations, or at least did not involve them in a critical fashion. A cursory reading of a subset (10) of the trial cases indicated that this may be generally true. While most of the 10 cases involved temporal events, only a couple appeared to turn specifically on event sequence. In most of the cases, the mere existence of certain facts was more critical than the order of those facts.

However, such an explanation cannot fully explain why temporal knowledge also did not play a significant role in the test executed with the foundational cases. As discussed in Section 2.2.3, the analysis of the NSPE BER cases seemed to indicate that temporal knowledge *did* play a role in the decision making of the review board and many of those cases were included in the foundational case base. On the other hand, even if a fair number of the foundational cases did rely on temporal considerations, this number may not have been significant enough when compared to the cases that did not rely on temporal considerations.

Other explanations are also possible. It might be relatively rare that pairs of cases exist such that (1) a difference in temporal ordering leads to two different ethical interpretations and (2) both cases are found in the NSPE BER case base. For instance, it certainly only makes sense that one would report a dangerous situation after they learn of it, not before. In other words, SIROCCO's opportunities to identify important differences in cases, based on temporal differences, may be rare. Thus, it may be the case that the only way to truly test SIROCCO's capability to discern differences in cases based on temporal considerations is to modify hypothetically cases such that the time ordering changes and a case that once involved an ethical obligation no longer does. Running SIROCCO on a test set of such cases could test whether the program successfully distinguishes among the meaningful and unmeaningful scenarios.

On the other hand, some facts are indicative of the relevance of specific codes regardless of temporal considerations and would thus be unaffected by any differences in temporal ordering. For instance, consider example case 90-5-1, discussed in Chapters 1, 2, and 3. The fact that the safety of the tenants in the apartment building was at risk is indicative of the relevance of a public safety code. While deciding whether such a code is violated may depend on the sequence of temporal events, the relevance of the code probably requires less specific knowledge of the order of events.

Another possible explanation for the results of Experiment #3 is that certain Facts typically occur in approximately the same temporal position in Fact Chronologies, thus producing less variability and perhaps less temporal distinction between the chronologies. For instance, primitives dealing with employment, such as "hires the services of" or "is employed by," tend to be at the beginning of Fact Chronologies and, in addition, are quite often assigned the Time Qualifier "Pre-existing fact." Such a Fact designated as a Critical Fact of a *Code Instantiation* or *Case Instantiation* is less likely to produce a differentiating effect in the structural mapping process.

Inaccuracies in the assignment of Time Qualifiers by case enterers is another possible explanation. As discussed in Section 3.1.2, the task of assigning temporal relations between Facts is not easy for people and although the Time Qualifiers – disjunctive groups of Allen's temporal relations – are intended to somewhat ease this difficulty, the task is still hard and prone to error. In fact, a substantial portion of the post editing of the foundational cases provided by independent case enterers involved the correction of Time Qualifiers.

Finally, the particular implementation and use of temporal knowledge within SIROCCO may be a contributing factor. First, the underlying use of disjunctive groups of Allen's temporal relations between Facts tends to lead to a somewhat "forgiving" approach in accepting structural mappings. Because SIROCCO's algorithm performs an intersection of the temporal relations between pairs of Facts, there may be situations in which only a single, common possible relation allows a mapping to succeed. While this is strictly correct, it may lead to the acceptance of unlikely mappings. Second, temporal knowledge may simply not have a big enough role in SIROCCO's structural mapping scoring algorithm and selection heuristics. For example, a critical subsequence of steps in a *Code Instantiation* may be rejected according to SIROCCO's algorithm, but the *Code Instantiation* itself may still succeed if other Facts match well enough for the Instantiation to receive a relatively high score. Consider the example discussed in Section 3.3. It showed how the enforcement of temporal relations led to a structural mapping of only two Facts, instead of three. Although this sufficiently reduced the match core so that SIROCCO correctly eliminated the associated code in this particular example, it is possible in other situations that the code might still achieve a high enough score to be considered relevant. Also, recall that the selection of codes and cases by SIROCCO's Analyzer is based on an *accumulation* of evidence supporting individual codes and cases and not on the success of individual mappings. This implies that the failure (or success) of individual mappings may not always be enough to alter the suggestions made by the program. A final implementation issue is that all temporal relations are treated equally when, in fact, certain temporal relations might be critical, while others are not. If the case enterer had a means of designating critical temporal relations, and SIROCCO's algorithm was sensitive to these, this might make a difference in SIROCCO's accuracy.

Most likely, some combination of the above reasons underlies SIROCCO's failure to gainfully use temporal knowledge in the selection of codes and cases. Although the results of this experiment do not support the secondary thesis of the dissertation, it is possible that addressing some of the case-acquisition and implementation issues discussed above might make a difference. This difference might occur if a significant number of the cases do, in fact, rely on temporal considerations, as was conjectured from the analysis of the NSPE BER cases. An investigation of these issues remains for future work (see Section 6.3.1).

4.4. Testing Accuracy, Efficiency, and Scalability

Ultimately, SIROCCO is intended to be a practical tool to be used by engineers and engineering students. Thus, accuracy must be balanced with computational efficiency. In addition, it is important to understand the impact of an ever-increasing case base size on SIROCCO's computational efficiency. In practical use, one would expect SIROCCO's case base to increase beyond its current size, particularly if it were to cover the Non-Selected Topics as well as the Selected Topics. This section presents empirical results that explore these issues.

All of these experiments were run on a 400 Megahertz Power Macintosh with 128 megabytes of RAM.

4.4.1. Experiment #4: Accuracy versus Efficiency as Case Base Size Increases

Experiment #4 Process

Experiment #4 was executed to evaluate the impact on SIROCCO's accuracy, as measured by the F-Measure, as the case base gradually increased from a minimal size to its current maximum size. The experiment involved 20 separate test runs in which SIROCCO processed all 58 trial cases. The default SIROCCO parameter settings, as applied in earlier experiments, were also applied in this experiment. With each test run SIROCCO was provided access to a case base 10 cases larger than the prior test. In the initial test, SIROCCO had access to a case base of 62 cases (i.e., the 58 trial cases, plus the 4 unused trial cases). For each subsequent test, 10 cases were chosen at random from the foundational cases and added to the case base (except for the final test, in which only 4 cases remained to be added), up to a total of 246 cases. As in the earlier experiments, the exact and inexact F-Measures were calculated per case and as a mean over all 58 cases per test run. In addition, the overall run time for all 58 cases per test run was captured.

Quantitative Results

Figure 4-12 depicts the results of Experiment #4. As the case base size increases, mean inexact matching gradually grows, from an F-Measure of 0.315 at case base size 62 to an F-Measure of 0.462 at case base size 246. Although the nonparametric bootstrap was not applied to this data, note that such a difference in F-Measure was enough to grant statistical significance to comparisons in the earlier experiments.

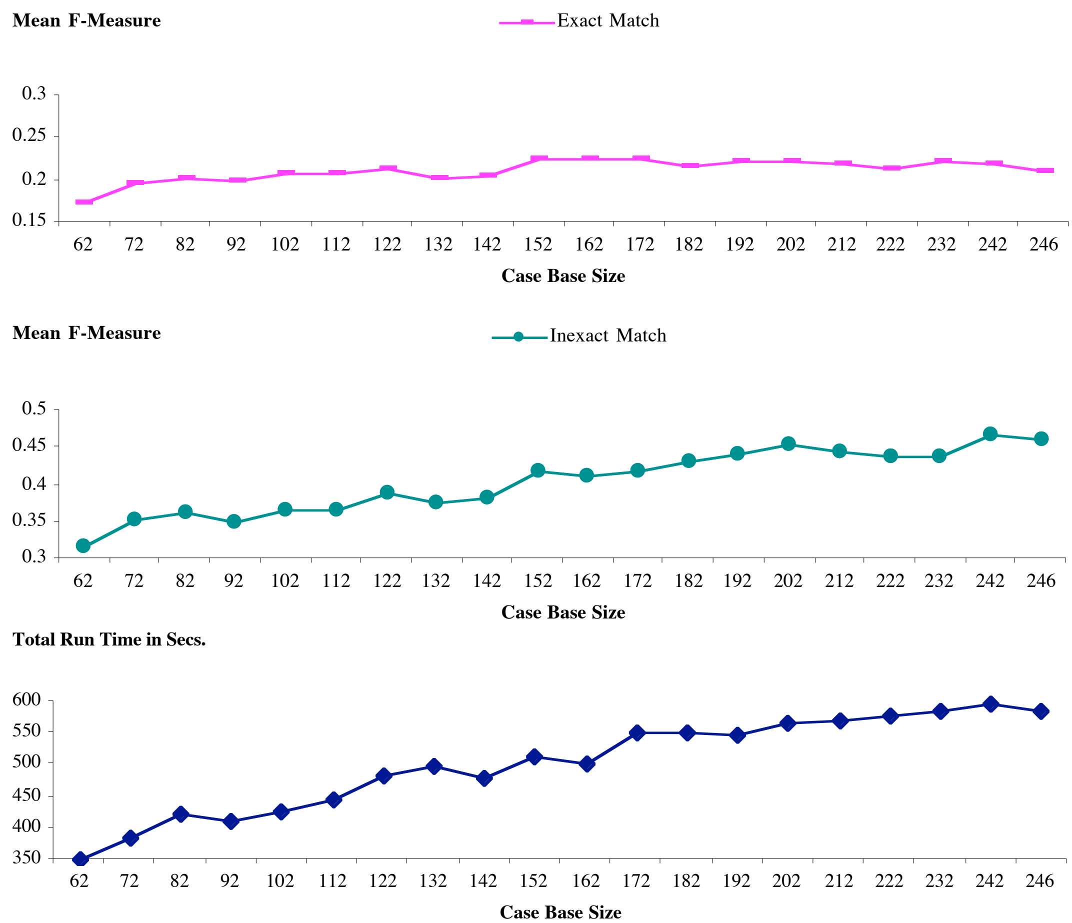


Figure 4-12: Experiment #4, SIROCCO's Accuracy Compared to Its Efficiency as Case Base Size Increases

As can be seen in the bottom two charts, the increase in slope of the inexact F-Measure curve is very similar to the increase in slope of the run-time curve (although the scales obviously differ). Interestingly, both curves do not monotonically increase, with increases and decreases at almost identical readings. For instance, notice the increase in both the inexact F-Measure curve and the run-time curve between a case base size of 142 and a case base size of 152. The nonmonotonicity and similarity between the curves may be explained by a correlation between more-complex cases, with respect to the cases' EETL representations, and improved accuracy. That is, cases with more Critical Facts linked to *Code* and *Case Instantiations* may lead to better solutions, but such cases also require more search time.

The mean exact F-Measure increases over the first couple of readings but then generally levels off for the remainder of the tests. This may be explained by the relative infrequency of exact matches together with the vagaries of random selection. That is, since in general exact matches do not occur frequently, the relatively few situations in which they do occur just happened to be supported by the initial random groups of source cases.

From these charts it can be concluded that accuracy, at least in terms of inexact matching, in general increases with case base size. However, the marginal increase in mean inexact F-Measure after a case base size of 202 and, in fact, the decrease in mean F-Measure for the last reading may indicate that the benefit of continued increase in the case base is close to leveling off. However, without testing a greater number and wider range of unique cases, which would require substantially more case acquisition, one cannot be definitive about such a claim.

The run-time characteristics of SIROCCO also appear to be quite reasonable. The bottom chart of Figure 4-12 suggests that SIROCCO's run time is no worse than linear, and may be as low as log *N,* with respect to case base size. (However, this can of course only be established by theoretically analyzing the asymptotic time complexity.) For a case base of size 246, each of the 58 trial cases ran for an average of 10.1 seconds. From a user's point of view, such a run time is very reasonable.

4.4.2. Experiment #5: Accuracy versus Efficiency as Search Increases

Experiment #5 Process

Experiment #5 was executed to evaluate the impact on SIROCCO's accuracy, as measured by the F-Measure, as *N* gradually increases. The goal was to determine the value of *N* that maximizes SIROCCO's accuracy without requiring excessive computation. Recall that *N* is the user-specified parameter that designates the number of top-rated cases that are passed from SIROCCO's Stage 1 retrieval algorithm to its Stage 2 algorithm. This is a critical parameter, since Stage 2 applies the relatively expensive structural mapping algorithm to all of the Instantiations of the cases supplied to it.

The experiment involved 15 separate test runs in which *N* was varied from 1 to 15 with all other parameter settings fixed, as in earlier experiments. For each test run, SIROCCO processed all 58 trial cases using a case base of size 246 (i.e., the full case base, including the 58 trial cases, the 184 foundational cases, and the 4 unused trial cases). Also for each test run, the exact and inexact F-Measures were calculated per case, the mean exact and inexact F-Measures over all 58 cases were computed, and the overall run time per test run was recorded.

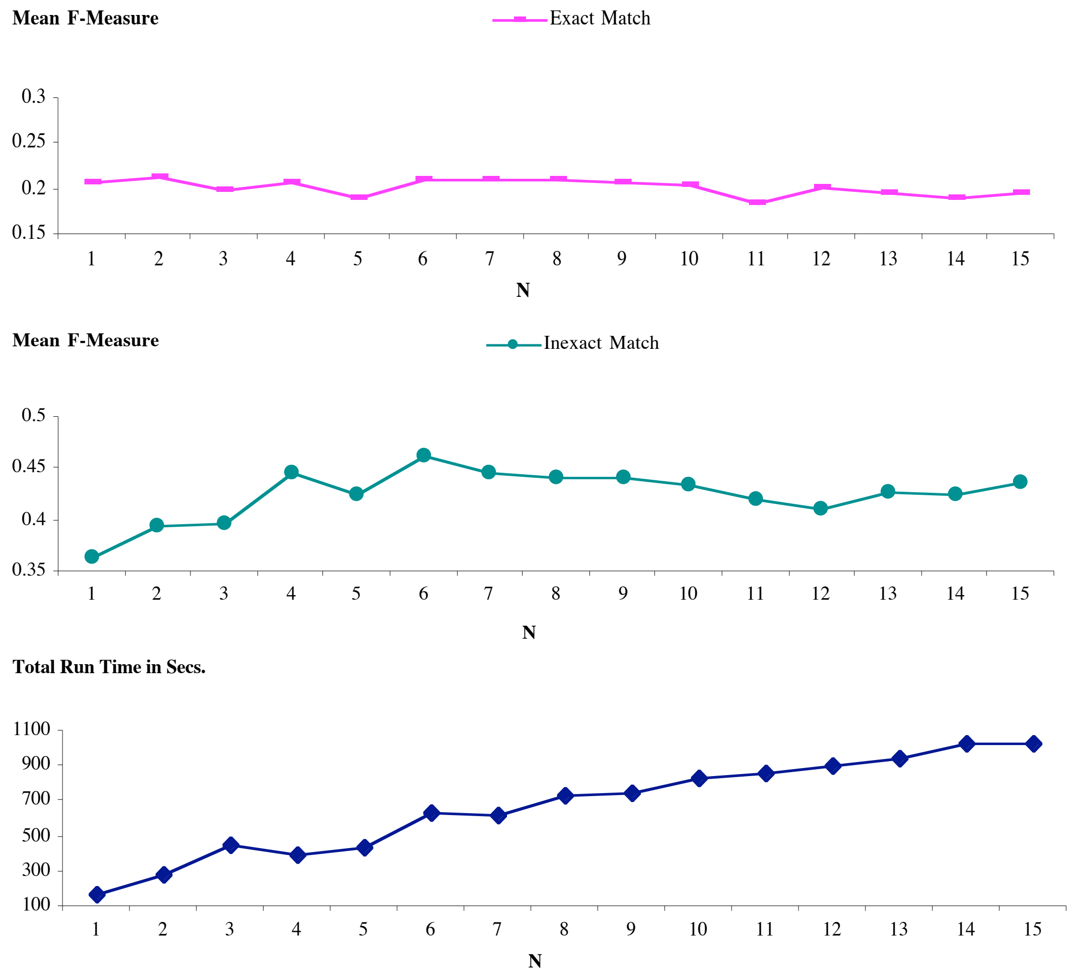


Figure 4-13: Experiment #5, SIROCCO's Accuracy Compared to its Efficiency as *N* Increases

Quantitative Results

The results of Experiment #5 are shown in Figure 4-13. The mean exact F-Measure reaches a peak at *N* = 2 (0.213) and then stays relatively flat for the rest of N, with a slight decrease over the last several readings. This indicates that when SIROCCO does find the best possible case – at least as judged by the review board – quite often that case is one of the top two cases according to Stage 1. Thus, by increasing N above 2, one does not improve the odds that SIROCCO will find the *exact* codes or cases cited by the review board.

The mean inexact F-Measure reaches a peak at *N* = 6 (0.462). The mean inexact F-Measure gradually (but not monotonically) increases from *N* = 1 to *N* = 6. After *N* = 6 the mean inexact F-

Measure gradually decreases. Also note that the mean exact F-Measure at $N = 6$ (0.212) is the second best of the 15 exact readings.

In summary, the results of Experiment #5 indicate that the best cases from which to select codes and cases are usually found within the top 6 cases selected by Stage 1. All of the $N > 6$ cases produce less-accurate results at an increasing run-time cost. Thus an N value of 6 is tentatively considered optimal for this domain. Note that this result and conclusion are informally confirmed by the experiment pre-tests, in which $N = 6$ was also found to be the optimal setting.

4.4.3. Experiment #6: Efficiency as Case Base Scales Up

Experiment #6 Process

Experiment #6 was executed to evaluate SIROCCO's capability to scale up. Because Stage 1's dot product potentially retrieves all cases in the case base, SIROCCO's run time is a function of case base size. The goal of this experiment was to determine whether SIROCCO could perform within reasonable run-time parameters even with a very large case base.

The experiment was somewhat tricky to conduct because SIROCCO's case base was limited by the number of cases that had been transcribed into EETL (246: 184 foundational cases plus 58 trial cases plus the 4 unused trial cases). Since the task of transcribing cases is labor-intensive, an alternate means of scaling up the case base was required. This was achieved by running a PERL script over the source files of the foundational cases, changing the names of all fact situations, cases, and related objects, and generating a series of 10 new files, each composed of all the foundational cases and assorted objects with new names. Such a tactic allowed each new file to be loaded into SIROCCO and interpreted as a unique set of cases. A total of 2,086 cases were now available for testing, including the original foundational cases and the trial cases.

SIROCCO was first run using the standard 246 cases as a case base. Ten additional test runs were subsequently executed in which, for each test run, the case base was increased by one of the new files, or 184 cases. For each test run, SIROCCO processed all 58 trial cases and the overall run time per test run was recorded.

Note that F-Measures were not calculated in this experiment. Because of the duplication of cases, the F-Measures in this context would not provide sensible and realistic data. The primary concern in this experiment was to gauge how fast SIROCCO would run with increasing case base size.

Quantitative Results

The results of Experiment #6 are shown in Figure 4-14. The run time of SIROCCO over the 58 trial cases increases in approximately linear time. Note, however, that the increase is not monotonic. At case base size 614, and again at case base size 1166, there is a downward spike. There is no obvious explanation for the two decreases, but probably these are a result of the duplication of cases in the case base and unusual computational effects that results from it.

With a case base size of 2,086, SIROCCO took approximately 29.4 seconds to process each of the 58 trial cases. This average run time is reasonable in the intended domain of use, in which a response is not required, or needed, in real time. When one considers the run time of a search query over the Internet, this performance does not compare unfavorably. Moreover, 2,000 represents the upper limit of the number of cases that SIROCCO is likely to require to be successful. SIROCCO's case base probably would not – or should not – grow larger than 2,000 cases. This question is addressed in the following section.

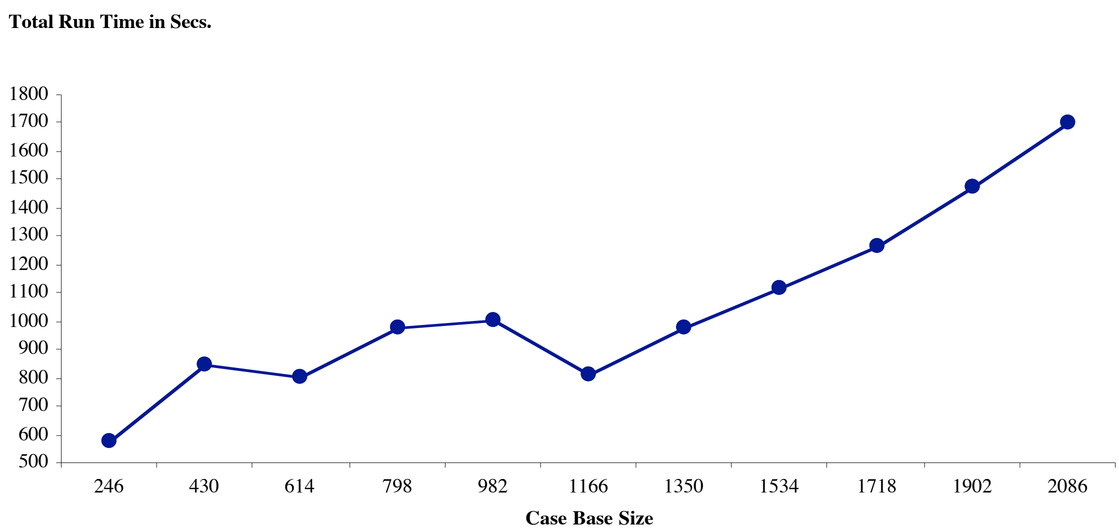


Figure 4-14: Experiment #6, SIROCCO's Efficiency as Case Base Size Increases

4.4.4. Discussion

The combined results of Experiments #4, #5, and #6 indicate that SIROCCO is a computationally feasible approach for suggesting codes and cases in the domain of engineering ethics. The program appears to run in no worse than linear time with respect to the size of the case base, and the average run time, approximately 10 seconds per case for a case base of size 246, is within reasonable limits for SIROCCO's intended use as a support tool for an engineer or engineering student.

SIROCCO's performance against a significantly larger case base, in particular of one with slightly over 2,000 cases, is slower at approximately 30 seconds per case, but it is still acceptable. Considering that not much time was spent in tuning SIROCCO for speed, as well as the fact that processors continue to get faster, this result suggests that SIROCCO could be used in practical application, even with a much larger case base.

The question remains whether SIROCCO actually requires – or would ever require – a case base of as many as 2,000 cases. Considering the intended use of the tool, it may be unnecessary for the program to access such a large case base. Consider, for instance, that in the NSPE BER domain there are a total of 46 relevant "subjects" [1996, p. 8]. That is, the review board has stated that there are 46 primary issues that its codes and cases must address. The current set of 572 NSPE BER cases decided between 1958 and 1998 is cross-referenced to this subject list, and every subject is covered by at least two of the existing cases with some subjects covered by many cases. Thus, for any new target case there are theoretically at least two cases that are relevant to the new case. Since the goal is not to solve the ethical dilemma, but rather to find relevant codes and cases related to the dilemma, this is at least adequate domain coverage.

Thus, a case base considerably smaller than 2,000 cases (of carefully represented cases) could provide appropriate coverage for a wide range of scenarios, in terms of case "competence," as described by Smyth and McKenna [1999]. Even conservatively estimating the need for covering 200 separate issues – many more than the 46 used by the NSPE BER – a case base of 2,000 cases would provide on average 10 exemplars of each issue, certainly more than SIROCCO requires to make good suggestions. In addition, since SIROCCO is expected to retrieve generally relevant, but perhaps imprecise, cases that humans then interpret, analyze, and adapt, there is perhaps less of a requirement for a comprehensive, detailed coverage of possible scenarios.

Because SIROCCO's Stage 2 algorithm can produce results only if it is provided with appropriate candidate cases with which to perform structural mapping, it is tempting to believe that one should increase N as much as possible to produce accurate results. However, the results of Experiment #5 clearly show that increasing N above the value of 6 does not improve accuracy yet greatly increases run time. As the second chart of Figure 4-13 shows, the accuracy of SIROCCO does not increase above $N = 6$; in fact, it steadily decreases. This is a result of what might be called "flooding" Stage 2. By providing Stage 2 with many cases to process, the chances increase, of course, that relevant codes and cases will be identified but, at the same time, more marginally relevant codes and cases are also found. In most of the cases in which SIROCCO at $N = 6$ outperforms SIROCCO at $N > 6$, it is because the latter run suggests not only the codes and cases that bolster the F-Measure score (i.e., exact or inexact matches to review board citations), but also additional codes and cases that do not.

4.5. Experiment Summary

This chapter described a series of experiments performed to test the primary and secondary theses of the dissertation. In particular, the experiments were designed to test whether a core set of SIROCCO's operationalization techniques allow the program to make accurate predictions of the principles and past cases that are relevant to the analysis of new cases (i.e., the primary thesis) and to test whether SIROCCO's temporal knowledge similarly supports accurate predictions (i.e., the secondary thesis). Also, several additional experiments were performed to test the efficiency and scalability of the computational model.

To test the primary thesis, SIROCCO was compared to five competitor methods, including a full-text retrieval system and a version of SIROCCO that does not employ operationalization techniques. The experimental results showed that a critical subset of SIROCCO's operationalization techniques (i.e., *Code Instantiation*, *Group Codes*, *Case Instantiation*, *Group Cases*, and *Reuse an Operationalization*) allow the program to make accurate predictions of the principles and past cases that are likely to be important in the analysis of new cases. Furthermore, the evidence convincingly showed that SIROCCO provided significant improvement in retrieval accuracy over all of the competitor methods. It was also shown that, in some instances, SIROCCO was able to make appropriate code and case suggestions that even the board did not make. These results support the primary thesis.

To test the secondary thesis, SIROCCO was compared to the performance of an ablated version of the program, NON-TEMP SIROCCO, that did not employ temporal knowledge. The experimental results of this portion of the experiments were surprising, and with respect to the secondary thesis, disappointing. The differences between SIROCCO with and without its temporal knowledge were essentially negligible. Some possible reasons why this result was obtained were discussed, including there being a limited number of trial cases involving temporal considerations and possible shortcomings of the representation and reasoning of SIROCCO.

Finally, some experiments were run to test SIROCCO's computational characteristics and to gauge how well the program scales up. These experiments indicated that SIROCCO's computational performance was clearly acceptable, performing satisfactorily with a case base of over 2,000 cases. It was argued that SIROCCO would probably not require such a large case base to make accurate predictions, so this "worst-case" result provides solid support for the practicality of SIROCCO's implementation[41].

[41] After all of the experiments had been run, the data analyzed, and a final draft of this chapter written, it was discovered that two infrequently cited NSPE BER codes were misclassified in the Code Hierarchy. Since this could have potentially changed the experimental results, as regards inexact F-Measures, Experiment #1 was re-run on both the trial and foundational cases to assess the impact. The quantitative results of the re-test revealed that the minor representation error had no impact on all tests except for MG running against the trial cases and, for this test, MG actually performed 0.005 worse *after* the correction. Thus, the minor representation problem led to only one insignificant quantitative discrepancy and no differences in the qualitative analysis reported in this chapter.

5. Related Work

5.1. Related Work in Interpretive Case-Based Reasoning

Interpretive case-based reasoning [Kolodner, 1993, p. 86-92] is a subarea of CBR in which computational methods are developed to interpret, evaluate, decide, and justify arguments through reference to past cases. As was discussed in Section 1.1, weak analytic domains are particularly appropriate for the application of interpretive CBR. These domains are governed by abstract laws, policies, principles, or theories, but often there are no authoritative or readily available intermediate rules that can connect the abstract rules to the specific facts of cases. Nevertheless, human decision makers apply the abstract rules in deciding actual cases. The record of such applications generates reusable information that can help bridge the gap between abstract rules and specific facts. The past cases help to frame a new problem. By evaluating similarities and differences between the new problem and the past cases, one is often able to appropriately interpret or resolve the new problem and reapply abstract rules that were applied in the past cases.

Interpretive CBR Applied to the Legal domain: A Contrast with Engineering Ethics

The majority of interpretive CBR work has been applied to the legal domain [Ashley, 1990; Branting, 1991; Rissland and Skalak, 1991; Rissland *et al.*, 1996; Aleven, 1997]. Interpretive CBR is well-suited to this domain for two primary reasons. First, legal reasoning involves argumentation: creating arguments, justifying arguments, and comparing arguments. In fact, the Anglo-American legal system essentially institutionalizes the concept of adversarial arguments. In a legal proceeding, two parties, the plaintiff and defendant, are pitted against one another and each presents arguments for a position and interest while refuting the opponent's position. Second, legal arguments are usually supported by previously decided cases [Llewellyn, 1930]. Laws, like ethical principles, are typically quite abstract and difficult to directly apply to specific cases. Because of the open-textured and abstract nature of laws, attempts to computationally address legal problems in a deductive fashion have had limited success [Sergot *et al.*, 1986; Berman and Hafner, 1986]. On the other hand, previously decided cases provide a way of interpreting laws and indirectly linking laws to new cases. In particular, comparing a new case to a previously decided case may allow one to draw an analogy that guides how the laws governing the previous case apply to the new case.

In an important and influential book, Twining and Miers [1976] analyze the difficulties of bridging the gap between abstract, legal rules and specific facts given the way problems are described in the legal domain. For instance, the authors discuss a structure called a "ladder of abstraction" that can be used to represent legal issues in cases by "a continuous sequence of categorizations from a low level of generality up to a high level of generality" [1976, p. 40, p. 45-46]. The ladder of abstraction idea appears to be a means of circumventing the problem of a lack of intermediate rules in the legal domain. The general idea is that laws and legal theories are applied to the abstract facts at higher levels of the ladder, leading eventually to the resolution of the more-specific facts at lower rungs. On the other hand, the approach appears to focus on the application of these "ladders" to specific, noteworthy cases, rather than on the development of a general-purpose representation. The Twining and Miers work could also be said to be an important precursor to the AI and Law work, as it suggests the possibility of resolving legal issues in a computational fashion by casting complex legal rules as algorithms and flowcharts [1976, p. 182-196].

As compared to legal reasoning, engineering ethics involves a less-explicit model of argumentation. There is no authoritative approach to resolving engineering ethics problems. Ethical arguments are typically more free-form in style and structure. An organization such as the NSPE BER provides one example of how engineering ethics problems can be argued and resolved, but their opinions are purely educational; by no means are they binding, in the same sense as legal decisions. A second distinction is that the decision-making process in engineering ethics does not always (or even typically) involve two adversaries who present arguments and counter-arguments. Also, decisions in engineering ethics cases are not constrained to binary conclusions (e.g., plaintiff winning or losing) but may suggest multiple actions and outcomes. For instance, in deciding whether a particular action is ethical or unethical in a case, the NSPE BER often suggests ways in which the protagonist might have avoided the ethical dilemma altogether or ways to correct an unjust action after the fact. The goal in evaluating engineering ethics problems is not to assign blame and/or punishment for unethical actions; rather, it is to provide an opportunity for interested parties to learn more about the ethical ramifications of various actions and to consider what could have been done differently in the given circumstances. It is also important that engineering practitioners recognize and understand the conflicting values in specific fact situations and learn to apply "creative middle way" solutions to those situations (i.e., resolutions that at least partially meet each of the conflicting values) [Harris *et al.*, 1999, p. 64-72]. Finally, the two domains differ in terms of access to case data. The legal domain has an extensive body of on-line cases, available through services such as Westlaw, while engineering ethics provides far fewer case examples and opinions to work with. For instance, the NSPE BER case set is very small in comparison to legal case libraries.

The codes and principles used to resolve dilemmas in engineering ethics are abstract, open-textured rules, and the domain clearly requires techniques to bridge the gap between these rules and the specific circumstances of cases. Harris and colleagues summarize this as follows: "No code of ethics (in engineering ethics) is self-interpreting. Its principles and rules are stated in general terms and need to be applied thoughtfully to particular circumstances; and some parts of a code might potentially conflict with another." [1999, p. 21] In other words, engineering ethics is a weak analytic domain, as discussed in Sections 1.1 and 2.1.1. Techniques that are applied to bridge the gap between abstract rules and the particular facts of cases are, in effect, a substitute for the lack of intermediate rules in the domain.

Summary of the Comparison Between SIROCCO and Other Interpretive CBR Work

In the following sections, various interpretive CBR programs, in both ethics and the legal domain, are discussed and compared to SIROCCO.

In summary, SIROCCO generally differs from the other CBR programs in the following ways. First, SIROCCO attempts to bridge the gap between abstract, open-textured rules and specific facts by using a detailed, narrative representation of cases, including temporal relationships between facts, and the application of the operationalization techniques to that representation. A couple of the interpretive CBR programs discussed below also attempt to bridge this gap, in particular GREBE and BankXX, but the approach they employ is somewhat different than SIROCCO's. GREBE's primary tool in bridging the gap between abstract rules and specific facts is the representation and reuse of a judge's justification in applying a law to a fact situation. While this tactic is similar to a *Code Instantiation* in SIROCCO, the other operationalization techniques discussed in this dissertation are either absent or figure far less prominently in GREBE's approach. BankXX attempts to bridge the gap between theories and facts by interlinking cases, legal theories, factors, and typical scenarios and by executing a heuristic search over this heterogeneous structure. However, in BankXX, the facts of a case are not represented in narrative detail and, as in GREBE, the full set of SIROCCO's operationalization techniques do not appear to support the program in bridging the gap.

Second, the use of a limited language to represent cases as narrative sets of temporally ordered steps differentiates SIROCCO from most other interpretive CBR programs. GREBE's representation scheme is not wholely dissimilar from SIROCCO's – actions and events of a scenario are represented – but GREBE's representation does not include formal temporal considerations and provides far fewer actions and events. As discussed in the following sections, none of the other interpretive CBR programs (i.e., TRUTH-TELLER, BankXX, and CATO) use a representation that is a narrative description of the facts of a case, and none incorporate temporal considerations in their representation and reasoning.

Some work in CBR has emphasized the use of limited languages, most notably projects descended from Roger Schank's conceptual dependency work [Lytinen, 1992; Schank, 1972], for example, SWALE [Kass *et al.*, 1986, Leake, 1991]. However, the main contrast between SIROCCO and SWALE is that SIROCCO's aim is very practical and functional: to achieve structural similarity assessment for the purpose of retrieving relevant cases and codes from a database of ethics cases. While the creators of SWALE were interested in generating detailed comparisons and explanations of anomalous events in different domains, I have been more concerned with applying SIROCCO over a wide variety of cases and testing this capability empirically.

This contrast with SWALE highlights the final key distinctions between SIROCCO and other interpretive CBR work. In this dissertation, SIROCCO has been shown to handle a relatively wide variety of case types in retrieving codes and cases. All of the interpretive CBR programs discussed below (i.e., TRUTH-TELLER, GREBE, BankXX, and CATO), as well as most other work in this area, are focused on providing detailed analyses and comparisons between cases and not on handling a wide variety of cases for purposes of retrieval, as is SIROCCO. Consequently, the domain coverage of these systems is significantly less than that of SIROCCO, which leverages its limited language, its case-acquisition web site, and matching techniques to provide a relatively wide domain coverage. Finally, the SIROCCO experiments represent the capability of the program more objectively than do the experiments and evaluations of the other interpretive CBR systems. In particular, all of SIROCCO's test cases – and a significant number of its foundational cases – were transcribed by independent case enterers. The base and test cases for TRUTH-TELLER, GREBE, BankXX, and CATO were all represented by the researchers themselves.

5.1.1. TRUTH-TELLER

In many respects, SIROCCO was inspired by earlier work that Kevin Ashley and I did on TRUTH-TELLER [1994a; 1994b; 1995a; 1995b; 1995]. TRUTH-TELLER is a program that compares pairs of cases presenting ethical dilemmas about whether or not to tell the truth. The program marshals ethically relevant similarities and differences between two input cases from the perspective of the "truth teller" (i.e., the person faced with the dilemma) and reports them to the user. In particular, it points out reasons for telling or not telling the truth that (1) apply to both cases, (2) apply more strongly in one case than another or (3) apply to only one case. TRUTH-

TELLER employs two abstraction hierarchies to support its task: a Reasons Hierarchy, which organizes reasons or rationales for telling the truth according to facets that are important in ethical reasoning [Bok, 1989], and a Relations Hierarchy, which represents human relations (e.g., spouse, friend, business associate) and the incumbent level of duty and trust expected in such relations [Aristotle, edited and published in 1924, Books VIII and IX; Jonsen and Toulmin, 1988, p. 290-293]. The two hierarchies are used to classify and moderate the support for and against telling the truth in each scenario. The hierarchies also help TRUTH-TELLER compare the cases by matching reasons across the cases.

The TRUTH-TELLER work was a first step in implementing a computational model of casuistic reasoning. SIROCCO, while not a direct descendant of TRUTH-TELLER, is nevertheless a step closer to that goal. As mentioned in Section 2.1.1, casuistry is a form of ethical reasoning in which decisions are made by comparing a problem to paradigmatic, real or hypothetical cases [Jonsen and Toulmin, 1988; Strong, 1988; Arras, 1991]. The TRUTH-TELLER project focused on comparing cases, but it ignored the problem of how potentially relevant cases are retrieved in the first place. The program compares any pair of cases it is provided, no matter how different they may be. SIROCCO, on the other hand, uses retrieval to determine which cases are most likely to be relevant to a given target case. After it has retrieved potentially relevant cases, SIROCCO compares the retrieved cases to the target and uses the results of those comparisons to collect relevant information to present to the user.

The representation perspectives of the TRUTH-TELLER and SIROCCO projects are also quite different. In TRUTH-TELLER, each case is focused on the main protagonist's reasons for and against telling the truth. For instance, Figure 5-1 depicts TRUTH-TELLER's representation of the case "Should Stephanie, a psychology researcher, lie to human subjects about the intent of an experiment in order to study some aspect of the subject's behavior?" In this case, Stephanie is the "truth teller" and the actor(s) who may receive the truth, i.e., the "truth receivers," are the experiment subjects. Stephanie can take several possible actions: tell the experiment subjects the truth, tell them a lie, or perhaps think of a compromise solution. Each of these possible actions has reasons that support it. For instance, two reasons for Stephanie to tell the truth are (1) the subjects have the right not to be deceived and (2) Stephanie may be professionally harmed if she is caught lying. TRUTH-TELLER's task is to compare one case to another by aligning and comparing the reasons represented in each case.

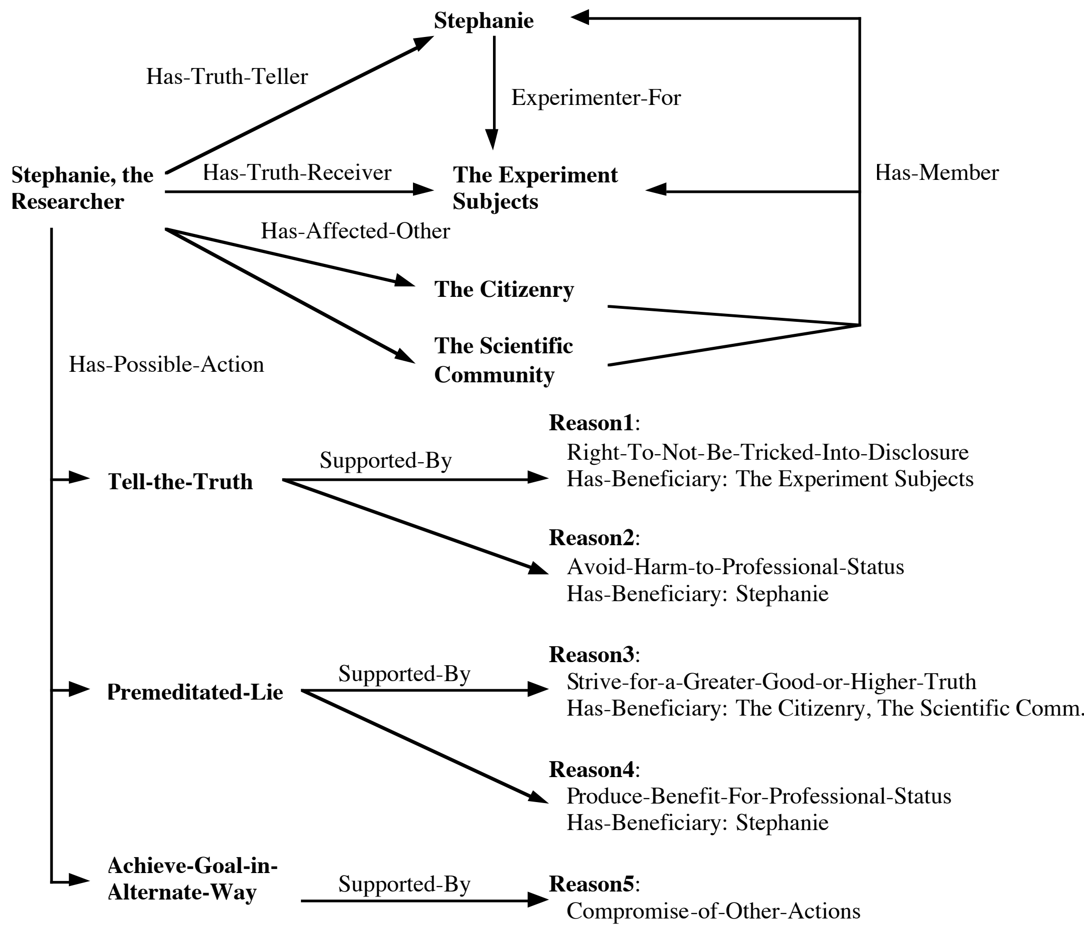


Figure 5-1: An Example of TRUTH-TELLER's Case Representation [Ashley and McLaren, 1995]

The smallest representational element in TRUTH-TELLER is a *reason*. To represent a case, case enterers must elaborate the ethical and nonethical reasons for (and against) telling the truth by interpreting the text of the case. Such a representation, while quite useful for generating the detailed and issue-focused comparison texts produced by the program, is constrained in its general applicability. Essentially, TRUTH-TELLER is very good at comparing truthtelling dilemmas in a sophisticated and meaningful way, but it cannot tackle other types of ethical problems without considerable augmentation of its case representation and Reasons Hierarchy.

In SIROCCO, primitives that closely model some of the actual actions and events of a fact situation are used to represent cases as complex narratives. In this sense, SIROCCO's representational approach is more general than TRUTH-TELLER's. This is key to SIROCCO's ability to address a much wider range of cases than TRUTH-TELLER addresses; not only does SIROCCO handle ethical issues regarding honesty, it can also handle scenarios regarding public safety, confidentiality, conflict of interest and more. In addition, SIROCCO's representation is more appropriate for untrained case enterers to transcribe cases – it requires far less abstraction from the actual facts of the case and thus enables the collection of a greater number and range of cases. On the other hand, SIROCCO's case comparisons are not nearly as precise and issue-oriented as TRUTH-TELLER's. This is the trade-off for addressing a wider variety of cases.

5.1.2. GREBE

GREBE classifies, resolves, and justifies new fact situations in the domain of worker's compensation law by retrieving explained legal precedents that are built, in part, from legal rules and theories [Branting, 1991]. GREBE combines rule-based reasoning and case-based reasoning by simultaneously applying both techniques to a new problem and then evaluating and comparing the explanations produced by each. A case in GREBE is represented by both the raw facts of the situation and the explanation of the court in reaching its conclusion. The program's explanation structures, exemplar-based explanations (EBEs), connect the criterial facts of a case to the legal rules or theories applied by a judge in that case.

There are a number of similarities between SIROCCO and GREBE. First, SIROCCO's representation of raw facts is similar in some respects to GREBE's. For instance, both SIROCCO and GREBE model actions and events in their respective representations. Second, SIROCCO's *Code Instantiations* serve a similar function as GREBE's EBEs. Both structures essentially provide fact-specific justifications for the application of abstract rules to cases and help to focus structural mappings between a target and a source case. Finally, both programs perform A* search to find the best structural mappings and are able to gainfully use even those mappings that do not match precisely.

There are also significant differences between SIROCCO and GREBE. SIROCCO's representation of actions and events is much more extensive than GREBE's. SIROCCO provides a total of 190 actions and events, while GREBE provides approximately 70 to 90[42]. While SIROCCO's representation emphasizes a narrative description of the facts of a scenario, GREBE provides some elements of narrative description but more strongly emphasizes the explanation of a scenario's actions and events. Part of SIROCCO's emphasis on narrative representation and comparison are evident in the formally defined temporal relations and a well-defined algorithm for matching such relations. GREBE provides an incomplete and informal set of temporal relations (e.g., concurrent, occurred-during), and it does not include a temporal matching scheme

[42] This is an estimate based on GREBE's 147 relations [Branting, 1991, Appendix]. Of the 147 relations, many are clearly used to represent GREBE's explanations and not the raw facts of a case.

as part of its structural mapping routine. Other than *Code Instantiations*, GREBE does not appear to have equivalents to SIROCCO's operationalization techniques, the foundational elements of SIROCCO's architecture. For instance, GREBE does not appear to have equivalents to the operationalization techniques *Group Codes*, *Group Cases*, *Rewrite a Code*, *Define or Elaborate a General Issue or Principle*, and *Apply a Hypothetical to a Code*. GREBE takes inexact structural mappings and attempts to improve those mappings (i.e., the explanations) by hypothesizing facts in the target case. SIROCCO, on the other hand, does not attempt to improve mappings but, rather, uses inexact (but good) matches as accumulated evidence for the relevance of particular codes and cases.

As previously discussed, one of the primary goals of the SIROCCO project was to address a wider range of cases than predecessor interpretive CBR systems. GREBE, on the other hand, focuses on reasoning over a much narrower range of cases in order to provide extensive arguments and explanations of each. The greater number of actions and events available in SIROCCO is one key to SIROCCO's wider domain coverage. Another is the use of the case-acquisition web site. The web site provides examples, guidelines, and implicit constraints on how cases are to be represented. The web site also supported an objective evaluation of SIROCCO. Twelve different case enterers represented foundational cases for SIROCCO and all of the trial cases were represented by two of these case enterers. Branting represented his own cases to test GREBE.

GREBE depends on consistency of representation without providing any practical means to achieve it. That is, GREBE's structural mapping approach fails unless cases are consistently represented. In contrast, SIROCCO's web site and limited language provide some means to achieve consistency. In addition, SIROCCO's generalized matching techniques attempt to reduce the need for perfect consistency.

Yet GREBE also has advantages over SIROCCO. As discussed above, GREBE's representational focus on explanations of past decisions, in the form of causal and evidential relations, allows GREBE to provide detailed analyses of target problems, something SIROCCO is incapable of doing. The evidential relations of GREBE's EBEs are richer and more various than SIROCCO's Instantiation links from Facts (i.e., the individual steps of a Fact Chronology) to relevant codes and past cases. Also, as a system that combines rule-based reasoning and case-based reasoning, GREBE provides alternative means for addressing new cases. SIROCCO depends exclusively on cases for retrieving appropriate codes and cases.

5.1.3. BankXX

The BankXX program represents an attempt to model the process of a junior associate in a law firm searching for, or "harvesting," relevant information in developing an argument for a given case [Rissland *et al.*, 1993; 1996]. The program operates by performing heuristic search over a highly interconnected network of cases, theories, and other relevant legal knowledge in the domain of bankruptcy law. Its goal is to collect as much information as possible to support a variety of "argument pieces" (e.g., supporting cases, leading cases, applicable legal theories), given a limited run time. The argument pieces represent segments or portions of legal knowledge that would prove useful to an attorney in developing an argument.

BankXX's goal is very similar to SIROCCO's. In both programs the aim is to collect relevant information that could be used in arguing or justifying a conclusion. However, although BankXX organizes its retrieved information according to the argument pieces and SIROCCO provides explanation that could support an argument, neither program actually generates an argument. The programs also similarly combine knowledge-base indexing and heuristic search. Finally, Rissland and colleagues have performed a series of empirical experiments to test the capabilities of BankXX [1997]. The design of those experiments provided the foundation for the SIROCCO experiments presented in this dissertation.

A key difference between BankXX and SIROCCO is the way in which heuristic search is used. In BankXX heuristic search is the primary tool for harvesting information. The program starts at a node in its case graph and performs best-first search, iteratively collecting relevant information until it runs out of allotted search time. SIROCCO, on the other hand, uses heuristic search in a much more focused manner. SIROCCO performs heuristic search on only a handful of potential mappings, those that are retrieved by its knowledge-base indexing procedure (i.e., the combined dot product calculation). In addition, the search performed by SIROCCO is focused on just a portion of each retrieved source case, in particular, the Critical and Questioned Facts of each case.

An impressive aspect of BankXX is its capability to view and search cases from a number of different perspectives. BankXX's case graph is organized as a collection of "spaces," including a Fact Situation Space, a Legal Theory Space, a Legal Factor Space, a Legal Citation Space, and a Legal Story Space. This structure is depicted in Figure 5-2. Objects in one space may be linked to other objects in the same space, as Case A is linked to Case B in Figure 5-2, or to objects in another space, as Case B is to Theory A. Searching in and between these different spaces allows the program to harvest information that is relevant to the different perspectives. SIROCCO provides no equivalent capability.

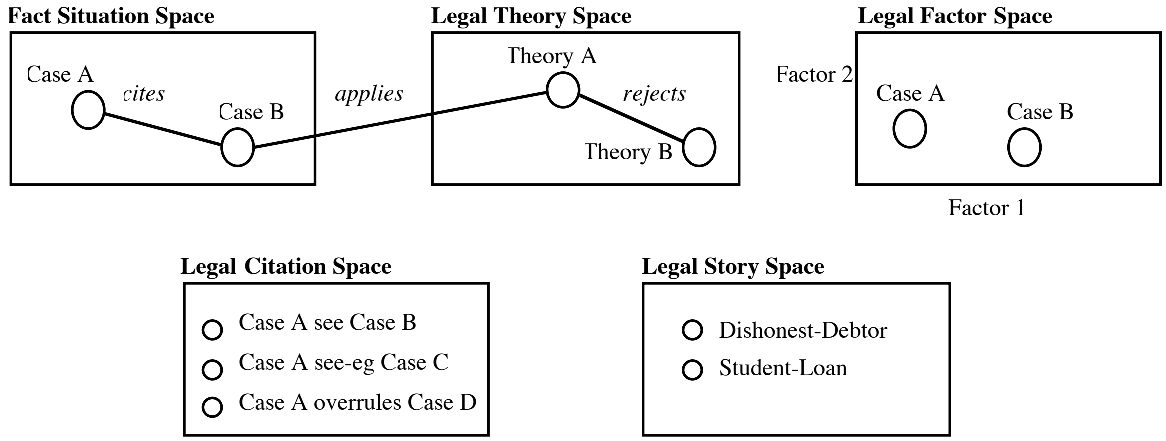


Figure 5-2: An Example of BankXX's Case Representation [Rissland *et al*., 1993]

Cases in BankXX are essentially clusters of objects from different perspectives that are related to one another. However, unlike SIROCCO, BankXX's representation does not address the narrative description of a case, i.e., it does not model actions and events or the temporal relations between those actions and events. BankXX bridges the gap between theories (i.e., the abstract rules in this domain) and facts by providing links from the Fact Situation Space to the Legal Theory Space. However, because (1) the nodes are not really a representation of the raw facts and temporal relationships between those facts and (2) the links between these two spaces do not relate to detailed facts and are not necessarily the emphasis of BankXX's search it is less clear that this approach makes the same impact in bridging the abstraction gap that, for instance, SIROCCO's focused *Code Instantiations* make.

Finally, BankXX also differs from SIROCCO in some of the same ways that GREBE differs from SIROCCO. Again, other than *Code Instantiations* and perhaps *Case Instantiations*, BankXX does not appear to have analogs to most of SIROCCO's operationalization techniques. In particular, BankXX does not appear to have equivalents to the operationalization techniques *Group Codes*, *Group Cases*, *Rewrite a Code*, *Define or Elaborate a General Issue or Principle*, and *Apply a Hypothetical to a Code*. Also, although the authors performed extensive experiments with the program, the experiments did not demonstrate that BankXX is capable of handling cases outside of its relatively narrow domain. The BankXX experiments were also arguably less objective than the SIROCCO experiments because independent case enterers were not used to define cases. In the BankXX experiments, the authors themselves represented all of the cases.

5.1.4. CATO

CATO is an intelligent tutoring system applied to the legal domain [Aleven, 1997]. CATO makes a contribution to the case-based reasoning literature by applying abstract, background knowledge to the task of similarity assessment. The program uses its background knowledge, represented in a Factor Hierarchy, to assess and explain the significance of similarities and differences between cases. CATO generates alternative arguments that a partially matched source case is reasonably close (or not) to a target case.

As with TRUTH-TELLER, GREBE, and BankXX, CATO differs from SIROCCO in its representation scheme. CATO's core representation element is a *factor*. Inherited from Ashley's work [1990], a factor does not represent a detailed action or event of a scenario, but rather it provides an interpretation as to whether certain facts favor one side or the other in a legal dispute. In other words, it is an abstraction of the raw facts. For instance, consider Figure 5-3. This is the representation of a case in which a bar owner, Mason, filed a legal claim against the Jack Daniel Distillery. Mason, the plaintiff, claimed that the distillery had misappropriated his secret formula for a mixed drink, backing out on a promise that Mason's logo would be used in the distillery's sales promotion. The elements shown below the case title in Figure 5-3 are the factors of the case. Three of those factors (i.e., F6, F15, and F21) favor Mason, the plaintiff, and two (i.e., F1 and F16) favor Jack, Daniels, the defendant.

Mason v. Jack Daniel Distillery
F1: Disclosure-In-Negotiations (d)
F6: Security-Measures (p)
F15: Unique-Product (p)
F16: Info-Reverse-Engineerable (d)
F21: Knew-Info-Confidential (p)

Figure 5-3: An Example of CATO's Case Representation [Aleven, 1997]

The goal of CATO is to provide relatively detailed comparisons of cases, and such a representation supports that goal. The factors summarize key facts in a way that makes it possible for CATO to compare the issues of one case with those of another. On the other hand, CATO's factors tend to limit the possibility of representing a wide variety of cases. Widening domain coverage would involve the careful identification and representation of the key issues of other subdomains of the law. In contrast, many of SIROCCO's actions and events are not specific to a particular subdomain of engineering ethics (e.g., "employed-by," "inspects," "informs") and therefore support the representation of a wider range of case types. As a result, SIROCCO's domain coverage is significantly wider than is CATO's. At the same time, this feature limits SIROCCO's capability to provide detailed comparisons, such as those generated by CATO.

Like SIROCCO, CATO compares cases at multiple levels of abstraction. However, the primary representational devices used by each program to support abstract comparisons – i.e., CATO's Factor Hierarchy and SIROCCO's Action/Event Hierarchy – are different in both purpose and structure. Whereas the Factor Hierarchy is employed by CATO to develop arguments and reason symbolically about similarities and differences between cases, SIROCCO's Action/Event Hierarchy is used to support the retrieval and (numeric) evaluation of cases. CATO's Factor Hierarchy is not used to support case retrieval. The Factor Hierarchy relates basic concerns (represented as factors) to higher-level or more-abstract concerns and issues of the legal domain. The links between nodes in the hierarchy are evidentiary, providing strong support, weak support, weak refutation, or strong refutation from the lower to higher nodes. This also contrasts with SIROCCO's Action/Event Hierarchy, in which the links are more taxonomic in nature. In particular, higher-level nodes of the Action/Event Hierarchy are simply more-abstract actions or events than lower-level nodes. Finally, note that widening CATO's domain coverage would involve not only the representation of additional factors, as discussed above, but also significant extensions and augmentation to its Factor Hierarchy.

In some respects, CATO's Factor Hierarchy can be viewed as one approach to the problem posed in this dissertation: relating abstract, open-textured rules to specific facts. At the top level of the Factor Hierarchy are Legal Issues such as Confidential-Relationship and Info-Trade-Secret. Below these are intermediate nodes, akin to intermediate hypotheses, while at the leaves of the hierarchy are the base-level Factors (e.g., Knew-Info-Confidential, Agreed-Not-to-Disclose) used to represent cases. Thus, a pattern of Factors representing a case in some sense "invokes" an abstract rule (i.e., a Legal Issue) through a series of evidentiary links.

The difference between this and SIROCCO's approach, however, highlights the key concept of this dissertation: operationalization. CATO's Factor Hierarchy, like SIROCCO's Action/Event Hierarchy, is strictly an intensional representation; it is a predefined knowledge structure generalized from an analysis of the domain. On the other hand, SIROCCO's primary problem solving structures, the operationalization techniques, are purely extensional. The operationalization techniques are represented and defined as patterns of specific facts and citation information from the past cases that applied them. Thus, while CATO primarily uses generalized abstraction knowledge to achieve its aims, SIROCCO partially uses such knowledge but emphasizes the use of operationalized and specific knowledge in retrieving and analyzing cases.

5.2. Related Work in Analogical Reasoning

In many respects, work in analogical reasoning, a subfield of cognitive science, is similar to work in case-based reasoning. Both analogical reasoning and case-based reasoning have roots in early work by Thomas Evans [1968] and, somewhat later, in work by Patrick Winston [1980]. Evans was the first to write a computational model of analogy; his program solved problems of proportional analogy, such as those typically found on an intelligence test (e.g., A is to B, as C is to ?). Winston's program was the first to use structured representations of analogs and also was the first to emphasize the importance of both semantic similarity and purpose. In the present day, much of analogical reasoning research, as in CBR research, is focused on how one can understand and/or resolve a new problem or fact situation in terms of an established, authoritative, or prototypical past fact situation. In essence, both CBR research and analogical reasoning research stress and explicitly acknowledge the importance of analogy in intelligent reasoning.

On the other hand, there are significant differences in the two fields of research. In analogical reasoning, researchers are typically interested in specifically how *humans* create and use analogies [Keane *et al.*, 1994; Holyoak and Thagard, 1995]. While researchers in case-based reasoning are also interested in human analogy – and they often take cues from human reasoning in developing automated case-based reasoners [Kolodner, 1993, p. 27-29] – the goal in CBR, especially in recent years, tends to be much more practical and computationally-oriented. That is, researchers in case-based reasoning typically want to build computational models that address or solve practical problems using cases, regardless of whether their programs directly model the reasoning of humans. In contrast, analogical reasoning researchers usually build programs to test whether, and how well, the programs model human reasoning.

Another key difference is that analogical reasoning research tends to emphasize cross-domain analogies. For instance, a classic analogical reasoning problem is "The Tumor and the Fortress" [Holyoak and Thagard, 1995, p. 110-116]. In this problem, participants in an experiment are presented a scenario to solve in which a medical patient has an inoperable tumor. The only way to destroy the tumor is to concentrate a special kind of ray on it that will unfortunately also destroy the surrounding healthy body tissue. The participants are also provided with a solved case in which an invading army attacked and overran a fortress by dividing into smaller units and converging simultaneously on the fortress from different directions. This tactic was necessary because the leader of the fortress planted land mines along roads leading to the bastion, and these mines would detonate if a large number, but not a small number, of people passed over them. By using a cross-domain analogy, the inoperable cancer problem could be solved by reapplying the fortress solution. In particular, if the rays were aimed at the tumor in thin beams from many angles, the life-threatening growth could be destroyed while at the same time the patient's healthy tissue was preserved.

Analogical reasoning researchers are primarily interested in studying and understanding the mental processes that allow people to disregard surface features and perform creative cross-domain analogies in problems like "The Tumor and the Fortress." "Theoretical attention is inevitably drawn to spontaneous analogy: That is, to structural similarity unsupported by surface similarity, as in (a periodic table reminding one of octaves in music.)" [Forbus *et al.*, 1994, p. 142] On the other hand, although there are exceptions, for instance, the SWALE work [Kass *et al.*, 1986, Leake, 1991], CBR researchers primarily focus on how analogy, or case similarity, is used within a single domain.

There also tend to be differences in the types of computational models built by researchers in these two fields. While analogical reasoning programs typically use structural representations and implement analogical mapping techniques between these structures, CBR programs tend to focus more on flat feature representations, fact summaries, and surface characteristics of cases. Many CBR programs adapt retrieved analogs to guide current decision making and save solutions for future reuse; most analogical reasoning programs focus strictly on retrieving analogs. These differing approaches stem directly from the different objectives of each field: analogical reasoning attempts to model humans and their ability to recognize cross-domain analogies, while CBR attempts to provide computationally efficient solutions to practical problems.

The SIROCCO research, however, overlaps with analogical reasoning research perhaps more than most other CBR work. First, the SIROCCO program is based on a study of how human reasoners actually solve (or justify) problems in engineering ethics. The results of this study, in particular the operationalization techniques, provided the framework for the design of SIROCCO. Second, the underlying algorithm of SIROCCO has similarity to some of the better-known analogical reasoning systems, in particular in their use of structural analogy. In the following sections, two of these analogical reasoning systems are briefly discussed and compared to SIROCCO.

5.2.1. MAC/FAC

MAC/FAC (Many Are Called but Few Are Chosen) is a computational model of analog retrieval that attempts to represent psychological phenomena in a two-stage algorithm [Forbus *et al.*, 1994]. In particular, Forbus and Gentner purport to model the following human approach to analogy: (a) superficial similarities are most important in retrieval, (b) structural similarities are most important in analog comparisons, and yet (c) purely structural retrievals sometimes occur. The program's first stage (MAC) uses a computationally inexpensive "filter" algorithm that focuses on surface features of a target case and retrieves a small number of the most-likely candidates. The second stage (FAC) identifies the best analog among the first-stage candidates by using structural mapping to compare the target to all of the candidates.

An obvious, but somewhat superficial, similarity between MAC/FAC and SIROCCO is the two-stage algorithm. In fact, SIROCCO's algorithm is inspired, in part, by MAC/FAC's. However, there are many differences in the particular two-stage algorithm implemented in the two programs. First, MAC/FAC relies totally on exact matches in both its first and second stages. SIROCCO, as we have seen, is capable of matching target and source cases at several levels of abstraction, using its Action/Event Hierarchy. Second, MAC/FAC, unlike SIROCCO, does not use "goals" as a means of focusing its retrieval. Forbus and colleagues make the following claim: "By treating goals as just one of many kinds of higher-order structures, we escape making the erroneous prediction of many case-based reasoning systems: that retrieval requires common goals." [1994, p. 192] The contention that CBR systems erroneously focus on goals is, at least in the context of SIROCCO, not supported. SIROCCO's focus on goals is a key to its retrieval success. SIROCCO's emphasis on questioned and critical facts – essentially, its retrieval "goals" – is a critical reason why it outperformed EXTENDED-MG and NON-OP SIROCCO in the experiments reported in this dissertation. Further evidence that the purpose or goals of retrieval are important in analogical mapping is provided by Spellman and Holyoak [1993].

A third difference between MAC/FAC and SIROCCO is the particular way structural mapping is deployed. In MAC/FAC, structural mappings are attempted between the complete representations of the target and each of the source cases. In SIROCCO, structural mapping is efficiently focused on only a portion of the target and source cases, i.e., those Facts that are part of the source's *Code* and *Case Instantiations*. Fourth, while MAC/FAC completely relies on its first stage of processing to collect viable candidates, SIROCCO, as we saw in the program trace of Chapter 3, can select cases as reasonable analogs even if the first stage happens to miss them.

Finally, while SIROCCO's architecture includes a formal notion of temporal relations and their structural mapping across cases, MAC/FAC's does not.

It should also be mentioned that the programs differ markedly in their objectives, as do most analogical reasoning and CBR programs. While MAC/FAC is designed to model and test cognitive psychological phenomena, SIROCCO is designed to optimize the performance of retrieval and analysis in the domain of engineering ethics. The empirical study of the engineering ethics domain, and the identification of the operationalization techniques, informed SIROCCO's design and development, but the program is not intended to be a mental model of human reasoning.

5.2.2. ARCS

ARCS (Analog Retrieval by Constraint Satisfaction) is another two-stage analogical reasoning program [Thagard *et al.*, 1990]. ARCS's initial stage collects source analogs that share some semantic similarity to a target case. It achieves this by using an electronic thesaurus called WordNet [Miller *et al.*, 1990] that stores words in a semantic network that is intended to model how concepts are organized in human memory, using relations such as "part-of" and "kind-of." For instance, the word "lion" might be stored as a "kind-of" feline, which is a "kind-of" carnivore, which is a "kind-of" mammal, and so on. For each source candidate identified in the first stage, ARCS's second stage constructs a constraint network that accounts for structure, purpose, and similarity in the mapping between source and target. ARCS then executes a parallel constraint satisfaction algorithm on the networks to identify the candidate that is most similar to the target[43].

ARCS, like MAC/FAC, is intended to model human analogical reasoning. For several reasons, one could argue that it is, in fact, a better approximation of human reasoning than MAC/FAC. First, the connectionist approach of ARCS could be claimed to more closely resemble the combined physical and logical functioning of human memory than the symbolic approach of MAC/FAC. Second, the use of WordNet allows ARCS to retrieve inexact matches, something humans are clearly capable of doing, but MAC/FAC is incapable of. Third, ARCS, unlike MAC/FAC, incorporates the purpose or goal of retrieval in its mapping process. As discussed in Section 5.2.1, there is empirical evidence that incorporating purpose into the retrieval process does make a difference in the type of analogs human's retrieve.

---

[43] Holyoak and Thagard refer to *three* stages in describing ARCS [1995, p. 254]. They divide the construction of the constraint networks and the execution of the relaxation algorithm into separate stages. However, the combination of these steps is effectively equivalent to MAC/FAC's and SIROCCO's structural mapping stage, so ARCS will be referred to as a two-stage algorithm.

For the second and third reasons above, ARCS could be considered more similar to SIROCCO than MAC/FAC. That is, ARCS, like SIROCCO, provides a means of supporting inexact matches. ARCS's use of WordNet serves a similar function as SIROCCO's Action/Event Hierarchy. Also, ARCS's emphasis on the purpose of analogical mapping is similar to SIROCCO's focus on mapping critical and questioned facts of source cases.

On the other hand, there are clear differences between the two programs. First, ARCS's connectionist approach to analogical mapping contrasts sharply with SIROCCO's purely symbolic approach. Second, similar to the contrast discussed above between SIROCCO and MAC/FAC, ARCS performs a mapping over the entire source and target representations, while SIROCCO's mapping process is focused on the most relevant portion of the source cases, as defined by the *Code* and *Case Instantiations*. Third, the ARCS model, like MAC/FAC, does not provide a means for mapping temporal relations between cases. Finally, ARCS is claimed to model human analogical reasoning; no such claim is made about SIROCCO.

5.3. Related Work in Practical Ethics

The domain of practical ethics, and engineering ethics in particular, is a novel domain for applying AI case-based reasoning techniques. Other than the earlier work that Kevin Ashley and I did with TRUTH-TELLER, discussed in Section 5.1.1, to my knowledge there has been no other intelligent system or computational model developed to address the interesting and challenging problems that arise in this domain.

However, there have been research-related efforts to develop computer programs to assist students in grappling with the thorny problems of practical ethics. Two of these programs, the Ethos System and the Dax Cowart CD, are briefly discussed in the following sections. Neither is an intelligent system *per se*, but each provides a valuable pedagogical resource. The programs share a number of similarities, including an open exploratory environment, the use of video clips to provide a visceral experience of ethical problems and problem solving, and a detailed focus on a single, or very few, ethical dilemma(s).

5.3.1. The Ethos System

The Ethos System was developed by Donald Searing to accompany the engineering ethics textbook written by Harris and colleagues [1999]. The program implements the HARPS methodology, an attempt to encourage rational and consistent ethical problem solving [Searing, 1998, p. 1-2]. The goal of HARPS is (1) to provide a framework in which one can rationally apply moral beliefs and (2) to record the step-by-step decisions taken by a moral decision-maker in resolving an ethical dilemma, so that those steps can later be analyzed and better understood. The types of ethical scenarios presented and explored by the Ethos System involve ill-defined concepts and conflicting principles and obligations.

The Ethos System and HARPS methodology essentially provide a means for the user to experiment with and apply the problem-solving approach advocated in the Harris *et al.* textbook. In particular, the program breaks down moral decision making into three major steps: (1) framing the problem, (2) outlining the alternatives, and (3) evaluating the alternatives. Framing the problem involves identification of the factual, conceptual, and moral issues of the dilemma. Outlining the alternatives involves the application of interpretive techniques such as line-drawing and conflict resolution. Finally, evaluating the alternatives entails the application of classic moral evaluation techniques such as utilitarianism [Mill, edited and published in 1979] and respect for persons [Kant, edited and published in 1969]. The user is able to traverse the steps by pointing and clicking a graphical representation of the overall process. At each step, the user answers questions and provides information related to that step. The program allows the user to skip between steps, repeat steps, and generally explore the problem space.

The program provides several pre-packaged example dilemmas for the students to experiment with and explore. One of the example dilemmas, involving an accident during the installation of a 330-foot tall antenna in Houston in the early 1980s that resulted in the deaths of several people, is represented in great detail, including video clips showing the participants and the accident as it occurred.

5.3.2. The Dax Cowart CD

The Dax Cowart CD is an interactive, multimedia program designed to explore the practical ethics issue of a person's right to die [Cavalier and Covey, 1996]. The program focuses on a single case, that of Dax Cowart, a victim of severe burns, crippling injuries, and blindness who insists on his right to die throughout enforced treatment for his condition. The central issue of the case is whether Dax should be allowed to die and what reasons support or conflict with that decision. The program presents video clips of interviews with Dax's doctor, lawyer, mother, nurses, and Dax himself to allow the user to experience the issue from different viewpoints. The program also presents clips of Dax's painful burn treatment to provide an intimate sense of his predicament.

The program operates in two modes: (1) a guided mode in which questions are posed in Socratic style to challenge the user to consider the relevant facts, moral issues, and viewpoints and (2) an exploratory mode in which the user can freely view video clips and investigate the basic ethical issues involved with the case, including the quality of life, patient rights, and the conflicting obligations of medical personnel. In the guided mode, the user is periodically asked to make judgments on whether Dax's request should be granted and, dependent on how one answers, the program branches to present information and viewpoints that may cause reconsideration of that judgment.

While neither the Ethos System nor the Dax Cowart CD are directly comparable to SIROCCO, given the different goals of the programs, it is instructive nevertheless to discuss how they differ. Both the Ethos System and the Dax Cowart CD present a much deeper and detailed view of the ethics domain to the user than does SIROCCO. Both programs thoroughly cover at least one problem and provide video clips to viscerally engage the user in that problem. Each program is intended to instill a deep appreciation of the complexities of ethical decision making. On the other hand, neither the Ethos System nor the Dax Cowart CD involves any intelligent processing. All of the steps and displays of both systems are effectively "canned." SIROCCO's retrieval and analysis functionality is far more complex than the processing of either of these pedagogical programs. Also, SIROCCO covers a much wider range of issues and problems than do either of the pedagogical programs. This is largely because the knowledge engineering involved in representing a single case in both the Ethos and Dax programs is enormous, i.e., one must create (or acquire) and integrate video clips and copious supporting information into an ethics problem definition. Also, while the Ethos System and the Dax Cowart CD are focused on iterative user interactivity, SIROCCO is intended to perform the noninteractive and focused task of retrieving the relevant codes, cases, and operationalizations of a given case description.

Finally, while both the Ethos System and the Dax Cowart CD focus on one case or one case at a time, each could benefit from an ability to support users in finding similar cases. One approach would be to "hard wire" links to similar cases, but a more powerful and flexible approach would be to provide an intelligent retrieval capability. SIROCCO provides such a capability and could make a useful adjunct to either (with the appropriate case base, interface, pedagogical curriculum, guidance, and some feedback).

6. Conclusion

This dissertation has explored the problem of applying abstract, open-textured rules to specific fact situations. Such rules cannot be applied without knowledge that "bridges the gap" between the rules and the concrete facts. In some domains, like the law, ethics, and policy decision making, there are often no authoritative or readily-available intermediate rules that can be used to bridge this gap. Nevertheless, the decision makers in these domains do make and record decisions, and by using operationalization techniques, some extensional connections between abstract rules and scenarios may be discerned and applied to new fact situations for the purposes of retrieval.

I have proposed, developed, and empirically tested a computational model, SIROCCO, that implements a set of operationalization techniques, which help bridge the gap between abstract rules and facts in the domain of engineering ethics. The computation model tested two theses. First, it tested whether a subset of the observed operationalization techniques, those that are most objectively verifiable, allow a computation model to make accurate predictions of the principles and past cases that are likely to be relevant in the analysis of new cases. Second, it tested the secondary thesis that temporal knowledge can also be used to support the computational model in making accurate predictions.

The experiments provided strong evidence that the primary thesis is well founded. SIROCCO performed significantly better than several competitor models, including a full-text retrieval system and a version of SIROCCO that did not employ operationalization techniques. In addition, it was shown that augmenting the information retrieval system with an operationalization technique improved the performance of that system. The experiments did not, however, support the secondary thesis. A test comparing the use of SIROCCO with and without temporal knowledge resulted in negligible difference between the performance of the two models. This was surprising in light of the analysis of the NSPE BER cases (reported in Chapter 2) and the fact that some actual examples of SIROCCO in action (see, for instance, Section 3.3) indicated that temporal relations do make a difference. On the other hand, at least SIROCCO's temporal knowledge did not prevent good retrieval. Finally, the experiments showed that SIROCCO's computational characteristics are satisfactory, and that the system can be expected to scale up to 2,000 cases and still perform adequately.

It has also been shown in the dissertation that SIROCCO provides advantages that are not empirically quantifiable. For instance, the program is able to display the results of structural mappings between an input target case and the relevant Instantiations it retrieves. This information is potentially quite useful to a user, as it indicates the Facts (i.e., the individual steps of a Fact Chronology) that may be relevant in assessing the case. The program is also capable of displaying additional suggestions that are based on the application of the operationalization techniques that were not directly involved in the selection of codes and cases.

In this chapter, I discuss the contributions of the research, the lessons learned in building the computational model, and future directions of the work.

6.1. Contributions of the Dissertation Research

The contributions of the research presented in this dissertation are summarized below and the following sections discuss each contribution in turn.

Contribution #1: The identification and cataloging of a set of operationalization techniques, through an empirical analysis of a set of recorded decisions.

Contribution #2: The implementation and testing of a computational model that uses the operationalization techniques to support retrieval and analysis.

Contribution #3: The development and application of a limited, but expressive, case-representation language to represent the detailed facts of a case.

Contribution #4: The successful application of the computational model for purposes of retrieval to a wider range of cases than any predecessor interpretive CBR system.

Contribution #5: The implementation and testing of a computational model that uses temporal knowledge to support retrieval and analysis.

6.1.1. The Empirical Identification of the Operationalization Techniques

There has been little empirical work in AI that investigates the role principles play in computational models of decision making. As has been discussed, principles are abstract rules with typically no available intermediate rules to connect them to specific fact situations. However, humans manage to apply principles to fact situations, so it is useful to study and understand the techniques they employ to accomplish this. Understanding how to apply and evaluate principles is important if one wants to build computational models in weak analytic domains, such as ethics and the law.

Research in AI and Law has perhaps come the closest to addressing this problem. Various researchers have investigated the connection between legal theories, similar to principles in many respects, and concrete cases. For instance, Karl Branting investigated how to bridge the gap between legal theories and the specific facts of precedent cases [1993; 1994]. As the foundation for a proposed computational model, Branting employed the concept of *ratio decidendi*, a legal metatheory that determines how a precedent can be authoritative to subsequent cases. In BankXX [Rissland *et al.*, 1993, 1996], theories are part of an extensive legal network that is searched to develop arguments for one side or another in a legal dispute.

A key distinction of this dissertation is that it presents an explicit, structured, and comprehensive list of the techniques used to bridge the gap between abstract, open-textured rules and specific facts in a particular domain of interest. In previous work, various techniques have been discussed and implicitly "recorded," but no other project, to my knowledge, presents an explicit and structured list of empirically-observed techniques, such as reported here. In particular, I observed, analyzed, recorded, and cataloged the systematic reasoning techniques employed by an ethics review board in deciding 475 published engineering ethics cases. Although some of the identified techniques – for instance, *Define the Terms of the Code*, *Cite an Analogous Precedent* – should be recognizable to those who have studied the interaction between rules and cases in other weak analytic domains, such as the law, the identification of the full group of techniques, applied in various combinations in analyzing and resolving ethical dilemmas, is a research contribution. Further, some of the identified techniques, such as *Group Codes* and *Rewrite a Code*, are clearly not addressed in comparable work, see for instance the comparison of SIROCCO to GREBE (Section 5.1.2) and SIROCCO to BankXX (Section 5.1.3).

Finally, while this specific group of techniques was identified in the particular domain of engineering ethics – and in a particular engineering society within that domain – there is reason to believe that the operationalization techniques may have more general applicability. A more-comprehensive investigation into the general applicability of the techniques remains for future work; some preliminary ideas are discussed in Section 6.3.4.

6.1.2. The Implementation and Testing of a Computational Model That Uses Operationalization Techniques

Part of the process of identifying and cataloging the operationalization techniques involved understanding how the techniques could be computationally realized. Although the review board uses the operationalization techniques to resolve the ethical dilemmas and/or justify their conclusions, the goal for SIROCCO was to employ the techniques to *retrieve* relevant information, i.e., codes, past cases, and other facts, about a dilemma. Modeling the board's resolutions and justifications as operationalization techniques, and implementing them computationally, successfully supported retrieval. This attests to the central role of the operationalization techniques as organizing structures in understanding, analyzing, resolving, and arguing cases.

In essence, the operationalization techniques link and annotate the facts of a case with codes and past cases, thus providing "indices" for future use of the operationalizations. For instance, the designated Critical and Questioned Facts of source cases provide indices, at various levels of abstraction, for Stage 1's retrieval process. The Critical and Questioned Facts also focus the Stage 2 heuristic search. The links between grouped codes and cases, or between one code and another code that it overrides, allow the program to indirectly access (or index) citations that might be relevant in the context of a new case.

This knowledge representation allows SIROCCO to operate without the intermediate rules found in many domains. Instead of chaining from the bare facts to intermediate conclusions to final conclusions, as in a deductive approach, SIROCCO compares the bare facts of a target case to the important facts of past source cases, harvesting various operationalizations that are linked to those facts in the process. The program then evaluates the retrieved operationalizations to determine if they apply in the context of the new target case and reuses those that apply. The operationalizations thus applied are essentially extensional, rather than intensional (i.e., rule-like), ways to deal with open-textured terms and can be thought of as replacing the reasoning inherent in intermediate rules.

The development of SIROCCO and the dissertation project in general took very seriously the notion of operationalization and its role in reasoning. In particular, the operationalization techniques "make explicit … how a piece of knowledge was applied or what particular strategies for accomplishing a goal were used" and "they capture knowledge that might be too hard to capture in a general model." [Kolodner, 1993, p. 9]. SIROCCO represents an effort to take these general concepts and realize them computationally in a way that is extensional, yet detailed given a narrative representation of cases as sets of temporally ordered actions and events. In particular, SIROCCO provides a framework in which operationalization knowledge is made explicit and is demonstrated in a difficult reasoning task.

The execution of empirical tests to evaluate the contribution of the operationalization techniques was a key aspect of the dissertation project. As discussed above, comparing SIROCCO to a version of the program lacking virtually all of the operationalization techniques showed that the techniques make a significant difference in the retrieval of codes and cases.

Further, comparing an information retrieval system, MG, to a version augmented by one of the operationalization techniques provided further evidence of the value of the operationalization techniques.

6.1.3. The Development and Application of a Limited, But Expressive, Case-Representation Language

SIROCCO's case-representation language (see Appendix A) and case-acquisition methodology (see Appendix B) are key contributions of this dissertation to the case-based reasoning literature. The SIROCCO research provides an example of how one can represent the actions and events in a complex narrative using a limited language, yet still successfully represent a relatively wide range of cases. Emphasis was placed on enabling human users to represent the bare facts of a situation, without greatly interpreting, abstracting, or explaining those facts. The goal was to keep the representation as close to the narrative presentation of facts as possible while pushing off abstractions and generalizations to the computational process. Thus, while SIROCCO reasons about action and event abstractions, via its Action/Event Hierarchy, the abstractions are not part of the given case representations. An ontology of engineering ethics was developed as an important part of the dissertation project. The ontology includes a representation of actions, events, actors, objects, and the temporal relationships between actions and events (see Section 3.1.2).

SIROCCO's language is limited in that the case enterer must conform to restricted ways of phrasing facts. For instance, the Fact Primitive "x criticizes y" is the only way to express a case fact that deals with an actor finding fault with, condemning, or denouncing another actor. The criticism may be verbal, written, or implied, yet these distinctions are not available to the case enterer. On the other hand, SIROCCO's language is fairly expressive in the sense that each Fact is essentially a sentence, i.e., with a noun-verb-noun type construct, and the Fact Chronology reads like the facts of a case, expressing the important details. Even when Facts can be described in different ways in SIROCCO's limited language, SIROCCO's matching criteria helps it deal with a lack of consistency in the representation of different cases. The Action/Event Hierarchy is a means for relating similar Facts to one another and the retrieval algorithm uses this structure to relate similar, yet imprecisely matched, cases.

SIROCCO's case-representation language diverges from most previous work in CBR with regard to situation descriptions. For instance, in PROTOS [Bareiss, 1989] and CASEY [Koton, 1989], two early diagnostic CBR systems that support extensive situation descriptions, situations were represented as collections of symptoms, patient histories, and results of lab tests. While this information is clearly the most relevant for solving diagnostic problems, the representations provided by these programs do not capture actions, events, and the chronology of those actions and events, as SIROCCO does. CATO [Aleven, 1997], discussed in Section 5.1.4, provides a different kind of situation description. CATO's factors are interpretations of the detailed facts with respect to whether those facts favor one side or the other in a legal dispute. However, CATO's representation scheme, as with PROTOS and CASEY, does not provide a representation of the actual events of a case.

SIROCCO's limited language is probably most similar to the languages of SWALE [Kass *et al.*, 1986, Leake, 1991] and GREBE [Branting, 1991]. Both of these programs also provide representations for actions and events. However, both also emphasize the representation of semantic explanation structures. That is, a case is represented not only by the actions and events of the scenario but also by explanations of those actions and events. As explained in Section 5.1.2, GREBE's explanations represent the reasoning (or justification) of a court in applying an abstract rule to a case. In SWALE, the explanations provide a more abstract or stereotypical description of a story, so that it may be more easily matched to (anomalous) cases in other domains[44].

While SIROCCO's *Code* and *Case Instantiations* also provide a form of explanation (or justification), through the linking of Facts to code and case citations, the justification representation in SIROCCO is, by design, less semantic than in GREBE or SWALE. In effect, SIROCCO's justifications are extensional representations for the cited codes and cases and allow the program to match these representations in future retrievals. Limiting the complexity and semantic content of the justifications, combined with SIROCCO's much more extensive set of actions and events, made it possible to represent many more cases over a wide variety of topics. Unlike either GREBE or SWALE, SIROCCO has been empirically shown, in this dissertation, to have the capability to successfully address a relatively wide variety of cases. On the other hand, because of the limited semantic content of its justifications, SIROCCO is not capable of producing detailed analyses of fact situations, in contrast to GREBE and SWALE.

SIROCCO's case-acquisition web site (www.pitt.edu/~bmclaren/ethics) was an attempt to address the knowledge-acquisition bottleneck that is inherent in any AI system that relies heavily on knowledge representation. SIROCCO's web site addresses the bottleneck in a number of ways. First, publishing the tool on the Internet provides easy and convenient access to many people who are geographically dispersed. Second, providing a user's guide and tutorial helps to

[44] Here I am referring only to justifications provided manually by a case enterer. There are, of course, other CBR programs that use program-generated justification structures to help guide reasoning [Veloso, 1992; Hinrichs, 1992].

clarify and explain the use of each of the language components. Third, supplying numerous example transcriptions (47 in total) that are heavily cross-referenced by the components they use, encourages a measure of consistency in the case representations. Fourth, providing a predefined template for transcribing the analysis representation of each case, available by download from the web site, eases the burden of manual case entry. Finally, supplying the complete lists of all the Actions, Events, Actors, Objects, and Time Qualifiers allows the case enterer to access any language component using simple point-and-click.

6.1.4. The Application of the Computational Model to a Wide Range of Cases

SIROCCO addresses a wide range of cases within the engineering ethics domain. As has been discussed, the program was developed with a focus on cases within the Selected Topics group (i.e., those cases that cite at least one code related to one of the following areas: public safety, confidential information, duty to employer, credit for engineering work, proprietary interests, and honesty in reports and public statements) of the engineering ethics domain. However, a fair number of cases from outside this core group, specifically 50 foundational cases and 14 trial cases, were also represented and tested in SIROCCO. The Non-Selected Topics cases involved issues such as conflicts of interest, criticizing other engineers, and competence. As evidence that SIROCCO is at least competent in addressing the Non-Selected Topics cases, the results of Experiment #1 showed that the difference between the mean F-Measures for Selected Topics trial cases and Non-Selected Topics trial cases was not great.

As discussed in Chapter 5, the capability to address a wide range of cases separates SIROCCO from earlier interpretive CBR systems such as HYPO [Ashley, 1990] and CATO [Aleven, 1997], which handled trade secrets cases exclusively; GREBE [Branting, 1991], which reasoned only about workers' compensation cases; CABARET [Rissland and Skalak, 1991], which processed only home-office tax deduction cases; and BankXX [Rissland *et al.*, 1996], which handled only Chapter 13 personal bankruptcy cases. Although topics in the legal domain and topics in engineering ethics are somewhat disparate, it seems apparent that the topical area of each of the earlier systems is much narrower than the full set of Selected Topics. If one considers that SIROCCO has also been shown to address cases *outside* the Selected Topics group, it is clear that SIROCCO provides a wider domain coverage, at least for purposes of intelligent retrieval.

This is not to say that the earlier systems were somehow representationally and architecturally incapable of handling a wider range of cases. For instance, Aleven claims that CATO "is not specific to trade secrets law" [1997, p. 41], and it is easy to see how CATO's Factor Hierarchy has the flexibility to be applied to topical areas outside of trade secrets law.

However, it is certainly true that none of the earlier interpretive CBR systems actually *demonstrated* coverage of a wider domain, as is done with SIROCCO in this dissertation (see Section 4.2.1 and the discussion of Figure 4-8).

The earlier systems were constrained in breadth of coverage by the fact that they were all intended to provide more detailed support for argumentation than SIROCCO does. This is one of the trade-offs inherent in widening domain coverage. That is, there is a natural tension between developing a representation that is deep in particular and specialized knowledge, for instance, of argumentation, and a representation that is more broadly applicable to a wider range of situations but is not as capable of providing deep and detailed analysis information.

SIROCCO's wider breadth of coverage is at least partly due to its language and case-acquisition tool. The language provides enough general-purpose primitives to allow it to be applied to a variety of engineering ethics scenarios. As mentioned above, the use of the tool supports a measure of consistency in representation that allows one to address a wider range of cases and matching criteria.

6.1.5. The Implementation and Testing of a Computational Model That Uses Temporal Knowledge to Support Retrieval and Analysis

The overall assessment of the NSPE BER foundational cases, from which SIROCCO's architecture was influenced and designed, appeared to indicate that temporal knowledge played an integral role in the way the board decided many of its cases. In fact, in Chapter 2, several examples were presented and discussed in which the review board used temporal knowledge to analyze and decide a case. In addition, an example of how temporal knowledge was actually used by SIROCCO to correctly abstain from suggesting a code was presented in Section 3.3. This example illustrated that temporal knowledge can make a difference in the computational model's retrieval and analysis.

Unfortunately, the experiments, specifically Experiment #3, did not support the thesis that temporal knowledge assists SIROCCO in making accurate predictions of the principles and past cases that are relevant to the analysis of new cases. Specifically, the experiment showed that there was a negligible difference in the accuracy of standard SIROCCO run over the 58 trial cases versus a version of SIROCCO that did not employ temporal knowledge over the same cases. Again, at least it did not make things worse, however.

A number of reasons were posited as to why SIROCCO's temporal knowledge did not make a difference in the experiment. First, there may have simply been too many trial cases that did not involve temporal considerations. Second, uniform characteristics of the Facts, such as their typical temporal location in a Fact Chronology or their *prima facie* connection to relevant codes, may have sabotaged the flexibility and power of SIROCCO's use of temporal knowledge. Third, inaccuracies in the temporal knowledge transcribed by the independent case enterers may have played a role. Finally, the specific representation and implementation of temporal reasoning within SIROCCO may have been the cause of this finding. For instance, the representational scheme of SIROCCO lacks a way for the case enterer to designate certain temporal relations as more important than other temporal relations. More than likely, some combination of all of these reasons underlie SIROCCO's failure.

SIROCCO represents one of the few reported attempts to factor temporal considerations into the retrieval process of a case-based reasoner. Case-based planning systems, such as CAPLAN/CBC [Muñoz-Avila and Huellen, 1995], PRODIGY/ANALOGY [Veloso, 1992], and CHEF [Hammond, 1989] involve temporal sequencing in the solutions they generate, store and retrieve, but the retrieval indices of such systems consist of elements such as goals, partial ordering between the goals, and initial state descriptions, not the temporal relations between individual steps, as in SIROCCO. Another way to describe this distinction is as follows: the solution sought by a planning system is a set of temporally related steps, while the solution sought be SIROCCO is an *analysis* of a *given* set of temporally related steps.

One existing CBR system that uses temporally related steps to assist in retrieval is BROADWAY, a world-wide web browsing advisor [Jaczynski and Trousse, 1998]. BROADWAY uses a current sequence of browsed pages as indices into a case base of previous browsing sessions, represented as time-extended situations, to advise a user on what to do next. This system uses much simpler temporal constraints (i.e., before, after) than SIROCCO, but a more general architecture that performs matching of the Allen primitives underlies the application [Jaczynski, 1997].

Vila and Yoshino have developed a formal model of temporal reasoning in the legal domain [1998]. Although their work is somewhat preliminary and not directly applied to case-based reasoning, it may provide a formal foundation for future interpretive CBR systems.

6.2. Lessons Learned and Limitations of SIROCCO

In this section, various lessons learned and limitations of the dissertation project are discussed.

As discussed above, a central contribution of the SIROCCO project has been the use of a limited language for representing cases. However, the use of such a language does not come without problems. For instance, as reported in Chapter 4, some of the NSPE BER cases simply could not be transcribed into the language. One of the key reasons for this difficulty was that some fact situations did not involve only or primarily actions or events. For instance, several cases focused on the language of a contract between an engineer and a client or an engineer and his employer, while other cases dealt with the language used in engineering advertisements. Since these types of situations do not involve actions or events (other than, perhaps, the writing of the document or advertisement), it was not clear how they should or could be modeled in EETL. Clearly, a language that better accounts for fact situations that do not contain actions and events would have been advantageous. For instance, the language could have been supplemented with components describing what documents say, perhaps using some of the language components (i.e., speech acts) described by Anne Gardner [1987].

Another problem, mentioned earlier, is the difficulty for case enterers to correctly define the temporal relationships between actions and events. For this issue, it would have been helpful to have an interactive visual display depicting the current temporal relationships of a case, similar to that depicted in Figure 3-9. The visual display would automatically update, using the TIMELOGIC propagation scheme as its basis, when the case enterer adds a new Fact or modifies an existing Fact. Such a function might have improved the quality and accuracy of the Time Qualifiers submitted by the case enterers and made a difference in Experiment #3, which tested the effects of temporal knowledge.

In general, the case-acquisition scheme could have been more robust. In particular, the template used for entering cases could have been made interactive, automatically correcting syntax and simple semantic errors, allowing the user to move Facts from one position to another, providing access to all of the language components, and so on. Also, a capability to iteratively define a case and retrieve relevant codes and cases based on partial case descriptions could be a useful addition to both the case-acquisition process and to end-user retrievals. Such a capability would provide the user with a clear sense of how the addition of individual Facts, or groups of Facts, affect SIROCCO's retrieval and analysis process.

These "tool" issues highlight one of the key characteristics of a representation-intensive approach: a heavy reliance on the ability of humans to transcribe textual cases into the limited language. A fundamental question logically follows from this: Is a representation-intensive approach viable, considering the advances in representation-free full-text retrieval? There is no arguing the fact that representing cases is labor intensive. The case enterers on the SIROCCO project reported that it took, on average, 2 to 3 hours to transcribe a single source case (i.e., representing both a Fact Chronology and the review board's analysis) into the form usable by the program[45]. In addition, the imprecision that is introduced in such an interpretation process, due both to human error and to limitations in the expressiveness of the language, is cause for concern. This is especially a concern because a full-text retrieval approach does not require any representation and does not introduce the extra layer of subjective interpretation.

On the other hand, there are clearly advantages to the manual representation approach. For instance, this dissertation provides compelling evidence that the manual representation approach can result in more-accurate performance. In particular, it was shown that SIROCCO significantly outperformed MG, the pure full-text retrieval system, and even outperformed EXTENDED-MG, the full-text retrieval system augmented with an operationalization technique. In addition, the explanatory capabilities of SIROCCO, which rely on the manual representation of both the facts and the rationales, provide potentially relevant and important information that is simply not available from a full-text retrieval system. Finally, SIROCCO, as currently constituted, may provide a better fit for its ultimate goals as a retrieval tool for an engineer and as a retrieval component in an intelligent tutoring system. As a retrieval tool for an engineer, SIROCCO's explanatory capabilities are critical. It is hard to imagine a user putting much faith in such a system if it provides no explanation of its function. In a tutoring system, the goal is for engineering students to use SIROCCO interactively to analyze and explore cases. Asking the student to provide manual case representations is intended, in part, to have pedagogical value. In particular, thinking through the actions and events of a scenario, and the chronology of those actions and events, may help the student to better understand the relevant facts. SIROCCO's explanatory capabilities are also likely to be critical for tutoring.

In general, the answer to the question of whether a representation-intensive approach is viable, particularly in comparison to full-text retrieval, likely rests on the purposes intended for the program. If one desires a system that can simply provide a quick possible answer, with minimal effort on the part of the user, then full-text retrieval is clearly a better choice. If, on the other hand, one desires a system that can explain its actions and provides an opportunity for the user to analyze and dissect a problem into its constituent steps, a representational approach to the problem, such as that embodied by SIROCCO, may be preferable.

---

[45] Note, however, the case enterers also said that the majority of their time was spent transcribing each case's *analysis representation*. Representing the facts as a Fact Chronology did not take nearly so long. A typical user would not need to represent the analysis, only the facts. Thus, source cases typically take much longer to transcribe than target cases. The analysis representations probably take more time because (a) the analysis portion of the case texts are typically much longer than the fact descriptions, (b) the analysis representations are more of an abstraction of the text and, therefore, require more time to interpret, and (c) the board's analyses are much less consistent than the fact descriptions from case-to-case.

It may be feasible to combine the representation-intensive and full-text retrieval approaches to take advantage of the respective benefits of each. For instance, both approaches could be applied to a target case and the results of one method, say full-text retrieval, could be used to verify the results of the other. Since it is relatively cheap to use full-text retrieval, not much additional human or computing resource would be expended, but an additional validating data point would be gained. And, by deploying the representation-intensive approach in such a scenario, one is able to obtain explanations of the retrieved codes and cases.

Recent work in textual CBR also shows promise in this area. Steffi Brüninghaus and Kevin Ashley have developed a system called SMILE that takes a small set of manually represented legal cases and automatically assigns indices for a larger set of textual cases [1999]. SIROCCO's ontology, the hierarchies and database of represented cases provide a valuable resource for experimenting with these techniques in the engineering ethics domain.

Another lesson learned in the course of the dissertation work concerns the viability of a two-phase algorithm. Karl Branting identified a potential problem with two-phase algorithms [1992], by evaluating several programs and building his own method, called match refinement by structural difference links (MRSDL). In particular, he points out that the success of such an approach hinges on the size of the set of initial candidates. If the size of the set is too small, then similar cases may not be found. On the other hand, if the size of the set is too large, then the computational expense of structural mapping increases prohibitively.

In SIROCCO's architecture this problem does not appear to exist. Recall that the results of Experiment #5 showed that $N = 6$ (SIROCCO's initial candidates list) was the optimal value for selecting initial candidates. N > 6 not only increased run time but also decreased SIROCCO's accuracy. This occurs for a couple of reasons. First, SIROCCO, unlike other two-stage algorithms, such as MAC/FAC [Forbus *et al.*, 1994] and ARCS [Thagard *et al.*, 1990], is not designed to find the single best analog. Rather, it uses its selection heuristics to suggest codes and cases from a *set* of well-matched analogs (i.e., the initial candidates list). What is critical is that there are at least reasonable matches in the candidates list and that there is some agreement in citations with those candidates. Second, as was illustrated in Section 3.1.5, SIROCCO has selection heuristics that allow it to cite cases (and codes cited by those cases) that were *not* in the original list of candidate cases. This was shown in the example of Chapter 3 when Case 84-5-1 was cited, even though it was not in the original list. Thus, the original candidates list is not as essential to SIROCCO's selections as it is to other two-stage algorithms.

Finally, the results of the temporal experiment (i.e., Experiment #3) do not dissuade me from the potential importance of temporal relations in the retrieval process of a case-based reasoner.

Most previous work in CBR has either assumed instantaneous situation descriptions, represented as attribute-value pairs, or abstracted away the temporal information, for instance, by defining factors or other structures that summarize situation characteristics. However, the world that we live in is temporally continuous and a great number of interesting problems to address require some understanding of temporal sequence and overlap of actions and events. For instance, in the domain of engineering ethics, one may have a moral obligation to report a public safety hazard, but not before one learns of the risk. Important and interesting problems such as this make it hard to imagine the field of case-based reasoning advancing without addressing the issue of how temporal knowledge can be incorporated into a computational model.

6.3. Future Research Directions

In this section some possible future research directions are discussed.

6.3.1. Improving the Performance of SIROCCO

There are a variety of changes and enhancements that could be made to SIROCCO to potentially improve its performance. Possibilities arise from the discussion in Section 3.1.2, in which it was mentioned that a number of the components of SIROCCO's knowledge representation were not used by the retrieval and analysis algorithms. Finding a way to gainfully exploit at least some of the unused components might make a difference in the program's accuracy and qualitative suggestions. Most of the unused components were not used because it was not obviously clear how they might be incorporated into SIROCCO's algorithm. However, it might be worthwhile to experiment with some possibilities. For example, distinctions in the importance of Facts, represented by the Fact Modifiers "partially," "substantially," "limited," and "extensive," could be incorporated into SIROCCO's Stage 2 algorithm to strengthen or weaken structural mappings. Distinctions in the importance of *Code* and *Case Instantiations*, represented by the importance levels "more importance" and "less importance," could influence SIROCCO's selection heuristics in proposing codes, cases, and additional suggestions. Performing at least limited structural mapping between Internal Fact-Phrases, i.e., those fact structures that are nested within Facts, may also provide some benefit.

Clearly, the program did not meet expectations regarding the secondary thesis, so a number of changes relevant to temporal reasoning could also be explored. Clues to what could be changed are discussed in Section 4.3.2. For instance, currently SIROCCO treats all instances of Time Qualifiers as equal. However, just as particular Facts of a chronology are more important than others, it is also true that particular Time Qualifiers are more important than others. Thus, a scheme in which Time Qualifiers can be designated as more crucial, leading to greater or lesser match scores, might be helpful. Another possible change related to weighting the Time Qualifiers would involve the development of a capability to override SIROCCO's default approach of accumulating evidence in support of particular codes and cases. In particular, there may be instances in which a temporal match (or mismatch) completely outweighs the combined evidence from a variety of other case or code matches. Thus, the development of an exception handler, or specialized heuristics, within SIROCCO's Analyzer might be fruitful.

6.3.2. Using SIROCCO to Support Intelligent Tutoring

One of the stated long-term goals of this research is to incorporate SIROCCO into an intelligent tutoring system as a retrieval component. SIROCCO could be valuable in a number of ways.

First, students need to learn to read cases carefully. The exercise of representing a narrative case into an ordered set of actions and events would likely help the student to focus reading. In addition, by studying the general types of available Fact Primitives – a necessary activity in successfully transcribing cases – the student may gain some understanding of the types of facts that matter.

Second, a problem represented as a Fact Chronology could be used to retrieve relevant codes and past cases. Perhaps the student could make his or her own suggestions of relevant codes and cases before invoking SIROCCO and then compare those suggestions to SIROCCO's retrieved codes and cases.

Third, SIROCCO's explanatory capabilities could help focus students on the Critical Facts of a given target case. By examining the Fact mappings displayed when SIROCCO compares the target to *Code* and *Case Instantiations* of source cases, the student could discern the most relevant Facts. SIROCCO's additional suggestions could also be helpful in this regard. For instance, if the program suggests that a pair of codes might conflict in the context of the present case, the student could examine the Facts associated with those conflicting codes to determine which are most central to the conflict.

Finally, SIROCCO could be used to support a line-drawing exercise. Line-drawing is a moral analysis technique in which scenarios are situated on a continuum ranging from "clearly unethical" to "clearly ethical." [Harris *et al.*, 1999, p. 60]. The idea is to determine where an input case lies on this continuum by comparing and contrasting it to other cases on the continuum. The student could use SIROCCO's code and case suggestions to help in placing an input case, and variations of that case, on a continuum and thereby gain a greater understanding of how the case and variations differ. By modifying the Facts in the Fact Chronology of the input case, i.e., adding or deleting Facts or changing Time Qualifiers, a student could perform a series of retrievals and observe how the possibly relevant codes, cases, and suggestions change. (The proposed iterative case-description capability, discussed in Section 6.2, could support the rapid definition of a group of variant cases.) Although SIROCCO does not suggest a conclusion (i.e., ethical, unethical, or a qualified "middle" solution) to help with this process, the retrieved citations would provide clues on where each variation should reside on the continuum.

6.3.3. Developing Tools for Mapping Case Texts to the Ethics Transcription Language

The usability of SIROCCO would improve dramatically if there were a way to more easily provide target cases to SIROCCO, i.e., to map case texts into ETL. As has been discussed, the process of interpreting a source case and transcribing that case into EETL is time-consuming. Transcribing a target case, i.e., the facts only, is significantly less time-consuming than transcribing a source case, but the effort involved is still non-trivial. Although on one hand this is useful, as it encourages the user, perhaps a student, to think carefully about the relevant facts of a case, on the other hand, it is somewhat tedious. A middle-ground solution might be to provide tools that perform some automated translation of the text with the user having to edit the translations into a final form.

For instance, one could imagine a tool that takes every sentence of a case text and searches SIROCCO's Action/Event Hierarchy for the Fact Primitive that most closely matches the primary verb of that sentence. A grammar checker, combined with a thesaurus, could handle such a task. The resulting set of matching Fact Primitives would then be presented to the user as an ordered list of Facts with as much Actor and Object information filled in as the system is able to discern. Of course, such output would be only a rough approximation of an appropriate Fact Chronology. The user would still have to correctly designate Actors and Objects and decide which Facts are worthy of inclusion in the chronology, which Facts should be Internal Fact-Phrases, etc. But certainly such a tool could save considerable time in creating target cases to provide to SIROCCO.

Textual CBR, discussed above, could be of benefit in this area. SIROCCO's knowledge representation should help research in building programs that deal directly with textual ethics cases, just as CATO's representation helped SMILE address textual legal cases [Brüninghaus and Ashley, 1999].

6.3.4. Evaluating the Generality of SIROCCO and the Operationalization Techniques

A final research direction is to evaluate the generality of SIROCCO and the operationalization techniques by applying the computational model to different domains. There are, of course, constraints on the types of domains in which the model would be applicable. For instance, a candidate domain would be required to have, at minimum: (1) a documented set of codes, theories, principles, or policies and (2) a recorded set of cases, each of which includes a fact description, at least one question about the fact description, an outcome, and an analysis of the fact description that includes citations to the codes, theories, principles, policies, and other cases. One would also need to verify that the operationalization techniques discussed in this dissertation are used in the new domain. Finally, the engineering ethics ontology would need to be extended to include new Fact Primitives, Actors, and Objects that are relevant to the new domain. An estimated one-third to one-half of the existing components, those that are not specific to engineering, would likely carry over to most domains.

A natural domain to address is the law. In particular, might the list of operationalization techniques and tools for knowledge representation generated during this dissertation project provide any new insights into or techniques for retrieval of legal texts? Clearly the legal domain satisfies the minimum applicability constraints because it provides published laws, theories, and decided legal cases. Many of the operationalization techniques also appear to be used in legal reasoning. For instance, there is little doubt that *Code* and *Case Instantiations* are relevant, as it is institutionalized within the legal profession to interpret the facts of a new case in light of established laws and precedent cases. If one or more subdomains of the law were chosen as the focus, such as worker's compensation, tax law, and trade secrets, it might be feasible to extend ETL appropriately to cover a wide number of fact situations.

A more unusual domain to address is "social etiquette" or "pop psychology." Take, for instance, the published rules of etiquette and the many letters answered by "Miss Manners" [Martin, 1998] over the years or the common sense rules for dealing with family and personal conflicts in "Dear Abby" [Van Buren, 1989]. Trivial subdomains perhaps, but these are a major step toward the commonsense domains that AI has so long hankered to conquer. Arguably, these data sources satisfy the minimum applicability constraints. Usually, Miss Manners explicitly refers to specific rules of etiquette in her responses to letters, and she clearly cites, at least implicitly, the rules of etiquette and at least occasionally references her past letters and responses. Abby is somewhat less specific in referencing rules. Whether the operationalization techniques are applicable in these domains is an empirical question, but there is at least some reason to believe that they are. Clearly Miss Manners interprets the rules of etiquette for her readers. Often the letters center on a conflict of the rules, and Miss Manners suggests resolutions to those conflicts. Probably the most difficult task in applying SIROCCO to this domain would be in extending ETL. The types of personal situations addressed in such commonsense domains are clearly quite variable, thus leading to the need for a wide range of actions, events, actors, and objects.

A more serious and potentially applicable domain is government and political decision making. For instance, Neustadt and May explore how politicians sometimes employ historical precedence and analogy in their decision making [1986]. An example they provide is the 1962 Cuban missile crisis and how President Kennedy and his advisors discussed strategy, in part, by arguing about whether a decision to abruptly attack Cuba could be considered analogous to Japan's bombing of Pearl Harbor during World War II. Holyoak and Thagard discuss a more recent example of this type of political reasoning in which President George Bush justified the 1991 attack on Iraq and Saddam Hussein by evoking an analogy to World War II Germany and Hitler [1995, p. 101-109]. The difficulty in applying SIROCCO to such a domain would be, first, to identify the general policies and rules that guide such decision making, especially over time, by different administrations, and even by different governments and, second, to acquire enough documented cases and analyses that clearly involve those policies and rules. Finally, as with the other possible domains, it would be necessary to verify that the operationalization techniques are, in fact, applied in such a domain.

Appendices

Appendix A: SIROCCO's Complete Knowledge Representation

A.1. The Ethics Transcription Language (ETL)

A.1.1. The Action/Event Hierarchy

The Action/Event Hierarchy for the engineering ethics domain is shown in its entirety below. Fact Primitives are displayed in italics, prefaced with two asterisks (**). If a Fact Primitive has multiple forms, all forms are shown, separated by vertical bars (|). Fact abstraction categories are shown in roman type, prefaced with the hierarchy level in angle brackets. A Fact Primitive may reside in multiple fact abstraction categories.

- <1> Obtain-Maintain-or-Alter-Engineering-Employment-or-Work-Status
    - <2> Pursue-Professional-Engineering-Employment-or-Services
        - <3> Seek-Employment-or-to-Employ
            - <4> Seek-Employment-By-Engineer
                - ** *seeks-employment-in*
                - ** *seeks-employment-with*
                - ** *is-successful-finding-work-in*
                - ** *writes-resume-to-emphasize-experience-in*
                - ** *writes-resume-listing-the-achievement*
            - <4> Seek-to-Employ-By-Employer
                - ** *seeks-to-employ*
                - ** *seeks-to-employ-an-engineer-in*
                - ** *conducts-an-extensive-employment-search-for*
        - <3> Seek-to-Provide-or-Obtain-Professional-Engineering-Services
            - <4> Seek-to-Provide-Professional-Engineering-Services
                - ** *offers-services-to* | *offers-services-to-for*
                - ** *submits-a-proposal-to-for*
                - ** *develops-proposal-for*
            - <4> Seek-to-Obtain-Professional-Engineering-Services
                - ** *seeks-to-hire-the-services-of* | *seeks-to-hire-the-services-of-for*
                - ** *seeks-to-hire-the-services-of-an-engineer-in*
                - ** *develops-request-for-proposals-for*
                - ** *requests-proposals-for* | *requests-proposals-from-for*
    - <2> Handle-an-Offer-of-Professional-Engineering-Employment-or-Services
        - <3> Offer-Accept-or-Refuse-Employment
            - ** *is-offered-employment-by*
            - ** *accepts-an-offer-of-employment-from*
            - ** *asks-to-start-employment-with-at-time*
        - <3> Accept-or-Refuse-Offer-to-Perform-Professional-Engineering-Services
            - ** *agrees-to-perform-services-for*
    - <2> Work-as-an-Employed-or-Contract-Professional-Engineer
        - ** *employs* | *employs-as*
        - ** *hires-the-services-of* | *hires-the-services-of-as* | *hires-the-services-of-for*
        - ** *is-retained-by-as-a-technical-consultant-for-a-legal-or-arbitration-proceeding*
    - <2> Stop-Work-as-an-Employed-or-Contract-Professional-Engineer
        - <3> Suspend-Professional-Employment

** *is-on-paid-leave-from*
** *is-on-unpaid-leave-from*
<3> Terminate-Professional-Engineering-Employment-or-Services
<4> Terminate-Employment-By-Employee
** *resigns-employment-with*
** *retires-from*
** *submits-resignation-to*
<4> Terminate-Employment-By-Employer
** *terminates-employment-of*
** *is-informed-of-termination-by*
<4> Inform-of-Termination-of-Employment
** *submits-resignation-to*
** *is-informed-of-termination-by*
<4> Terminate-Services-By-Engineer-or-Firm
** *completes-services-for*
** *terminates-services-for*
<4> Terminate-Services-By-Client
** *terminates-the-services-of*
** *is-terminated-by-as-a-technical-consultant-for-a-legal-or-arbitration-proceeding*

<1> Strike-or-Protest-against-an-Employer
** *participates-in-a-strike-against*
** *participates-in-a-protest-regarding*

<1> Possess-an-Engineering-Qualification
<2> Specialize-or-Provide-Experience-in-a-Particular-Field-of-Engineering
** *specializes-in*
** *has-experience-in*
** *is-a-key-engineering-employee-of*
<2> Work-as-a-Registered-Engineer
** *is-a-registered-engineer* | *is-a-registered-engineer-in*
<2> Possess-an-Academic-Engineering-Degree
** *has-a-Bachelors-degree* | *has-a-Bachelors-degree-in* | *has-a-Bachelors-degree-from* | *has-a-Bachelors-degree-in-from*
** *has-a-Masters-degree* | *has-a-Masters-degree-in* | *has-a-Masters-degree-from* | *has-a-Masters-degree-in-from*
** *has-a-professional-degree* | *has-a-professional-degree-in* | *has-a-professional-degree-from* | *has-a-professional-degree-in-from*
** *has-a-PhD-degree* | *has-a-PhD-degree-in* | *has-a-PhD-degree-from* | *has-a-PhD-degree-in-from*

<1> Have-Problems-with-Engineering-Registration
** *has-engineering-registration-revoked-because*
** *warns-that-engineering-license-could-be-revoked-if*

<1> Participate-in-an-Engineering-Society
** *is-engineering-society-member-of*
** *is-engineering-society-president-of*

<1> Work-or-Study-in-an-Academic-Environment
<2> Work-as-a-Student-in-an-Academic-Environment
** *is-an-engineering-graduate-student* | *is-an-engineering-graduate-student-at*
** *takes-an-academic-course-with*
<2> Work-as-a-Teacher-in-an-Academic-Environment
** *is-an-engineering-faculty-member* | *is-an-engineering-faculty-member-at*

<1> Start-or-Own-an-Engineering-Business-or-Joint-Venture
<2> Own-an-Engineering-Company-or-Joint-Venture
** *founds-the-company*
** *buys-the-company*
** *owns-the-company*
** *keeps-the-company-name*
** *forms-a-joint-venture-with*
<2> Propose-or-Start-a-Joint-Venture
** *proposes-a-joint-venture-with*
** *forms-a-joint-venture-with*

<1> Declare-Personal-or-Company-Bankruptcy
** *declares-bankruptcy*

<1> Perform-Professional-Engineering-Services-or-Work
<2> Perform-Technical-Engineering-Work
<3> Perform-Miscellaneous-Engineering-Work
** *provides-engineering-services-on* | *provides-engineering-services-on-for*
** *in-his-capacity-as-takes-the-action*
<3> Perform-Engineering-Design-or-Analysis-Work
** *designs*
** *redesigns*
** *reviews-and-analyzes*
<3> Perform-Engineering-Analysis-Review-or-Testing-Work
** *reviews-and-analyzes*
** *inspects*
** *performs-test* | *performs-test-on*
** *collects-test-samples-from*
** *records-the-existence-of*
<3> Propose-an-Engineering-Solution
** *proposes-the-solution* | *proposes-the-solution-as*
<3> Perform-Material-Removal
** *removes-material-from*
<3> Perform-Engineering-Approval-Work
** *approves-the-plan*
** *signs-the-plan-or-report*
** *issues-permit*
<2> Write-an-Engineering-Related-Document-Article-or-Paper
** *writes-paper/article* | *writes-paper/article-about*
** *publishes-in*
** *includes-the-information-in*
<2> Engage-in-Negotiations-for-Engineering-Work
<3> Negotiate-for-Engineering-Work
** *negotiates-with-for*
<3> Terminate-Negotiations
** *reaches-a-negotiating-impasse-with*
** *terminates-negotiations-with-regarding*
<3> Assist-in-Negotiations
** *offers-to-mediate-the-dispute-between-and*
** *mediates-the-dispute-between-and*
<2> Develop-or-Request-an-Engineering-Proposal
<3> Develop/Submit-an-Engineering-Proposal
** *develops-proposal-for*
** *submits-a-proposal-to-for*

<3> Request-an-Engineering-Proposal
** *develops-request-for-proposals-for*
** *requests-proposals-for* | *requests-proposals-from-for*
<2> Manage-in-an-Engineering-Environment
<3> Manage-People-in-an-Engineering-Environment
** *supervises*
** *heads-the-company*
** *owns-the-company*
<3> Manage-Projects-in-an-Engineering-Environment
** *is-responsible-for*
** *heads-the-company*
** *owns-the-company*
<2> Perform-Engineering-Research
<3> Publish-Engineering-Research
** *performs-research*
** *writes-paper/article*
** *publishes-in*
<3> Deal-with-Engineering-Research-Data
** *finds-research-data*
** *is-supportive-research-data-for*
** *includes-the-information-in*
<2> Advertise-Solicit-or-Market-an-Engineering-Product-Project-or-Work
<3> Market-or-Advertise-an-Engineering-Product
** *markets-products-to*
<3> Solicit-Engineering-Projects-or-Work
** *advertises-or-solicits-engineering-business-using*
** *offers-services-to* | *offers-services-to-for*
<3> Organize-Deliver-or-Attend-an-Engineering-Event-or-Seminar
** *organizes-event*
** *invites-to-event*
** *carries-out-event*
** *provides-as-part-of-the-event*
** *attends-event*
<3> Receive-Publicity-for-Engineering-Services
** *receives-favorable-publicity-for*
<2> Enter-an-Engineering-Design-Competition
** *enters-design-competition-using*
** *wins-design-competition*
** *enters-design-competition-against*
<2> Manufacture-Construct-or-Ship-an-Engineering-Artifact
<3> Manufacture-or-Construct-an-Engineering-Artifact
** *manufactures*
** *constructs*
<3> Ship-an-Engineering-Artifact
** *ships-to*

<1> Perform-Nonprofessional-Work
** *performs-nonprofessional-duties-for*

<1> Engage-in-a-Professional-Engineering-Relationship
<2> Engage-in-an-Adversarial-Engineering-Relationship
** *are-competitors*
<2> Engage-in-a-Collaborative-Engineering-Relationship
** *is-a-member-of*

<1> Engage-in-Civic-Activity
** *participates-in-a-protest-regarding*
** *petitions-for*

<1> Engage-in-Political-Activity
<2> Run-For-Or-Hold-a-Political-Office-or-Appointment
<3> Run-For-Political-Office
** *runs-for-political-office* | *runs-for-political-office-in*
** *are-competitors-for-the-same-political-office*
<3> Hold-Political-Office
** *is-an-elected-official-of*
<3> Hold-Political-Appointment
** *serves-as*
<2> Terminate-Political-Office-or-Appointment
** *ends-political-service-for*
<2> Give-a-Political-Contribution
** *is-asked-for-a-political-contribution-by* | *is-asked-for-a-political-contribution-of-by* | *is-asked-for-a-political-contribution-for-by*
** *gives-a-political-contribution* | *gives-a-political-contribution-of*
<2> Set-Limits-on-Political-Contributions
** *is-the-maximum-possible-political-contribution*

<1> Engage-in-Legal-or-Arbitration-Activity
<2> Initiate-Legal-or-Arbitration-Proceedings
** *calls-a-hearing-regarding*
** *petitions-for*
** *files-a-lawsuit-or-arbitration-action-against* | *files-a-lawsuit-or-arbitration-action-against-because*
<2> Reach-a-Conclusion-in-Legal-or-Arbitration-Proceedings
** *loses-a-lawsuit-or-arbitration-action-against*
<2> Provide-Expert-Consultation-in-a-Legal-or-Arbitration-Proceeding
<3> Start-Expert-Consultation-in-a-Legal-or-Arbitration-Proceeding
** *is-retained-by-as-a-technical-consultant-for-a-legal-or-arbitration-proceeding*
<3> Terminate-Expert-Consultation-in-a-Legal-or-Arbitration-Proceeding
** *is-terminated-by-as-a-technical-consultant-for-a-legal-or-arbitration-proceeding*
<2> Provide-Testimony-in-a-Legal-or-Arbitration-Proceeding
<3> Provide-Expert-Testimony
** *agrees-to-provide-expert-testimony-for* | *agrees-to-provide-expert-testimony-for-regarding*
** *provides-expert-testimony-for* | *provides-expert-testimony-for-regarding*
<3> Provide-Witness-Testimony
** *agrees-to-testify-regarding*
** *testifies-that*
<2> Has-Legal-Representation
** *is-legally-represented-by*

<1> Give-or-Claim-Credit-for-Engineering-Work
<2> Give-Credit-for-Engineering-Work
** *gives-credit-to*
<2> Claim-Credit-for-Engineering-Work
** *claims-credit-for*

<1> Give-or-Accept-a-Gift
<2> Give-a-Gift
** *offers-to*

** gives-a-gift-of-to
** gives-a-political-contribution-to | gives-a-political-contribution-of-to
** gives-to
<2> Accept-a-Gift
** accepts-a-gift-of-from

<1> Give-Receive-or-Request-Remuneration
<2> Give/Receive-Remuneration
** pays-for
** offers-to-pay-for
<2> Request-Remuneration
** requests-payment-from-for
** requests-additional-payment-from | requests-additional-payment-from-for
<2> Reject-a-Request-for-Remuneration
** rejects-the-request-for-payment-from
** rejects-the-request-for-additional-payment-from

<1> Ascribe-Quality-to-Engineering-Work
** provides-high-quality-engineering-work
** provides-average-quality-engineering-work
** provides-low-quality-engineering-work

<1> Disclose-or-Withhold-Information
<2> Disclose-Information
** informs-that
** reminds-that
** disseminates-to
** is-permitted-to-disseminate-to
** claims-that
<2> Withhold-Information
** withholds-information-from-regarding
** provides-limited-information-to-regarding

<1> Request-or-Provide-Engineering-Advice
<2> Request-Engineering-Advice
** requests-engineering-advice-from | requests-engineering-advice-from-for
<2> Provide-Engineering-Advice
** provides-engineering-advice-to | provides-engineering-advice-to-regarding
** provides-a-favorable-recommendation-of | provides-a-favorable-recommendation-of-to | provides-a-favorable-recommendation-of-as
** provides-an-unfavorable-recommendation-of | provides-an-unfavorable-recommendation-of-to | provides-an-unfavorable-recommendation-of-because | provides-an-unfavorable-recommendation-of-to-because

<1> Request-Provide-or-Withhold-a-Recommendation
<2> Request-a-Recommendation
** requests-a-recommendation-regarding-from
<2> Provide-a-Recommendation
** provides-a-favorable-recommendation-of | provides-a-favorable-recommendation-of-to | provides-a-favorable-recommendation-of-as
** provides-an-unfavorable-recommendation-of | provides-an-unfavorable-recommendation-of-to | provides-an-unfavorable-recommendation-of-because | provides-an-unfavorable-recommendation-of-to-because
** recommends-products-for
<2> Withholds-Recommendation

** *withholds-recommendation-of* | *withholds-recommendation-of-to*

<1> Give-or-Receive-Criticism
<2> Criticize-or-Accuse
** *criticizes* | *criticizes-to*
** *accuses-of-unethical-behavior* | *accuses-of-unethical-behavior-because*
** *has-substantive-evidence-for-the-accusation*
<2> Receive-Reprimand-for-Wrongdoing
** *is-found-guilty-of-unethical-behavior* | *is-found-guilty-of-unethical-behavior-by* | *is-found-guilty-of-unethical-behavior-because* | *is-found-guilty-of-unethical-behavior-by-because*
** *is-publicly-reprimanded-for* | *is-publicly-reprimanded-by-for*

<1> Request-or-Order-Something
<2> Request-Something
** *asks-for*
<2> Order-Subordinate-to-Perform-Task
** *instructs-to*
<2> Refuse-Request
** *refuses-the-request-by-to*
** *objects-to*

<1> Deal-with-Disputes-or-Contentious-Issues
<2> Deal-with-Agreements-or-Disagreements
<3> Agree-About-a-Fact
** *consents-to*
** *agrees-with-regarding*
<3> Disagree-About-a-Fact
** *objects-to*
<2> Deal-with-a-Dispute
<3> Apologize-Regarding-Dispute
** *apologizes-to-regarding*
<3> Mediate-Dispute-Between-Engineers
** *offers-to-mediate-the-dispute-between-and*
** *mediates-the-dispute-between-and*

<1> Deal-with-Official-Policies-or-Written-Agreements
** *has-a-policy-that*
** *has-a-policy-that-may-not-provide-engineering-services-on-projects-to-be-approved-by*
** *has-an-agreement-with-prohibiting-work-for-other-clients*
** *signs-the-agreement-with*

<1> Relate-Engineering-Artifacts-to-One-Another
<2> Relate-Similar-or-Encompassing-Engineering-Artifacts
** *are-largely-identical-to-one-another*
** *is-part-of-the-engineering-product-or-artifact*
** *is-based-upon*
<2> Relate-Different-or-Competing-Engineering-Artifacts
** *are-competing-products*
** *provides-a-comparison-of*
** *is-a-lower-standard-than*
<2> Relate-Engineering-Projects-to-One-Another
** *is-a-work-segment-of-the-engineering-project*

<1> Relate-the-Physical-Location-of-One-Object-or-Actor-to-Another
<2> Specify-Location-of-Residence

** *resides-in*
** *resides-next-to*
<2> Specify-Location-of-Work
** *practices-in*
<2> Specify-Physical-Proximity
** *is-located-next-to*

<1> Transfer-Possession-of-an-Object
** *gives-to*
** *disseminates-to*

<1> Own-or-Purchase-Something
<2> Own-Something
** *owns*
** *owns-facility*
<2> Purchase-or-Pay-for-Something
** *purchases-from*
** *pays-for*
** *offers-to-pay-for*

<1> Deal-with-Dangers-Hazards-or-Injuries
<2> Deal-with-Known-Dangers-or-Hazards
** *A-failure-of-would-be-hazardous-to-safety*
** *fails-standards-and-is-hazardous-to-safety*
** *is-a-safety-hazard* | *is-a-safety-hazard-due-to*
** *is-hazardous-material*
<2> Deal-with-Potential-Dangers-or-Hazards
** *A-failure-of-might-be-hazardous-to-safety*
** *fails-standards-and-may-be-hazardous-to-safety*
** *may-be-hazardous-to-safety* | *may-be-hazardous-to-safety-due-to*
** *may-be-hazardous-material*
<2> Inform-others-about-Dangers-or-Hazards-Real-or-Potential
** *should-be-informed-about-the-hazard-or-potential-hazard*
<2> Endure-an-Accident-or-Injury
** *sustains-an-injury*
** *is-involved-in-an-accident*

<1> Assess-the-Value-of-an-Object
** *is-worth* | *is-worth-to*

<1> Assess-the-Results-of-a-Test
** *fails*

<1> Know-Believe-or-Expect-Something
<2> Know-or-Believe-Something
** *knows*
** *discovers-that*
** *believes* | *believes-because*
<2> Expect-Something
** *expects-in-the-future*

A.1.2. The Actors and Objects Hierarchy

The Actors and objects Hierarchy for the engineering ethics domain is shown in its entirety below. The Actors branch of the hierarchy is shown on the left; the Objects branch of the hierarchy is shown on the right. The leaves of the hierarchy are the general types that can be used in case transcriptions.

- <1> Actor
  - <2> Individual-Engineering-Role
    - <3> Engineer
    - <3> Principal-Engineer
    - <3> Engineering-Manager
    - <3> Engineering-Technician
    - <3> Engineer-In-Training
    - <3> President-or-Director-of-Engineering-Firm
  - <2> Group-Engineering-Role
    - <3> Engineering-Firm
    - <3> Team-of-Engineers
    - <3> Engineering-Society
    - <3> Merger-of-Engineering-Organizations
    - <3> Engineering-Registration-Board
  - <2> Client-Role
    - <3> Client
    - <3> Client-Firm
  - <2> Commercial-Role
    - <3> Business-Leader
    - <3> Labor-Union
    - <3> Employment-Agency
    - <3> Engineering-Supplier
    - <3> Commercial-Organization
    - <3> Newspaper-Company
  - <2> Manufacturing-Role
    - <3> Manufacturing-Firm
    - <3> Workers-at-a-Manufacturing-Firm
  - <2> Construction-Role
    - <3> Development-or-Construction-Firm
    - <3> Construction-Contractor
  - <2> Academic-Role
    - <3> Group-Academic-Role
      - <4> University-or-College
      - <4> Unaccredited-College-or-University
      - <4> Technical-School
      - <4> University-or-College-Committee

- <1> Object
  - <2> Field-Within-Engineering
    - <3> Generic-Engineering
    - <3> Specialized-Type-of-Engineering
    - <3> Engineering-Management
  - <2> Engineering-Work-Object
    - <3> Engineering-Artifact
    - <3> Engineering-Project
    - <3> Design-Work
    - <3> Test-Samples
    - <3> Test
  - <2> Construction-Work-Object
    - <3> Construction-Work
    - <3> Construction-Supervision
    - <3> Construction-Permit
  - <2> Research-Object
    - <3> Engineering-Research
    - <3> Research-References
    - <3> Research-Data
    - <3> Research-Paper
  - <2> Paper-Letter-or-Document
    - <3> Technical-Report
    - <3> Evaluation-Report
    - <3> Published-Article
    - <3> An-Engineering-Magazine
    - <3> A-Newspaper
    - <3> Proposal
    - <3> Design-Document
    - <3> Legal-Employment-Agreement
    - <3> Letter
  - <2> Object-Belonging-to-an-Actor
    - <3> House
    - <3> Real-Estate
    - <3> Investments
    - <3> Facility-or-Site
  - <2> Educational-Object
    - <3> Technical-Curriculum
    - <3> Educational-Seminar-or-Course
  - <2> Names-of-Things
    - <3> An-Engineering-Firms-Name

<4> Academic-Accrediting-Team <3> Individual-Academic-Role <4> Graduate-Student <4> University-or-College-Professor <4> Authority-at-University-or-College <2> Government-or-Political-Role <3> Government-Role <4> Governmental-Body <4> Government-Authority <3> Political-Role <4> Political-Office-Holder <4> Political-Candidate <4> Political-Party <2> Legal-Role <3> Attorney <3> Law-Firm <3> Plaintiff <3> Defendant <3> Court-of-Law-or-Arbitration-Board <2> Public-Role <3> General-Public <3> Charitable-Organization <3> Public-Action-Committee <3> Community-Leader	<2> Design-Issue <3> Design-Errors <3> Design-Philosophy <2> Gift-or-Contribution <3> Tangible-Favor <2> Time <3> Period-of-Time <3> Absolute-Time <2> Money <3> Cost-of-Work <3> Amount-of-Money <3> Range-of-Money

A.1.3. Time Qualifiers

The Time Qualifiers and an explanation of how each is used are provided below. These explanations are slightly edited versions of those provided to case enterers on the case-acquisition web site. Note that whenever "Event" is mentioned in the table, this could refer to either an action or an event, i.e., a Fact in a Fact Chronology.

Time Qualifier	Explanation of Use
Pre-existing fact	**Intended Use**: This is a special-case qualifier intended primarily to handle general conditions that hold over the course of a scenario. That is, if there are certain events (i.e., Facts) that are true (or start) before the beginning of the facts of a story, this qualifier is used to denote such facts. Note that this qualifier is different from all of the others in that it relates to *all* other Facts of the scenario (except others that are also "Pre-existing fact"), and it typically relates to Facts that occur at a later time. **What is known**: ° Event A starts before the start of every other event in the scenario, except other events with the qualifier "Pre-existing fact" ° This is essentially the inverse of "After the start of …" except that it assumes a temporal relationship between Event A and *all* other events. **What is not necessarily known**: ° All that is known is that Event A starts before all of the other events. Event A could conceivably: conclude before the start of other events, subsume the duration of other events, or overlap with the other events.
After the start of ...	**Intended Use**: Use this qualifier when the only information known is the relative starting time of two events, i.e., that Event B starts after the start of Event A. **What is known**: ° Event B clearly starts after the start of Event A. **What is not necessarily known**: ° How long after the start of Event A Event B starts. *Subsumed Qualifier*: "[X time] after the start of…" ° Whether Event B starts after the conclusion of Event A. *Subsumed Qualifier*: "After the conclusion of…" ° Whether Event B starts immediately after the conclusion of Event A. *Subsumed Qualifier*: "Immediately after the conclusion of…" ° If Event B starts after the conclusion of Event A, how long after the conclusion of Event A. *Subsumed Qualifier*: "[X time] after the conclusion of…" ° Whether Event B overlaps with the ending of Event A. *Subsumed Qualifier*: "Ends…"

Time Qualifier	Explanation of Use
Starts at the same time as ...	**Intended Use**: Use this qualifier when the only information known is that Events A and B start at the same time. If it is also known that Events A and B occur over precisely the same time period, use the more specific qualifier "Occurs concurrently with…" If it is also known that Event A has a longer duration than Event B, i.e., that Event A subsumes Event B, use the qualifier "Occurs during…" / "Occurs as part of …" **What is known**: ° Event B clearly starts at the same time that Event A starts. **What is not necessarily known**: ° Whether Event B and Event A have precisely the same duration *Related Qualifier*: "Occurs concurrently with…" ° Whether Event B occurs during Event A, whether Event A occurs Event B. *Overlapping Qualifier*: "Occurs during…" / "Occurs as part of …"
<X time> after the start of ...	**Intended Use**: Use this qualifier when the following information is known: (1) Event B starts after the start of Event A and (2) the specific amount of time between the start of Event B and the start of Event A. If it is also known that Event A subsumes Event B, use the more specific qualifier "Occurs during…" / "Occurs as part of …" **What is known**: ° Event B starts a specific amount of time after Event A starts. ° There is some overlap in the durations of Events B and A. That is, it is known that Event B doesn't start after the conclusion of Event A. **What is not necessarily known**: ° Whether Event B occurs strictly during Event A or whether Event B concludes later than A concludes. *Overlapping Qualifier*: "Occurs during…" / "Occurs as part of …"
After the conclusion of ...	**Intended Use**: Use this qualifier when the only information known is that Event B starts after the conclusion of Event A. If it is also known that Event B starts immediately after the conclusion of Event A, use the more specific qualifier "Immediately after the conclusion of …" If it is also known that Event B starts a specified amount of time after the conclusion of Event A, use the more specific qualifier "[X time] after the conclusion of…" **What is known**: ° Event B clearly starts after the conclusion of Event A. **What is not necessarily known**: ° Whether Event B starts immediately after the conclusion of Event A. *Subsumed Qualifier*: "Immediately after the conclusion of…" ° Specifically how long after the conclusion of Event A Event B starts. *Subsumed Qualifier*: "[X time] after the conclusion of…"

Time Qualifier	Explanation of Use
Immediately after the conclusion of ...	**Intended Use**: Use this qualifier when it is known that Event B starts immediately after the conclusion of Event A. If the only information known is that Event B starts sometime after the conclusion of Event A (i.e., it is not known whether Event B starts immediately after Event A) use the more general qualifier "After the conclusion of …" **What is known**: ° Event B starts immediately after the conclusion of Event A. **What is not necessarily known**: ° Nothing. This qualifier fully specifies the temporal relationship between Events A and B.
<X time> after the conclusion of ...	**Intended Use**: Use this qualifier when it is known that Event B starts a specific amount of time after the conclusion of Event A. If the only information known is that Event B starts after the conclusion of Event A (i.e., the specific time lag between Event B and Event A is not known), use the more general qualifier "After the conclusion of …" **What is known**: ° Event B starts a specific amount of time after the conclusion of Event A. **What is not necessarily known**: ° Nothing. This qualifier fully specifies the temporal relationship between Events A and B.
Ends ...	**Intended Use**: Use this qualifier when Event B signifies, or triggers, the end of Event A. For instance, use it in the following situation: 1. Engineer A **<is employed by>** Company X. 2. Engineer A **<resigns from>** Company X. Ends 1 Typically, the type of "terminating event" represented by 2, above, cannot be specifically defined as occurring before, during, or after the "terminated event." But the duration of the "terminating event" clearly overlaps, in some way, with the end time of the "terminated event." **What is known**: ° Event B starts after the start of Event A. ° Event B either finishes, meets, or overlaps with the conclusion of Event A. **What is not necessarily known**: ° Whether Event B starts after the conclusion of Event A. *Overlapping Qualifier*: "After the conclusion of…" *Subsumed Qualifier*: "Immediately after the conclusion of…" ° Whether Event B occurs during Event A. *Overlapping Qualifier*: "Occurs during…" / "Occurs as part of…"

<table>
<tr><th>Time Qualifier</th><th>Explanation of Use</th></tr>
<tr><td>Occurs during ... / Occurs as part of ...</td><td>Intended Use: Use this qualifier when it is known that Event B occurs during, or as part of, Event A. Two possible phrasings are provided because there are some situations in which "during" is the more appropriate phrasing and other situations in which "part of" is the more appropriate phrasing. This qualifier should be favored over 3 of the 4 qualifiers that overlap with it, i.e., "Starts at the same time as …", "[X time] after the start of …", and "After the start of …" In particular, if the relationship between the start times of Events B and A is known, but it is also known that Event B occurs fully within the duration of Event A, the "Occurs during…" / "Occurs as part of…" qualifier should be used instead of the other possible qualifiers. The only exception is the "Ends" qualifier. The "Ends" qualifier should always be used in cases in which one event logically demarcates the termination of another.
What is known:
° The duration of Event B occurs within (or as part of) the duration of Event A.
° Event B does not occur concurrently with Event A.
What is not necessarily known:
° Whether Event B starts at the same time as Event A. Overlapping Qualifier: "Starts at the same time as …"
° Whether Event B concludes at the same time that Event A concludes. Overlapping Qualifiers: "Ends …", "[X time] after the start of …", "After the start of…"</td></tr>
<tr><td>Occurs concurrently with ...</td><td>Intended Use: Use this qualifier when it is known that Events B and A occur over precisely the same time period and are equal in duration.
What is known:
° The duration of Event B occurs at the same time as and is equivalent to Event A.
What is not necessarily known:
° Nothing. This qualifier fully and unambiguously specifies the temporal relationship between Events A and B.</td></tr>
</table>

A.2. The Extended Ethics Transcription Language (EETL)

The following two tables summarize the values that may be used in the code and case tables of the analysis representation.

Code Table Values	
Column Heading	**Possible Values**
Code	The code number cited by the review board in their analysis of the case.
Code Status	***Violated***: The code is violated in the current case. ***Not Violated***: The code is not violated in the current case. ***Not Applicable***: The code is not applicable in the current case. ***Changed***: The code was changed prior to the decision in the current case or will change as a result of the current case. ***Unknown***: The status of the code in the current case is unknown.
How Cited	***Explicitly Discussed***: The review board explicitly discusses the code in their analysis of the current case. ***Referenced Only***: The code is cited by the review board, but not discussed, in the analysis of the current case.
Grouped With	Other codes cited by the review board that are grouped with this code.
Overrides	Other codes cited by the review board that are overridden by this code.
Why Relevant?	A statement or statements expressing why this code is relevant to the current case. Each statement is linked to at least one of the following: ***A Fact number*** (from the ***Fact Chronology***): The Fact or Facts from the chronology that support the statement. ***Hypo***: A hypothetical was posed to support the statement. A quotation from the review board's analysis is also supplied. ***Unstated Assumption***: The statement appears to be supported by an unstated assumption on the part of the board. ***Inference Based on Facts***: The statement can be inferred from other facts of the case.
Why Violated, Not Violated, Changed, or Not Applicable?	A statement or statements expressing why this code is violated, not violated, or not applicable in the context of the current case. Each statement is linked to at least one of the following: ***A Fact number*** (from the ***Fact Chronology***): The Fact or Facts from the chronology that support the statement. ***Hypo***: A hypothetical was posed to support the statement. A quotation from the review board's analysis is also supplied. ***Unstated Assumption***: The statement appears to be supported by an unstated assumption on the part of the board. ***Inference Based on Facts***: The statement can be inferred from other facts of the case.

Case Table Values	
Column Heading	**Possible Values**
Case	The fact situation number cited by the review board in their analysis of the case.
Citation Type	***Analogous Precedent***: The cited case is analogous to the current case. ***Distinguishing Precedent***: The cited case is analogous to the current case, but has at least one critical characteristic distinguishing it from the current case. ***Relevant, But Not Controlling***: The cited case is relevant, perhaps to highlight a relevant issue or to provide background information, but it is not analogous. ***Unknown***: The cited case's relationship to the current case is unknown.
How Cited	***Explicitly Discussed***: The review board explicitly discusses the code in their analysis of the current case. ***Referenced Only***: The code is cited by the review board, but not discussed, in the analysis of the current case.
Grouped With	Other cases cited by the review board that are grouped with this case.
Q#	The question number of the cited fact situation. The value of the "Case" column concatenated with this column provides the full case name of the cited case.
Why Relevant?	A statement or statements expressing why this case is relevant to the current case. Each statement is linked to at least one of the following: ***A Fact number*** (from the ***Fact Chronology***): The Fact or Facts from the chronology that support the statement. ***Hypo***: A hypothetical was posed to support the statement. A quotation from the review board's analysis is also supplied. ***Unstated Assumption***: The statement appears to be supported by an unstated assumption on the part of the board. ***Inference Based on Facts***: The statement can be inferred from other facts of the case.
Why Distinguished or Analogous?	A statement or statements expressing why this case is distinguished from or analogous to the current case. Each statement is linked to at least one of the following: ***A Fact number*** (from the ***Fact Chronology***): The Fact or Facts from the chronology that support the statement. ***Hypo***: A hypothetical was posed to support the statement. A quotation from the review board's analysis is also supplied. ***Unstated Assumption***: The statement appears to be supported by an unstated assumption on the part of the board. ***Inference Based on Facts***: The statement can be inferred from other facts of the case.

A.3. The Code Hierarchy

The NSPE BER Code Hierarchy is shown in its entirety below. Code names are displayed in italics, followed by the text of the code. Code abstraction categories are shown in roman type, prefaced with the hierarchy level in angle brackets. A Code may reside in multiple code abstraction categories. A code that has been previously defined in another abstraction category is indicated by the text "(see …)" after the code name.

<1> Importance-of-Engineering-Code-of-Ethics

Code Preamble. Engineering is an important and learned profession. The members of the profession recognize that their work has a direct and vital impact on the quality of life for all people. Accordingly, the services provided by engineers require honesty, impartiality, fairness and equity, and must be dedicated to the protection of the public health, safety and welfare. In the practice of their profession, engineers must perform under a standard of professional behavior which requires adherence to the highest principles of ethical conduct on behalf of the public, clients, employers and the profession.

<1> Questionable-Associations-with-Others

Code II.1.d. Engineers shall not permit the use of their name or firm name nor associate in business ventures with any person or firm which they have reason to believe is engaging in fraudulent or dishonest business or professional practices.

Code III.9.b. Engineers shall not use association with a nonengineer, a corporation, or partnership as a "cloak" for unethical acts, but must accept personal responsibility for all professional acts.

<1> Competence

Code II.2.b. Engineers shall not affix their signatures to any plans or documents dealing with subject matter in which they lack competence, nor to any plan or document not prepared under their direction and control.

Code II.2.c. Engineers may accept assignments and assume responsibility for coordination of an entire project and sign and seal the engineering documents for the entire project, provided that each technical segment is signed and sealed only by the qualified engineers who prepared the segment.

Code II.3.b. Engineers may express publicly a professional opinion on technical subjects only when that opinion is founded upon adequate knowledge of the facts and competence in the subject matter.

<2> Qualifications-for-Work

Code I.2. Engineers, in the fulfillment of their professional duties, shall perform services only in areas of their competence.

Code II.2. Engineers shall perform services only in the areas of their competence.

Code II.2.a. Engineers shall undertake assignments only when qualified by education or experience in the specific technical fields involved.

<1> Duty-to-Public-Safety

Code I.1. Engineers, in the fulfillment of their professional duties, shall hold paramount the safety, health and welfare of the public in the performance of their professional duties.

Code II.1. Engineers shall hold paramount the safety, health and welfare of the public in the performance of their professional duties.

Code II.1.a. Engineers shall at all times recognize that their primary obligation is to protect the safety, health, property and welfare of the public. If their professional judgment is overruled under circumstances where the safety, health, property or welfare of the public are endangered, they shall notify their employer or client and such other authority as may be appropriate.

Code II.1.b. Engineers shall approve only those engineering documents which are safe for public health, property and welfare in conformity with accepted standards.

Code III.2. Engineers shall at all times strive to serve the public interest.

Code III.2.a. Engineers shall seek opportunities to be of constructive service in civic affairs and work for the advancement of the safety, health and well-being of their community.

Code III.2.b. Engineers shall not complete, sign or seal plans and/or specifications that are not of a design safe to the public health and welfare and in conformity with accepted engineering standards. If the client or employer insists on such unprofessional conduct, they shall notify the proper authorities and withdraw from further service on the project.

<1> Community-Service-and-Civic-Affairs

Code III.2. (see “Duty-to-Public-Safety”)

Code III.2.a. (see “Duty-to-Public-Safety”)

Code III.2.c. Engineers shall endeavor to extend public knowledge and appreciation of engineering and its achievements and to protect the engineering profession from misrepresentation and misunderstanding.

<1> Confidential-Information

Code II.1.c. Engineers shall not reveal facts, data or information obtained in a professional capacity without the prior consent of the client or employer except as authorized or required by law or this Code.

Code III.4. Engineers shall not disclose confidential information concerning the business affairs or technical processes of any present or former client or employer without his consent.

Code III.4.a. Engineers in the employ of others shall not without the consent of all interested parties enter promotional efforts or negotiations for work or make arrangements for other employment as a principal or to practice in connection with a specific project for which the Engineer has gained particular and specialized knowledge.

Code III.4.b. Engineers shall not, without the consent of all interested parties, participate in or represent an adversary interest in connection with a specific project or proceeding in which the Engineer has gained particular specialized knowledge on behalf of a former client or employer.

<1> Conflict-of-Interest

Code II.3.c. Engineers shall issue no statements, criticisms or arguments on technical matters which are inspired or paid for by interested parties, unless they have prefaced their comments by explicitly identifying the interested parties on whose behalf they are speaking, and by revealing the existence of any interest the engineers may have in the matters.

Code II.4.a. Engineers shall disclose all known or potential conflicts of interest to their employers or clients by promptly informing them of any business association, interest, or other circumstances which could influence or appear to influence their judgment or the quality of their services.

Code III.4.a. (see “Confidential-Information”)

Code III.4.b. (see “Confidential-Information”)

Code III.5. Engineers shall not be influenced in their professional duties by conflicting interests.

<2> Conflict-of-Interest-Related-to-Compensation

Code II.4.b. Engineers shall not accept compensation, financial or otherwise, from more than one party for services on the same project, or for services pertaining to the same project, unless the circumstances are fully disclosed to, and agreed to by, all interested parties.

Code II.4.c. Engineers shall not solicit or accept financial or other valuable consideration directly or indirectly, from contractors, their agents, or other parties in connection with work for employers or clients for which they are responsible.

Code III.5.a. Engineers shall not accept financial or other considerations, including free engineering designs, from material or equipment suppliers for specifying their product.

Code III.5.b. Engineers shall not accept commissions or allowances, directly or indirectly, from contractors or other parties dealing with clients or employers of the Engineer in connection with work for which the Engineer is responsible.

Code III.7.a. Engineers shall not request, propose, or accept a professional commission on a contingent basis under circumstances in which their professional judgment may be compromised.

<2> Conflict-of-Interest-Between-Public-and-Private-Roles

Code II.4.d. Engineers in public service as members, advisors or employees of a governmental or quasi-governmental body or department shall not participate in decisions with respect to professional services solicited or provided by them or their organizations in private or public engineering practice.

Code II.4.e. Engineers shall not solicit or accept a professional contract from a governmental body on which a principal or officer of their organization serves as a member.

<1> Credit-for-Engineering-Work

Code III.3.c. Consistent with the foregoing (i.e., III.3.a.), Engineers may prepare articles for the lay or technical press, but such articles shall not imply credit to the author for work performed by others.

Code III.10. Engineers shall give credit for engineering work to those to whom credit is due, and will recognize the proprietary interests of others.

Code III.10.a. Engineers shall, whenever possible, name the person or persons who may be individually responsible for designs, inventions, writings, or other accomplishments.

Code III.10.c. Engineers, before undertaking work for others in connection with which the Engineer may make improvements, plans, designs, inventions, or other records which may justify copyrights or patents, should enter into a positive agreement regarding ownership.

<1> Duty-to-Disclose

Code II.4.a. (see "Conflict-of-Interest")

Code III.1.a. Engineers shall admit and accept their own errors when proven wrong and refrain from distorting or altering the facts in an attempt to justify their decisions.

Code III.1.b. Engineers shall advise their clients or employers when they believe a project will not be successful.

<1> Duty-to-Employer

Code I.4. Engineers, in the fulfillment of their professional duties, shall act in professional matters for each employer or client as faithful agents or trustees.

Code II.4. Engineers shall act in professional matters for each employer or client as faithful agents or trustees.

Code III.1.e. Engineers shall not actively participate in strikes, picket lines, or other collective coercive action.

<2> Outside-Employment/Moonlighting

Code III.1.c. Engineers shall not accept outside employment to the detriment of their regular work or interest. Before accepting any outside employment they will notify their employers.

Code III.7.b. Engineers in salaried positions shall accept part-time engineering work only to the extent consistent with policies of the employer and in accordance with ethical considerations.

Code III.7.c. Engineers shall not use equipment, supplies, laboratory, or office facilities of an employer to carry on outside private practice without consent.

<1> Truthfulness-and-Honesty

Code II.3.a. Engineers shall be objective and truthful in professional reports, statements or testimony. They shall include all relevant and pertinent information in such reports, statements or testimony.

<2> Honesty-in-Advertising

*Code I.5.*_ Engineers, in the fulfillment of their professional duties, shall avoid deceptive acts in the solicitation of professional employment.

Code II.5. Engineers shall avoid deceptive acts in the solicitation of professional employment.

Code II.5.a. Engineers shall not falsify or permit misrepresentation of their, or their associates', academic or professional qualifications. They shall not misrepresent or exaggerate their degree of responsibility in or for the subject matter of prior assignments. Brochures or other presentations incident to the solicitation of employment shall not misrepresent pertinent facts concerning employers, employees, associates, joint venturers or past accomplishments with the intent and purpose of enhancing their qualifications and their work.

Code III.3.a. Engineers shall avoid the use of statements containing a material misrepresentation of fact or omitting a material fact necessary to keep statements from being misleading or intended or likely to create an unjustified expectation, or statements containing prediction of future success.

<2> Honesty-in-Public-Statements

Code I.3. Engineers, in the fulfillment of their professional duties, shall issue public statements only in an objective and truthful manner.

Code II.3. Engineers shall issue public statements only in an objective and truthful manner.

Code II.3.b. (see "Competence")

Code III.3. Engineers shall avoid all conduct or practice which is likely to discredit the profession or deceive the public.

<2> Honesty-in-Criticism-of-Other-Engineers

Code II.3.c. (see "Conflict-of-Interest")

Code III.7. Engineers shall not attempt to obtain employment or advancement or professional engagements by untruthfully criticizing other engineers, or by other improper or questionable methods.

Code III.8. Engineers shall not attempt to injure, maliciously or falsely, directly or indirectly, the professional reputation, prospects, practice or employment of other engineers, nor untruthfully criticize other engineers' work. Engineers who believe others are guilty of unethical or illegal practice shall present such information to the proper authority for action.

<2> Honesty-in-the-Hiring-of-Engineers

Code III.11.e. Engineers shall provide a prospective engineering employee with complete information on working conditions and proposed status of employment, and after employment will keep employees informed of any changes.

<3> Recruiting-Engineer-from-Another-Engineer

Code III.1.d. Engineers shall not attempt to attract an engineer from another employer by false or misleading pretenses.

Code III.3.b. Consistent with the foregoing (i.e. III.3.a.), Engineers may advertise for recruitment of personnel.

<1> General-Integrity

Code III.1. Engineers shall be guided in all their professional relations by the highest standards of integrity.

<1> Gifts-and-Political-Contributions

Code II.4.c. (see "Conflict-of-Interest")

Code II.5.b. Engineers shall not offer, give, solicit or receive, either directly or indirectly, any political contribution in an amount intended to influence the award of a contract by public authority, or which may be reasonably construed by the public of having the effect or intent to influence the award of a contract. They shall not offer any gift, or other valuable consideration in order to secure work. They shall not pay a commission, percentage or brokerage fee in order to secure work except to a bona fide employee or bona fide established commercial or marketing agencies retained by them.

<1> Encouraging-Professional-Development

Code III.11. Engineers shall cooperate in extending the effectiveness of the profession by interchanging information and experience with other engineers and students, and will endeavor to provide opportunity for the professional development and advancement of engineers under their supervision.

Code III.11.a. Engineers shall encourage engineering employees' efforts to improve their education.

Code III.11.b. Engineers shall encourage engineering employees to attend and present papers at professional and technical society meetings.

<1> Professional-Responsibility

Code III.9. Engineers shall accept personal responsibility for their professional activities; provided, however, that Engineers may seek indemnification for professional services arising out of their practice for other than gross negligence, where the Engineer's interests cannot otherwise be protected.

Code III.9.b. (see "Questionable-Associations-with-Others")

Code III.10. (see "Credit-for-Engineering-Work")

<1> Ownership-of-Engineering-Info

Code III.10.b. Engineers using designs supplied by a client recognize that the designs remain the property of the client and may not be duplicated by the Engineer for others without express permission.

Code III.10.c. (see "Credit-for-Engineering-Work")

Code III.10.d. Engineers' designs, data, records, and notes referring exclusively to an employer's work are the employer's property.

<1> Registration-Laws

Code III.9.a. Engineers shall conform with state registration laws in the practice of engineering.

Code III.11.c. Engineers shall urge engineering employees to become registered at the earliest possible date.

<1> Remuneration

Code III.6. Engineers shall uphold the principle of appropriate and adequate compensation for those engaged in engineering work.

Code III.6.a. Engineers shall not accept remuneration from either an employee or employment agency for giving employment.

Code III.6.b. Engineers, when employing other engineers, shall offer a salary according to professional qualifications.

Code III.7.b. (see "Outside-Employment/Moonlighting")

<1> Reviewing-the-Work-of-Other-Engineers

Code III.8. (see "Honesty-in-Criticism-of-Other-Engineers")

Code III.8.a. Engineers in private practice shall not review the work of another engineer for the same client, except with the knowledge of such engineer, or unless the connection of such engineer with the work has been terminated.

Code III.8.b. Engineers in governmental, industrial or educational employ are entitled to review and evaluate the work of other engineers when so required by their employment duties.

Code III.8.c. Engineers in sales or industrial employ are entitled to make engineering comparisons of represented products with products of other suppliers.

<1> Self-Promotion

Code III.1.f. Engineers shall avoid any act tending to promote their own interest at the expense of the dignity and integrity of the profession.

Code III.4.a. (see "Confidential-Information")

Code III.7. (see "Honesty-in-Criticism-of-Other-Engineers")

<1> Proper-Utilization-of-Engineering-Workers

Code III.11.d. Engineers shall assign a professional engineer duties of a nature to utilize full training and experience, insofar as possible, and delegate lesser functions to subprofessionals or to technicians.

<1> Unethical-Practice-by-Others

Code II.1.e. Engineers having knowledge of any alleged violation of this Code shall cooperate with the proper authorities in furnishing such information or assistance as may be required.

Code III.8. (see "Honesty-in-Criticism-of-Other-Engineers")

Appendix B: The SIROCCO Case-Acquisition Web Site

This appendix provides a brief overview of SIROCCO's case-acquisition web site, located at www.pitt.edu/~bmclaren/ethics, and reproduces the participant's guide portion of the web site. Figure B-6-1 shows the main page of the site. The primary sections of the site are accessed by a menu bar found at the top of the main page (and also at the top of the main page of every other section). An arrow points to the menu bar in Figure B-6-1.

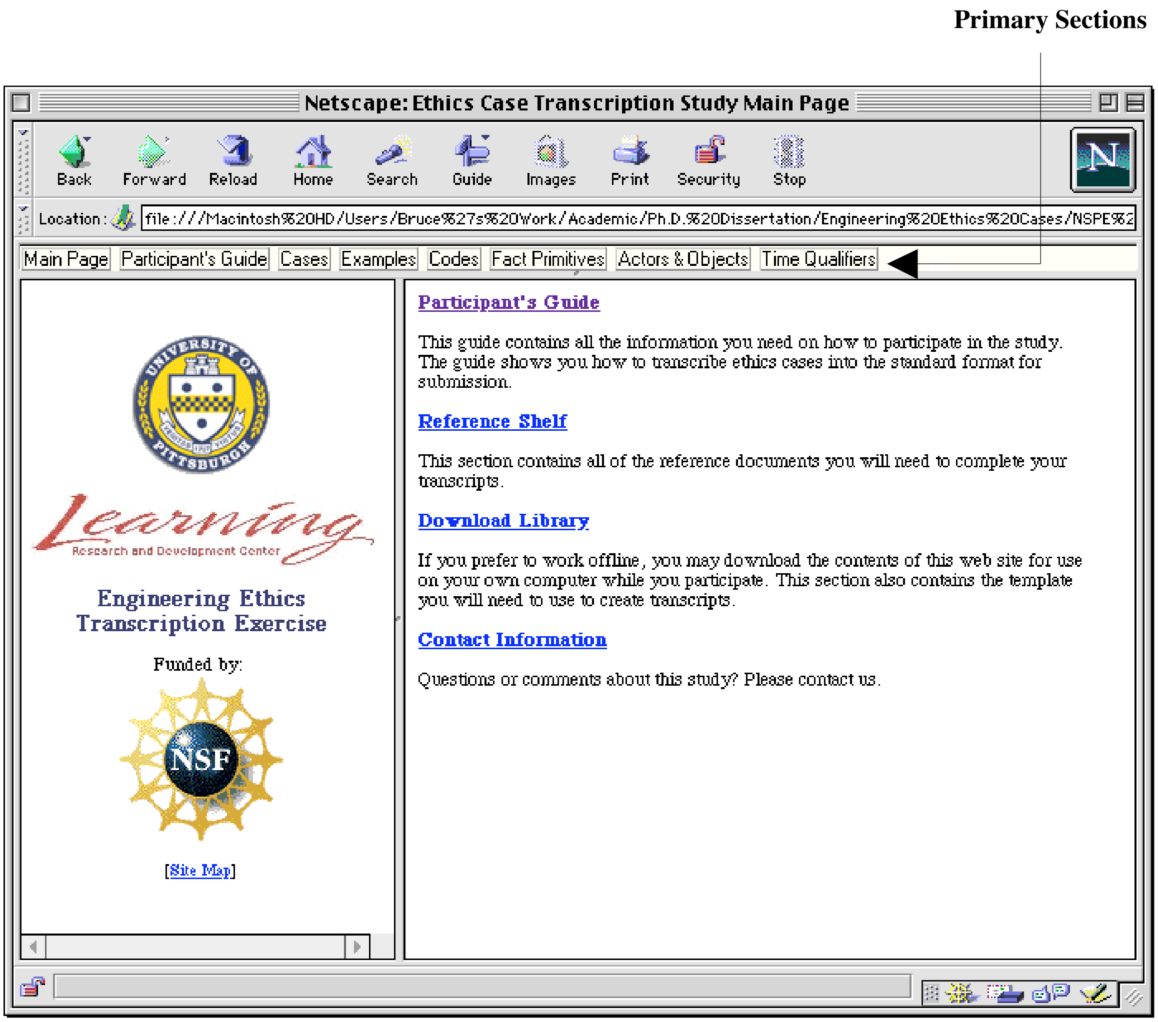


Figure B-6-1: A Screen Shot of the Case-Acquisition Web Site

The purpose of the case-acquisition web site is to support case enterers in transcribing cases into EETL. A "case representation template," used for filling in the representation of a single case, is available from the "Download Library," shown in the right frame of Figure B-6-1. The web site consists of the following primary sections:

- *The Participant's Guide*: The guide is intended to provide complete instructions on transcribing cases into EETL. It uses a tutorial approach. The guide traces the transcription of a single case into the language and explains each step of the process. The guide is reproduced below.
- *Cases*: The text of all of the NSPE BER cases, from 1958 through 1998, are available in this section.
- *Examples*: Examples of 47 transcribed fact situations are shown in this section.
- *Codes*: The full set of 75 NSPE BER ethics code provisions (revised in 1981) is shown in this section.
- *Fact Primitives*: The Action/Event Hierarchy, i.e., the complete set of Fact Primitives organized by category, is displayed in this section.
- *Actors and objects*: The complete set of Actors and Objects is provided in this section.
- *Time Qualifiers*: All of the Time Qualifiers, as well as explanations and examples of each, is shown in this section.

The case-acquisition web site is highly intralinked. For instance, in the participant's guide, there are numerous links from the textual explanations of various representational components to example uses of those components within the 47 transcribed fact situations. Also, each Fact Primitive, Actor, and Object is linked to every instance of its use in the 47 example transcriptions.

In the following sections, the participant's guide of the case-acquisition web site is presented in text format. Items that are hyperlinked to other locations on the web site are underlined and shown in a lighter font.

Note that the Extended Ethics Transcription Language (EETL) is not mentioned in the guide. For the case enterers, this is unnecessary information and, therefore, is not provided.

B.1. The Participant's Guide: An Introduction

Thank you for agreeing to participate in the engineering ethics transcription exercise. Participation entails reading engineering ethics dilemmas, and transcribing each dilemma using a restricted, predefined representation language. For each dilemma you will need to:

- Transcribe the text of the scenario into a set of chronological facts.
- Identify the actors (e.g. engineers, clients) and objects (e.g., technical reports,

products) involved in each scenario.

- Identify the questioned facts, the actor or actors whose ethical behavior is questioned, and the conclusion of the dilemma.
- Note the codes and cases cited by the board that led to their conclusion, and identify why and for what purpose each was cited.

It will take approximately two hours to transcribe each case. You are free to do the work in as many sessions as you need, and at any time. You are also free to consult with others (although not other participants in the exercise) or to refer to any written materials. Contact the exercise coordinator with any questions you may have.

The following documents are a tutorial that will cover all of the skills you will need to transcribe the cases. Please follow the tutorial all of the way through before attempting to transcribe any cases on your own. The exercise is complicated, but by understanding each of the tasks involved, and attempting each in turn, you will have no trouble.

In addition to completing this tutorial, you should study further examples of transcription work, and maybe try a few practice transcripts. A collection of example transcripts is available through the Reference Shelf area of this web site.

Before you can begin transcribing, you will need to download the MS Word 6.0 template that is already set up for transcription. This template facilitates much of the transcript formatting, and will save you the trouble of creating the transcript format from scratch. Your transcript ultimately must be submitted as an MS Word 6.0 file, but, if you wish, you may work with a newer version of Word while developing the transcript.

B.2. Helpful Hints

During this tutorial, and during transcription, some of the following practices may facilitate your work.

B.2.1. Be Careful with Transcript Structure and Syntax

It is important that you meticulously conform to the required structure, syntax, and special characters in your transcripts. The required format is discussed in this guide, and you can always study and refer to the example transcripts if you are unsure about any of the specifics.

A computer program will scan over all of your submitted transcripts, identifying any errors in structure and syntax. Transcripts with errors will be returned to you for revisions.

B.2.2. Contact the Exercise Coordinator if a Case Cannot be Transcribed

The reference materials that are provided to you on this web site should be sufficient to transcribe virtually every case assigned to you. However, you will occasionally encounter a case that simply cannot be transcribed with the provided materials, without grossly distorting the facts of the case. Don't attempt to transcribe such a case. In such a situation, you should promptly contact the exercise coordinator and alert him to this issue. He will likely assign a new case to you.

B.2.3. Opening Multiple Browser Windows

Especially during this tutorial, you will need to be able to refer to several documents at once. While you can achieve this by following links and using your browser's history commands, this method is somewhat ungainly, and often slow as it forces the browser to reload and redisplay pages.

You may find it easier to keep several browser windows open at once, each displaying a different document. Some web browsers have this facility built in. In some browsers instead of following a link by selecting it with the left mouse button you can open the link in a new window by selecting the link with the right mouse button (or by holding down the mouse button) and choosing "Open in New Window" from the menu that appears. You may also be able to run multiple browsers.

B.2.4. Dealing with Multiple Windows

If you are going to use multiple windows, you may also wish to familiarize yourself with an easy to way to switch between them. Under MS Windows, for instance, use the Alt and Tab keys simultaneously to switch between windows you have open.

The ability to switch easily between two or more windows will be very useful during transcription, as you will need to switch between the browser and MS Word often.

B.2.5. Using "Empty Template Rows" as an Aid in Doing Your Transcriptions

The last section of each transcript is comprised of several tables containing special characters. To avoid having to repeatedly retype these characters and also as a convenience for rapidly adding new rows, it is recommended that you always keep one "empty template row" (which is not *actually* empty, it will contain special characters) at the end of each table while you are transcribing a case. Whenever you need to add a new row to the table, simply execute a copy-and-paste of the empty template row. The transcript template that you will download for the transcription work contains the empty template rows. (Important Note: Be sure to delete the empty template rows when you are ready to submit the transcript.)

B.2.6. Printing

If your browser does not support multiple windows, if you cannot run multiple browsers, or if you simply work better with paper, you may wish to print out some of the files from this web site. In particular you may wish to print out the case used during this tutorial.

This web site does two things that make printing somewhat tricky, however. The first thing is that larger files (like this guide) have been broken into collections of smaller files to make browsing quicker. To keep you from having to view and print each section separately, some of these files have also been collected into single documents. You can find links to these documents (if they exist) at the bottom of the table of contents for each collection.

The other thing that makes printing tricky is the fact that this web site uses "frames." This site uses HTML frames to divide your browser window into several parts and display several documents simultaneously. In order to print one of the displayed documents you must first select (click in) the frame you wish to print before choosing your browser's "Print" command.

B.3. Reading the Case

The first step in transcribing a case is, of course, reading it. You'll save yourself a fair amount of time later if you read the case with the intent of transcribing it foremost in your mind, and keep your eye on the things you will be transcribing.

As you gain more experience transcribing, it may be faster to transcribe and read simultaneously. In the beginning, it will be more instructive to read the case first and begin transcribing once you have all of the information ready.

B.3.1. Finding Cases

All of the cases available for transcription are located in the Reference Shelf area of this web site. For this tutorial, you will be reading and transcribing Case 83-1.

During this tutorial, it will be helpful for you to have this case open in another browser window. If your browser supports multiple windows open the case in a new window and keep that window open for the duration of this tutorial. If your browser does not support multiple windows, or if you cannot run multiple browsers, you may wish to visit Case 83-1 now, print it, and return to this point.

When trying to find cases in the index to all cases, you can use your browser's search function to search for cases by number or name. Choosing a case from the list will display that case in your browser, with a reference frame beside the case.

B.3.2. Noting Facts

First, read the "Facts" section of Case 83-1. The facts section relates the scenario as it was brought to the attention of the board of ethical review (BER). As you read, identify the events, actors, and objects relevant to the scenario. Determine in what order the events happened. Determine what people were involved. Determine what objects were involved. Pay attention to the relationships between all these things, e.g. who did what to whom.

Some of these things, will be obvious. For instance, you can see right away that two people (Engineers A and B) are involved.

Some facts are not explicitly stated in the text, but will need to be inferred. For instance, in Case 83-1, it is explicitly stated that "Engineer A thereupon notified clients of Engineer B," but it may not be obvious that this fact also implies that Engineer B had clients. Look for these implied and important facts.

You should also notice that the facts of the case are laid out logically, and fairly sequentially. In most cases, the sentences are short and contain one event or fact per sentence. Also in most cases, events are presented in the order they occurred.

Lastly, try to identify if any of the facts are not really relevant to the case. The board may record some facts that, although perhaps interesting, have no real bearing on the case. You do not want to transcribe these extraneous facts.

B.3.3. Identifying the Question

Following the facts of the case is a list of the actual questions that were brought before the board. Make sure that you can identify each questioned action as a fact in the chronology, and which actor's actions are being questioned.

B.3.4. Studying the References

In this section, the board lists specific sections of the NSPE code of ethics that they believe have bearing on this case. The text of each code is included, but if you need a more detailed breakdown of how the code works, you can select the code number in the transcript to display a full representation of the code in the "reference frame", the upper right pane of the window.

The board will discuss how these codes relate to the matter at hand, but as you gain more experience transcribing cases, you may begin to anticipate their arguments.

B.3.5. Analyzing the Discussion

In the "discussion" section of the case, the board members lay out the arguments that lead to a decision regarding the questioned facts. Make note of where the board is absolutely sure of something, and where they are making assumptions and judgment calls. Pay attention to which codes and cases are cited, how they relate to each other, and how they are being used.

It may be useful to refer back to the "Questions" section, and also make note of which parts of the discussion relate to which questioned facts.

There may also be additional discussion following the conclusion of the board. You are not responsible for transcribing this additional material. Sometimes this material clarifies the board's decision, so you may wish to read it anyway. You may also find it interesting.

B.3.6. Noting the Conclusion

There will be one conclusion for each questioned fact. In your transcript, you will need to note these decisions, and who has been judged to be at fault or not at fault. At this point it may be useful to refer back to the discussion and determine which arguments were used to *support* specific conclusions, which arguments *conflict with* specific conclusions, and which were used informationally and *neither support nor conflict* with specific conclusions.

B.4. Creating the Transcript

Once you have read the text of the case, you are ready to begin creating your transcript of the case. There are five main steps to creating your transcript.

1. Breaking down the facts into actions and events, and identifying the time relationships between them.
2. Noting the actors and objects involved in the case, and assigning each a descriptive type.
3. Summarizing the board's decision of each questioned fact.
4. Listing the codes referred to by the board to reach their decision, and analyzing how each was used.

5. Listing the cases referred to by the board during discussion of the case, and analyzing how each was used.

This guide will continue to use Case 83-1, as an example, and refer to it often. If you have not already done so, you may wish to open this case in another browser window, or print it out so that you can refer to it easily.

B.5. Creating the Fact Chronology

The fact chronology section of the transcript is represented as a table of ordered facts and time qualifiers. The facts are presented in approximate time order, and the time qualifiers clarify how the facts relate to each other chronologically. The questioned facts are identified as well.

Each fact comprises one "fact primitive". These fact primitives indicate actions and events. Sometimes, the fact is simple, and just indicates an actor and an action. Often, the fact indicates a "subject-verb-object" relationship, describing who did what to whom, or who did what to which object. Occasionally, the fact relates an actor to another fact.

B.5.1. Choosing Fact Primitives

The next step is to separate the scenario of the case into individual facts. For this task, you'll need to become familiar with the "fact primitives". The fact primitives are essentially verb phrases, that indicate the specific actions and events in the fact chronology.

You can review the full list of fact primitives directly, or access it through this site's Reference Shelf. You may wish to open up the list in a separate browser window while you create the fact chronology. To make it easier to search the list, the list of basic facts is categorized by type.

The ellipses (i.e., "**...**") found in the fact primitives indicate that an actor, object, or possibly another fact must "fill in the blank." One to three actors, objects, or facts are required by each fact primitive. For instance, "**... <employs> ...**" requires two actors, one before "**<employs>**" to indicate the employer and one after "**<employs>**" to indicate the employee. An example of this fact primitive is: "Firm X **<employs>** Engineer A."

Selecting a fact in the list will cause a detailed description of that fact to display in the reference frame. The detailed description has several parts.

Event/State Type

The facts are categorized into three types to ensure that time qualifiers are used consistently. People might interpret the meaning and general duration of the same fact in different ways. For instance, the fact "Y **<employs>** X" could be viewed as a short duration event in which X *begins* employment with Y. It could also be viewed as X's "state" of employment, something that would probably persist over a long period of time.

Therefore, the facts have an explicit categorization that enables you to be more precise when defining the time relations between different facts. For example, if you think of "Y **<employs>** X" as being a long-term state, then subsequent facts such as "X **<designs>** Building" can be viewed as occurring *during* "Y **<employs>** X". If you were to think of "Y **<employs>** X" as being a short-term "starting" event, then you might view "X **<designs>** Building" as occurring after the conclusion of "Y **<employs>** X". This is an important difference in time sequence.

Event

An "event" fact primitive has a **relatively short duration**. Other events can occur within or overlap with "event" primitives, but typically only a small number of other primitives will overlap with an event primitive. For instance, the primitive "**… <accepts an offer of employment from> …**" is an "event" primitive; it occurs over a relatively short duration of time (perhaps seconds or minutes) and is not likely to overlap with many, if any, other primitives.

State

A "state" fact primitive has a **relatively long duration**. In effect, a "state" primitive covers *both* an event and a state. For instance, the primitive "**… <employs> …**" entails *both* the "event" when employment starts, as well as the "state" of being employed. The "**… <employs> …**" primitive typically occurs over a relatively long period of time (probably years, maybe decades) and it is likely to overlap with, or totally encompass, other events or states.

Event, Typically Terminates

An "event, typically terminates" fact primitive is a special type of event. This primitive typically (but not necessarily) terminates a state primitive. For instance, the primitive "**… <resigns employment with> …**" can be used to terminate the "**… <employs> …**" state primitive. These primitives thus typically use an "Ends..." time qualifier.

It is important that you check to make sure your interpretation of the duration of each fact you use matches that in the code representation.

Description

The description provides instruction on specific situations that should be described by this fact primitive. If the situation you are attempting to describe does not match the description of the primitive, search for a more appropriate one. If a primitive has optional clauses, their usage is also described.

Hint

Some fact primitives have hints that further delineate their usage. If another, more specific or more general fact would be more appropriate, the hint will indicate this.

Inverse Form

This indicates another form of the primitive that you can use if you wish to transpose the order of the actors, objects, or facts that surround the fact primitive. For instance, "**… <employs> …**" can also be phrased as "(**… <is employed by> …**)." The inverse form may provide natural phrasing in your fact chronology.

Plural Form

Since the fact primitives are essentially verb phrases, the conjugation of the verb may be different if any of the actors/objects/facts are plural. This form shows the primitive is different for that case.

Negative Form

This form is simply the logical negation of the primitive, for when you want to indicate the exact opposite situation to that represented by the primitive.

Inv.-Neg. Form

The Inverse Negative form is the logical negation of the inverse form.

Neg.-Plural Form

The Negative Plural form is the logical negation of the plural form.

References

Most (but not all) of the fact primitives are followed by numbers in parentheses (e.g. "58-1 [1]," "92-6 [1]"). These numbers are cross-references to specific transcriptions in the Examples that use the primitive. For instance, the number "58-1 [1]" following "… employs …" means that line number 1 of the fact sequence of Case 58-1 contains the "employs" fact primitive. Use the cross-references as a way to examine and verify specific usage of facts.

To choose the fact primitives for the chronology, go through the facts of the case, pick out actions in the scenario, and match fact primitives to them. The goal is to represent the important events in the case as accurately as possible, given the limited set of event primitives, actors, and objects. Sometimes, you may not be able to be completely accurate, but do the best you can.

This part of the tutorial also begins the actual creation of the transcript. Copy the transcript template to a new document, and name that document "83-1.doc" or something similar. Open the file and enter "83-1" at the top next to the word "Case". Place the cursor in the first cell of the Fact Chronology table, and begin.

Once again, the facts of Case 83-1 are as follows. You may also wish to refer to the full list of fact primitives during this exercise.

Fact Situation 83-1:

Engineer A worked for Engineer B. On November 15, 1982 Engineer B notified Engineer A that Engineer B was going to terminate Engineer A because of lack of work. Engineer A thereupon notified clients of Engineer B that Engineer A was planning to start another engineering firm and would appreciate being considered for future work. Meanwhile, Engineer A continued to work for Engineer B for several additional months after the November termination notice. During that period, Engineer B distributed a previously printed brochure listing Engineer A as one of Engineer B's key employees, and continued to use the previously printed brochure with Engineer A's name in it well after Engineer B did in fact terminate Engineer A.

Question 1: (Case 83-1-1)

Was it ethical for Engineer A to notify clients of Engineer B that Engineer A was planning to start a firm and would appreciate being considered for future work while still in the employ of Engineer B?

Question 2: (Case 83-1-2)

Was it ethical for Engineer B to distribute a brochure listing Engineer A as a key employee in view of the fact that Engineer B had given Engineer A a notice of termination?

Question 3: (Case 83-1-3)

Was it ethical for Engineer B to distribute a brochure listing Engineer A as a key employee after Engineer A's actual termination?

The word "worked" is the key verb in the first sentence. Fact primitives relating to employment are at the top of the list. Primitives relating to starting employment are in the first sub-section. The primitive **… <employs> … [<as> …]** seems to be fairly close to the situation of this fact. It is a "state" primitive, and will properly represent a period of time during which some of the other events of this case occur. Select it in the list, and view the detailed information. Since the description matches this situation, search the forms of the primitive to see if one is close to the structure of this sentence. The inverse form, **… <is employed by> …**, matches the structure of the sentence.

Copy the fact primitive from the browser window (if you decide to type by hand, be sure to include the "angle braces", i.e., "<" and ""), and paste it into the table as fact number 1. For readability, you may want to **boldface** the fact primitive text, as shown below. However, the boldface formatting is optional. Replace the ellipses in the primitive with the actors you identified, to create the fact,

1. Engineer A **<is employed>** by Engineer B.

End your fact with a period.

Reading along, you can see that in most cases, the events are presented roughly in chronological order. The second sentence relates a fact that quite obviously occurs after fact one, Engineer B informing Engineer A that his employment will be terminated.

In the list of fact primitives, scroll down to the section that contains fact primitives dealing with "Informing of Termination of Employment". The second primitive in this section, "**… <is informed of termination by> …**" correctly relates the employer and employee. Note that this primitive is an "event". Again, copy the primitive and paste it into the table, then replace the ellipses with the actors Engineer A and Engineer B to make the second fact in the chronology.

2. Engineer A **<is informed of termination by>** Engineer B.

The next sentence illustrates the situation of a single sentence containing two facts, one of which may not be immediately obvious. The most obvious fact related by this sentence is that Engineer A offers services to the clients of Engineer B. Since Engineer B has clients, and Engineer B's clients are central to the case, this is also fact worth noting, although it is not explicitly stated in its own sentence.

This also brings up another interesting issue. The fact that Engineer has clients obviously predates the termination notice, so when you enter this fact into the chronology, you will have to enter it before what is currently fact number two. This is not a problem. Place the cursor in fact number two, and select the menu command "Table Insert Rows". The rows should automatically renumber themselves.

If we assume that Engineer B was hired at some point to provide engineering services for those clients, the most appropriate fact primitive is found in the section, "Starting Work with a Client," and since "… hires the services of … [as … | for …]" is the only primitive in this section, the primitive you want must be either that one, or an alternate form of it. You could say that

2. Clients **<hire the services of>** Engineer B.

but since Engineers A and B are the primary actors in this case, it would be clearer to use the inverse form of the primitive, yielding

2. Engineer B **<is hired to provide services for>** Clients.

followed by

3. Engineer A **<is informed of termination by>** Engineer B.

which is now fact number three. Don't forget to add the obvious fact,

4. Engineer A **<offers services to>** Clients.

It is not always necessary, or even possible, to go through the facts of the case sentence by sentence, assigning fact primitives to every piece of information. First of all, some of the facts in the text of the case may not be fully relevant to the ethical dilemma. The next sentence of the case does not add any relevant factual information and does not need to be transcribed. It does add some information about the time relationships between some of the facts in the chronology. This information will come in handy soon.

The last and final sentence is so complex that it actually encompasses three major facts, and five facts total. We learn that Engineer B distributed a brochure, that Engineer A was eventually terminated, and Engineer B continued to distribute that brochure. Address the first fact first.

Sometimes, an ellipsis in a fact primitive is not filled by an actor or object, but by another fact. The next sentence of the case could possibly be represented as a complex relationship between Engineer B, a brochure, and the information contained within it. This complexity is not necessary, however, if you transcribe the information as a simple relationship of facts.

What did Engineer B do?

Engineer B distributed a brochure.

Why would Engineer B distribute a brochure?

To attract new business.

What's more important, the brochure itself, or the information contained within it?

The brochure is not described in any great detail, and is not as relevant to the case as idea that it lists Engineer A as a key employee.

When you replace an ellipsis with another fact, place the secondary fact in parentheses.

5. Engineer B **<advertises or solicits engineering business using>** (Engineer A **<is a key engineering employee of>** Engineer B).

The next thing we know for sure is that Engineer A was eventually terminated.

6. Engineer A **<is terminated by>** Engineer B.

The last piece of information in the case is that Engineer B continued to solicit new business using the same outdated brochure. Since the action is essentially the same as it was before Engineer A's termination, you can copy the earlier structure to create this new fact.

7. Engineer B **<advertises or solicits engineering business using>** (Engineer A **<is a key engineering employee of>** Engineer B).

Multiple Objects, Actors, and Internal Facts

Sometimes it will be necessary to fill in the ellipses (i.e., "...") of a fact primitive with a **conjunction** of objects, actors, or other facts. If more than one object, actor, or other fact is required, connect the elements with a series of commas and the conjunction (i.e., "&") character. (Examples: 64-10 [4, 5, 6, 8], 58-1 [4, 5, 7], 72-4 [4], 77-11 [15, 16, 17])

Using Fact Modifiers

Some facts will need to have "modifiers" attached to them. These modifiers are optional but sometimes necessary to appropriately express a fact. For instance, if you needed to relate that Engineer A developed most of a proposal before being terminated, there is a fact primitive that expresses the development of a proposal, but a modifier is needed to express the incomplete nature of that proposal. The valid modifiers are:

partially, substantially

> Use one of these two modifiers to express that the actor was responsible for a quantifiable portion of the fact that is either less than half or more than half, respectively. For instance, in Case 58-1 there is a fact "Engineer X **<designs>** ***{partially}*** Hydroelectric Plant", indicating that Engineer X did some portion, but not all and probably less than half, of the design.

limited, extensive

> Use one of these two modifiers to express that the actor was responsible for a portion of the fact that cannot be expressed in a strictly quantifiable way. For instance, in Case 72-11 there is a fact "Engineer Doe **<has experience in>** ***{limited}*** Engineering Management," indicating that Doe has experience, but the experience is limited in nature.

Include the modifiers in "curly braces." For readability, you may want to use ***bold-italics*** text for the modifier, as shown above. However, the bold-italics formatting is optional.

Indicating Questioned Facts

After identifying the facts and listing them chronologically, you need to indicate which actions in the fact chronology are being questioned by the board. These facts will need to be marked in the fact chronology. (Examples: 75-3 [6], 77-11 [13, 14, 15, 17])

If you refer to the "Questions" section of Case 83-1, you can see that there are three questioned facts.

1. Was it ethical for Engineer A to notify clients of Engineer B that Engineer A was planning to start a firm and would appreciate being considered for future work while still in the employ of Engineer B?

This question refers fact number four, where Engineer A offers services to the clients. Add the text "*[Questioned Fact 1]*" to the table cell that holds fact #4. Surround the text in brackets. For readability, you may want to use *italics* for the questioned fact text, as shown above. However, the italics formatting is optional.

2. Was it ethical for Engineer B to distribute a brochure listing Engineer A as a key employee in view of the fact that Engineer B had given Engineer A a notice of termination?

This question refers fact number five, where Engineer B solicits business. Add the text "*[Questioned Fact 2]*" to the table cell that holds fact #5.

3. Was it ethical for Engineer B to distribute a brochure listing Engineer A as a key employee after Engineer A's actual termination?

This question refers fact number seven, where Engineer B continues to use the misleading brochure. Add the text "*[Questioned Fact 3]*" to the table cell that holds fact #7.

B.5.2. Choosing Time Qualifiers

Once you have identified the facts, you need to assign qualifiers to those facts to clarify the time relationships between them. The Time Qualifiers are entered into the second column of the Fact Chronology table. Basically, each qualifier clarifies the time relationship between the current fact, and the facts before it.

At this point, you should load the Time Qualifiers list into a new browser window for reference. Selecting a qualifier from the contents list will move the display to the section of the document that describes the usage of that qualifier.

The description of each qualifier has four parts that describe its use.

Intended Use

The first section describes in a general way the relationship that this qualifier is meant to express.

What We Know

The next section describes specifically what information we need to know about the time relationship between two facts in order to choose this qualifier. If you are not absolutely sure about the information in this section, another qualifier may be more appropriate.

What We Don't Necessarily Know

This section lists information that you may or may not have about the time relationship between two facts. If you do have one of these pieces of information, the qualifier shown after it may be more appropriate.

References

The last section contains links to examples that use this qualifier in the fact chronology. The number or numbers in brackets beside each case number show which facts in that case use this qualifier.

By double checking the information you have against all the parts of the description of a qualifier, you can make sure you are using the qualifier that most accurately represents the relationship.

Each fact in the chronology must have at least one time qualifier. Some facts may have more than one qualifier. Go through the facts of Case 83-1, and assign qualifiers to each fact.

The ellipses (i.e., "**...**") found in the time qualifiers indicate that a fact number or numbers must "fill in the blank." For example, if fact number five ends the state represented by fact number three, include the qualifier "Ends 3" next to line 5 of the fact chronology.

1. *Engineer A* ***<is employed by>*** *Engineer B.*

   This state begins before any of the actions relevant to the case, and so earns the qualifier, "Pre-existing fact."

2. *Engineer B* ***<is hired to provide services for>*** *Clients.*

   This state also begins before any of the relevant actions. Since it is immaterial whether the relationship with the clients begins before or after Engineer A's employment, simply use "Pre-existing fact" here as well.

3. *Engineer A* ***<is informed of termination by>*** *Engineer B.*

   This event obviously occurs after the start of A's employment, but since we don't know exactly how long, use the qualifier, "After the start of 1."

4. *Engineer A* ***<offers services to>*** *Clients.*

   The case informs us that it is materially important that this event occurs during A's employment, and while these clients were still clients of B. Therefore, you must use the qualifier, "Occurs during 1, 2."

   Since you also know that this did not happen until after A was informed of his pending termination, you also need to include, "After the conclusion of 3."

5. *Engineer B* ***<advertises or solicits engineering business using>*** *(Engineer A* ***<is a key engineering employee of>*** *Engineer B).*

   It is not only relevant that this event occurs during facts one and two, necessitating the qualifier "Occurs during 1, 2", but that the solicitation occurs after B has informed A of A's impending termination, so you must include the qualifier "After the conclusion of 3."

6. *Engineer A* ***<is terminated by>*** *Engineer B.*

   It is explicitly stated that this event occurs "for several additional months after the November termination notice," and so rather than use the generic "After the conclusion of 3," qualifier, use the more specific "Several months after the conclusion of 3."

   Also, if you check the representation for this fact primitive, you will see that it is an event that typically terminates. Even if you don't check, it should be obvious that this event ends fact number one, so include the qualifier "Ends 1."

7. *Engineer B* ***<advertises or solicits engineering business using>*** *(Engineer A* ***<is a key engineering employee of>*** *Engineer B).*

   Since it is most relevant that this even occurs after A's actual termination, include the qualifier, "After the conclusion of 6."

You have now completed the fact chronology section of your transcript. Before you continue, you should save your transcript file (the Microsoft Word document).

B.6. Listing the Actors and objects

After creating the fact chronology, you must assign actor and object "types" to the proper names you gave in the chronology. If you identified them before beginning the transcript, you already have half of this part done.

Start by listing the actors and objects from the fact chronology in the transcript file as a numbered list under the heading, "Actor and Object Types." As you identified before, there are three actors in this case, and no objects.

Engineer A

Engineer B

Clients

Then add an "arrow" next to each, made of two dashes and a greater-than symbol.

Engineer A –>

Engineer B –>

Clients –>

You will add the type for each actor next to this arrow.

You can review the full list of actor and object types directly, or access it through this site's Reference Shelf. You may wish to open up the list in a separate browser window while you complete this section. The contents list in the left half of the window shows the categories of actor and object types. Select a category in the list to display the actor and object types of that category in the right half of the window.

B.6.1. Choosing Actor Types

For each actor in the case, find the actor type that most closely describes the actor, and add the type to the list.

Engineer A

This actor is an individual, so select "Individual Engineering Roles" in the contents list. Of the six roles, you only have enough information to assign the most general, "Engineer" role to Engineer A. Copy or enter **Engineer** next to "Engineer A" in the list. (Make sure it is displayed in bold type).

Engineer B

While you do know a bit more about Engineer B, (that he hired Engineer A) you do not know enough to assign a more specific type like "Principal Engineer" or "Engineering Manager".
As a result, you must again use the actor type, **Engineer**.

Clients

Select "Client Roles" in the contents list.
Since there is only one client role, "**Client Firm**" enter that in the list next to "Clients."

The list in your transcript should now look like this:

Engineer A –> **Engineer**.

Engineer B –> **Engineer**.

Clients –> **Client Firm**.

B.6.2. Choosing Object Types

Were there objects in this case, you would need to apply object types to them as well. List the objects in the same way, with "arrows" after them. Find object types in the list by looking under a general category, and add the specific object type to the list, after the object from the chronology.

Before you continue, you should save your transcript file.

B.7. Summarizing the Board's Decision

Once you have completed the fact chronology and list of actor and object types, you are ready to proceed to transcribing the board's decision. Even though the cases typically list "Discussion" before "Conclusion", you will first transcribe the decision, and then the analysis used to support that decision.

Scroll down through the text of the case to the "Conclusion" section. This section presents direct judgments on the questioned facts from the chronology. For each questioned fact, you will need to create a three-line summary of the board's decision of that fact. These summaries are placed in the transcript under the heading, "The Board's Analysis."

Questioned Fact(s) [number]:

If there is more than one questioned fact, replace the word *number* with the number of the question. That is, "Question 1" or "Q1" in the case becomes "Questioned Fact(s) 1" in the transcript.

To the right of this label, add the number of the questioned fact from the fact chronology.

Questioned Actor or Actors*:*

To the right of this label, list the actor or actors whose actions are being questioned in this fact.

The Board's Conclusion*:*

To the right of this label, add the decision of the board. Only one of two words is valid; either the decision is "Ethical" or it is "Unethical."

In Case 83-1, there are three questioned facts, so you need to create three summaries.

Q1. It was unethical for Engineer A to notify clients of Engineer B that Engineer A was planning to start a firm and would appreciate being considered for work while still in the employ of Engineer B.

This is the first questioned fact, so change the label to say, "Questioned Fact(s) 1". It corresponds to fact number four in the chronology, so enter "Fact 4" next to that label. Engineer A's action was questioned, so add "Engineer A" next to the second label. The board's decision was "Unethical", so add this next to the final label.

Q2. It was not unethical for Engineer B to distribute a previously printed brochure listing Engineer A as a key employee provided Engineer B apprised the prospective client during the negotiation of-Engineer A's pending termination.

This is the second questioned fact, so change the label to say, "Questioned Fact(s) 2". It corresponds to fact number five in the chronology, so enter "Fact 5" next to that label. Engineer B's action was questioned, so add "Engineer B" next to the second label.

The board's decision was "not unethical", so add the word "Ethical" next to the final label.

Q3. It was unethical for Engineer B to distribute a brochure listing Engineer A as a key employee after Engineer A's actual termination.

This is the third questioned fact, so change the label to say, "Questioned Fact(s) 3". It corresponds to fact number seven in the chronology, so enter "Fact 7" next to that label. Engineer B's action was questioned, so add "Engineer B" next to the second label. The board's decision was "unethical", so add that next to the final label.

Before you continue, you should save your transcript file.

B.8. Listing Cited Codes

Once you have summarized the board's decisions of the questioned facts in the case, you must transcribe how the board argued each of those decisions in the discussion. This is presented in the transcript as a set of tables, with between one and three tables following each decision. The first part of each table shows the codes referenced by the board in discussion.

The first table contains the evidence used to support the conclusion. The second table contains the evidence used that conflicts with the conclusion. Use the third table to record the evidence in the board's discussion that neither directly supports, nor directly conflicts with the board's conclusion.

The table has seven columns that detail how each code was used in the discussion.

B.8.1. Code

This column simply lists each code cited by the board. These code numbers are listed at the top of the case, so you can just copy them and paste them into the table, each starting its own row.

B.8.2. Code Status

Basically, use this column to record whether or not the code was violated. Here are the possible values that may be entered under this column:

Violated

Enter this value if one of the following is true: (1) the board *explicitly* indicates that the code was violated (2) the board does not explicitly indicate that the code was violated, but it *implies,* in general discussion that it was violated. (Examples: 84-6 [cited codes II.3., III.1.f.], 89-2 [cited codes II.4., II.5., III.7.])

Not Violated

Enter this value if one of the following is true: (1) the board *explicitly* indicates that the code was not violated (2) the board does not explicitly indicate that the code was not violated, but it *implies,* in general discussion that it was not violated. (Examples: 69-10 [cited codes 5, 12, 12(b)], 89-4 [cited code II.3.a.])

Not Applicable

Sometimes a code is referenced, but the board argues and ultimately decides that it does not apply to a given case. Enter this value if the board notes that this code does not apply to this case. (Examples: 84-6 [cited code III.1.e.], 89-4 [cited code III.4.b.])

Changed

Over time, the specific language of some codes might change, altering the detailed meaning of the particular codes in question. If the board notes that a code has changed, enter this value. Sometimes, it is the cited code that has changed, but other times the changed code is a previous version of the code from an earlier code set. If the code that was changed is a different code, you will need to list that code in the "Grouped With" column, described below. (Examples: 58-1 [cited code C27], 79-5 [cited code 3(f)], 85-6 [cited code II.4.a.])

Unknown

If you cannot tell from the discussion that one of the above terms is more appropriate you can enter this value. (Example: 92-1 [cited code III.5.a.])

You may also need to include special status modifiers to indicate the relative importance of cited codes. If the board's discussion cites several codes, but one of them seems to have more bearing than some of the others, include the "More Importance" modifier in the "Code Status" column. If one or more of the cited codes seem to have less bearing, include the "Less Importance" modifier. These importance values are optional. They should only be used when the relative importance of cases is made obvious in the board's discussion. (Examples: 60-9-A [cited code R1:4], 91-5 [cited code II.2.a.])

B.8.3. How Cited

There are two possible values for this column.

Explicitly Discussed

If the board cites the code in the "References" section, and also discusses its relevance to the case, enter this value in the "How Cited" column. (*Important note:* Sometimes the board will review a code in the discussion without explicitly stating its **code number**. In a situation such as this, the code should still be considered "explicitly discussed.")

(Examples: 71-4 [cited code 12], 77-11 [cited code 11(a)])

Referenced Only

If the board cites the code in the "References" section, but does not discuss its relevance to the case, enter this value in the "How Cited" column. (*Important note:* Just because the board doesn't explicitly mention the **code number** of a code in the discussion does not necessarily mean that the code should be considered "referenced only." If the board discusses a code without mentioning its number, it should be considered "explicitly discussed" instead of "referenced only.") (Examples: 71-4 [cited code 5], 77-11 [cited code 11])

B.8.4. Grouped With

If the code is mentioned along with other codes, and seems to be logically grouped with the other codes discussed by the board, list the numbers of those codes here. If the code is not grouped with others, enter "None" in this column. Note that this is a judgment call on your part. You may want to examine some example transcripts to better understand situations in which codes (and cases) should be grouped. (Examples: 61-9 [cited codes C4, C11], 79-5 [cited codes 3(f), 3(g)])

B.8.5. Overrides

If the board mentions that the violation or non-violation of this code overrides the importance of another code or codes, list the overridden code numbers in this column. If no codes are overridden, enter "None" in this column. *Be Careful:* You should only indicate that one code overrides another if the board *explicitly* indicates that one code holds more importance than another in the context of the current fact situation. (Examples: 61-9 [cited codes C4, C11], 89-5 [cited code III.4.b.])

B.8.6. Why Relevant?

When you select the number of a code in the case, a representation of the cited code displays in the "reference" frame of the window. This representation contains statements of when this code is relevant, next to the label, "Code is relevant when". If a code has several parts, you may need to scroll through the representation to find the most appropriate statement of relevancy. Copy this statement from the code representation, and paste it into this column of the table, between the two caret characters (i.e., "^") provided in the transcript template. (You *can* retype all of the code text if you wish, but copy/paste will assure that no errors are introduced.) If there are any "AND" or "OR" connectors in the sequence of statements you copied to this column, delete them.

If you are unable to determine why the code was cited (often this occurs in the third, "neither supports nor conflicts", table) enter "Unknown" in this column. This may be because the code was erroneously cited by the board, or cited briefly with insufficient discussion.

After each statement, indicate in brackets (i.e., "[" and "]") the numbers of the facts that support it. If no fact from the chronology table directly supports it, but it is supported by text in the discussion, one of three values can go in the brackets instead of a number:

Hypo

Use this value when the board makes an *explicit* assumption in their analysis. There must be some text in the discussion section that supports the idea that this is an assumption. Keys to look for are the words "assuming that ...", "provided that ...", and "if we assume ..." etc. If you use the "Hypo:" value, you must also include the portion of the discussion that includes the assumption. Copy the relevant quote from the case, and paste it inside the brackets, after the value "Hypo:". You may edit the quote to remove parts of the discussion that are not relevant. (Examples: 58-1 [cited code C19], 84-1 [cited code III.8.])

Unstated assumption

Use this value when it appears that the board may have made an assumption, but did not explicitly state the assumption in the discussion section. For example, situations in which the board cites a particular code, but none of the explicit facts of the case match the conditions in the representation of the code. In this kind of situation, you may infer that the board has made "unstated assumption" that allows the code to be relevant, violated, or not violated. (Examples: 65-9 [cited code 2(b)], 84-1 [cited code III.8.])

Inference based on facts

Use this value when it appears that the board inferred a new fact or facts based on existing facts. Inside the brackets, list the facts that act as the basis for this inference, followed by the "Inference based on facts" designation. (Examples: 76-3 [cited code 4(a)], 84-1 [cited code III.1.f.])

B.8.7. Why Violated, Not Violated, Changed, or Not Applicable?

If the code was violated or not violated

Following the statement of relevancy in the code representation are statements delineating when a code is violated and not violated. Which set of statements you use in this column depends, of course, on whether or not the board has decided the code was violated. Select the most relevant statement, copy it, and paste it into this column of the table, between the two caret characters (i.e., "^") provided in the transcript template. (You *can* retype all of the code text if you wish, but copy/paste will assure that no errors are introduced.) If there are any "AND" or "OR" connectors in the sequence of statements you copied to this column, delete them.

You may also need to enter additional information to support the code status. For instance, sometimes a code seems to be violated, but the board argues that, for a reason that is not specified in the code table, the code is actually *not* violated. In this situation, the you should use the "violated" form of the rule, but add an extra phrase to explain why it is actually not violated. (For an example, see Case 83-1, question 2, code III.3.a.)

Sometimes, a code seems to be *not* violated, but is actually violated for some reason that is not specified in the code table. In this instance, you should choose the "not violated" form of the rule, but add an extra phrase to explain why it is actually violated.

After each statement, indicate in brackets (i.e., "[" and "]") the numbers of the facts that support it. If no fact from the chronology table directly supports it, but it is supported by text in the discussion, use one of the three values (i.e., Hypo, Unstated assumption, Inference based on facts) discussed in the "Why Relevant?" section, above.

Lastly, sometimes there are simply extra conditions that the board discusses but which are not specified in the code table. You should add these extra conditions.

If there is no text that can be copied and pasted into the table, you should construct a statement and type it into the table. This statement should have the same structure and style as the statements from the code representation. Any text you create yourself should be surrounded in percentage signs (i.e., "%" and "%") in the transcript file, e.g., "%The Engineer has the right to seek and accept other employment in his field. %" (Examples: 69-10 [cited code 12], 87-5 [cited code II.4.c.])

If the code has changed

If the board proposed that the code be changed, describe how the code was changed. Include the new text of the code, and if the information is given, how it differs from the old text. Remember to surround in percentage signs (i.e., "%" and "%") any text that you create yourself. (Examples: 58-1 [cited code C27], 79-5 [cited code 3(f)], 85-6 [cited code II.4.a.])

Unknown

If you entered "Unknown" in the "Why Relevant" column, you will obviously also be unable to determine content for this column, and should enter "Unknown" here as well. (Example: 92-1 [cited code III.5.a.])

Before you continue, you should save your transcript file.

B.9. Listing the Cited Cases

Just as you transcribed the codes referenced by the board in discussion, you must also transcribe the referenced cases into tables.

The tables of referenced cases go just beneath the code tables. Once again, they are used to transcribe the precedents used to support the conclusion, the precedents that conflict with the conclusion, and the precedents in the board's discussion that neither directly support nor directly conflict with the board's conclusion.

Each table has seven columns that detail how each case was used in the discussion.

B.9.1. Case

This column simply lists each case cited by the board. These case numbers are listed at the top of the case, so you can just copy them and paste them into the table, each starting its own row.

B.9.2. Citation Type

Use this column to record how the case was cited, whether as an analogous precedent, as a distinguishing precedent, or as a relevant, but not controlling case.

Analogous Precedent

An analogous precedent is a cited case that, by certain similarity to the current fact situation, argues for the same conclusion in both. A cited case should be labelled as an analogous precedent when: (1) the board cites it because it is very similar, at a thematic or perhaps detailed level, to the current fact situation and (2) the board argues that the conclusion in the cited case (i.e., ethical or unethical) should be followed in the current case. (*Important Note:* Notice that by this definition a cited case can be an analogous precedent even if its facts are quite different from the current case. The key is that the cases are similar at a conceptual or abstract level.) (Examples: 77-11 [cited cases 76-5, 75-15], 79-2 [cited case 63-6], 88-1 [cited case 69-13-C])

Distinguishing Precedent

A distinguishing precedent is a cited case that essentially provides a counter-example to the current fact situation; it supports an opposite conclusion. A cited case should be labelled as a distinguishing precedent when: (1) the board cites it because it is similar, at *either* a detailed or thematic level, to the current case, (2) despite the similarities, the board notes at least one key distinction between the cases, and (3) the board argues that the conclusion in the cited case (i.e., ethical or unethical) should *not* be followed in the current case. (Examples: 79-5 [cited case 72-11], 83-1 [cited case 77-11], 88-1 [cited case 85-6])

Relevant, But Not Controlling

A Relevant, But Not Controlling case is an earlier case that is relevant, in some way, to the current case but not to the extent that it "controls" or directly supports the board's decision in the current case. The earlier and current cases may share a general issue, principle, or perhaps some facts. However, with a "Relevant, But Not Controlling" citation the board does *not* argue that the current case should or should not follow the earlier conclusion. Typical situations in which to use this designation are: cases cited in passing, cases cited to illustrate a general point, and cases cited for any reason other than as analogous or distinguishing precedents. (*Important Note:* You should *only* use the "Relevant, But Not Controlling" tag in the transcript section with the heading: "... information that neither directly supports nor directly conflicts with their conclusion." By definition, "Relevant, But Not Controlling" cases can neither directly support nor directly conflict with the board's conclusion.) (Examples: 72-4 [cited cases 62-10, 62-18, 64-9], 85-4 [cited cases 76-3, 82-2, 82-6])

Unknown

If you cannot tell from the discussion that one of the above terms is more appropriate you can enter this value.

You may also need to include special status modifiers to indicate the relative importance of cited cases. If the board's discussion cites several cases (and/or codes), but one of the cases seems to have more bearing than the others (or other codes), include the "More Importance" modifier in the "Citation Type" column. If one or more of the cited cases seem to have less bearing, include the "Less Importance" modifier. These importance values are optional. They should only be used when the relative importance of cases is made obvious in the board's discussion. (Examples: 71-4 [cited case 63-6])

B.9.3. *How Cited*

There are two possible values for this column.

Explicitly Discussed

If the board cites the case, and also discusses its relevance to the case, enter this value in the "How Cited" column. (Examples: 92-9 [cited case 77-3], 92-4 [cited case 88-6])

Referenced Only

If the board cites the case, but does not discuss its relevance to the case, enter this value in the "How Cited" column. This can occur when the board cites a case in passing but then does not discuss it, other than to broadly categorize it. (Example: 67-1 [cited cases 62-7, 62-21, 63-5])

B.9.4. Grouped With

If the case is mentioned along with other cases, list the numbers of those cases here. If the case is not grouped with others, enter "None" in this column. (Examples: 92-4 [cited cases 65-12, 82-5], 92-6 [cited cases 89-7, 90-5])

B.9.5. Q #

The cited case usually will be cited as part of the discussion concerning a particular questioned fact. List the number of the questioned fact in this column.

B.9.6. Why Relevant?

Since for cases there is no analog to the code representation, you will need to write a statement (or statements) in your own words that expresses the relevancy of this case to the current case. These statements should have the same structure and style as the statements from the code representation, and they should be placed between the two caret characters (i.e., "^") provided in the transcript template. Also, as before, text you create yourself should be surrounded by percentage signs (i.e., "%"); place the percentage signs around each individual statement.

After the statement of relevancy, indicate in brackets (i.e., "[" "]") the numbers of the facts that support your choice.

If you are unable to determine why the case was cited (often this occurs in the third, "neither supports nor conflicts", table) enter "Unknown" in this column. This may be because the code was erroneously cited by the board, or cited briefly with insufficient discussion.

After each statement, indicate in brackets (i.e., "[" and "]") the numbers of the facts that support it. If no fact from the chronology table directly supports it, but it is supported by text in the discussion, one of three values can go in the brackets instead of a number (note that these are the same values that can be used with cited codes):

Hypo

> Use this value when the board makes an *explicit* assumption in their analysis. There must be some text in the discussion section that supports the idea that this is an assumption. Keys to look for are the words "assuming that ...", "provided that ...", and "if we assume ..." etc. If you use the "Hypo:" value, you must also include the portion of the discussion that includes the assumption. Copy the relevant quote from the case, and paste it inside the brackets, after the value "Hypo:". You may edit the quote to remove parts of the discussion that are not relevant. (Examples: 65-9 [cited case 63-6], 79-2 [cited case 65-9])

Unstated assumption

Use this value when it appears that the board may have made an assumption, but did not explicitly state the assumption in the discussion section. In this kind of situation, you may infer that the board has made an "unstated assumption" that allows the case to be relevant, violated, or not violated. (Examples: 79-2 [cited case 65-9], 88-7 [cited case 63-6])

Inference based on facts

Use this value when it appears that the board inferred a new fact or facts based on existing facts. Inside the brackets, list the facts that act as the basis for this inference, followed by the "Inference based on facts" designation. (Examples: 72-4 [cited case 62-10], 77-11 [cited case 75-15])

B.9.7. Why Distinguished or Analogous?

Following the statement of relevancy are statements delineating how a cited case is similar (analogous) or different (distinguished) from the current case.

Again, you will need to write a statement (or statements) in your own words that expresses the distinguishing or analogous points. These statements should have the same structure and style as the statements from the code representation, and they should be placed between the two caret characters (i.e., "^") provided in the transcript template. Also, don't forget to surround each *individual* statement with percentage signs (i.e., "%").

After the statement, indicate in brackets (i.e., "[" "]") the numbers of the facts that support your statement. Alternatively, you can indicate one of the three values discussed in the previous section (i.e., Hypo, Unstated assumption, Inference based on facts).

If you are unable to determine why the case was cited (often this occurs in the third, "neither supports nor conflicts", table) enter "Unknown" in this column. This may be because the code was erroneously cited by the board, or cited briefly with insufficient discussion.

If the cited case is tagged as "Relevant, But Not Controlling," there likely will not be any points to enumerate here. For these cases, enter "NA" for "Not Applicable" in this column.

Before you continue, you should save your transcript file.

B.10. Submitting the Transcript

Once you have completed your transcript, read through it one last time to assure that:

1. you have included all of the necessary parts,
2. the transcript adheres to the required structure and syntax, and

3. the transcript expresses the case as well as it possibly can.

If you make any changes at this point, save your file one last time.

You will be assigned transcripts in groups of 5, 10, 15, etc. by the exercise coordinator. While you are working on an individual transcript of a particular assignment you can store it as a single file. However, when you are ready to submit your completed work, please append all of the individual files together as a single MS Word 6.0 file to be shipped to the exercise coordinator. Name the file using your last name, followed by the extension ".DOC".

Using your email program, create a new message addressed to the exercise coordinator. Include the single file containing all of your individual transcripts as an attachment to that piece of email. If you have any comments or questions about your transcript, the web site, the case, or the transcription process, you may include them in the body of the message.

The last step is sending your message. Once you send the message, you must wait until the exercise coordinator contacts you again by email. If there are any "bugs" in your transcript (e.g., invalid syntax, missing information), the file will be returned to you for revisions. Not until your transcript file is bug-free will it be considered complete. After completing a set of transcriptions, wait until the exercise coordinator provides you with a new set of cases to transcribe.

B.11. Conclusion

Thank you for participating in the Ethics Case Transcription Study.

If this is your first transcript, you may find it helpful to view further example transcriptions, and perhaps try a few more transcripts yourself. A collection of example transcripts is available through the Reference Shelf area of this web site.

Try transcribing one or more of these cases yourself, and compare your results to the example on this web site. Even if you do not choose to do a full practice transcript, reading through several examples will give you a better idea of what a transcription is like, and how the parts of a transcript are constructed. If you take the time to familiarize yourself with the style of the examples, creating your own transcripts will be more a matter of imitation rather than creation from scratch.

Thank you again for participating, and best of luck with your transcription tasks.

Bibliography

References

[Aleven, 1997] Vincent A. W. M. M. Aleven. *Teaching Case-Based Argumentation Through a Model and Examples*. Ph.D. Dissertation, University of Pittsburgh, 1997.

[Allen, 1983] James F. Allen. Maintaining Knowledge about Temporal Intervals. In the *Communications of the ACM* 26(11), 832-843, 1983.

[Arras, 1991] John D. Arras. Getting Down to Cases: The Revival of Casuistry in Bioethics. In *Journal of Medicine and Philosophy*, 16, 29-51, 1991.

[Aristotle, edited and published in 1924] Aristotle. *Nicomachean Ethics*. W. D. Ross, editor, Oxford, 1924.

[Ashley, 1990] Kevin D. Ashley. *Modeling Legal Argument: Reasoning with Cases and Hypotheticals*. Based on a Ph.D. Dissertation, University of Massachusetts, 1987, COINS Technical Report No. 88-01. Cambridge: MIT Press, 1990.

[Ashley and McLaren, 1995] Kevin D. Ashley and Bruce M. McLaren. Reasoning with Reasons in Case-Based Comparisons. In the *Proceedings of the First International Conference on Case-Based Reasoning*, Sesimbra, Portugal, 1995.

[Ashley and McLaren, 1994a] Kevin D. Ashley and Bruce M. McLaren. A CBR Knowledge Representation for Practical Ethics. In the *Proceedings of the Second European Workshop on Case-Based Reasoning*, *(EWCBR)*, Chantilly, France, 1994.

[Ashley and McLaren, 1994b] Kevin D. Ashley and Bruce M. McLaren. Evaluating Comparative Evaluation Models. In the *Working Notes, AAAI Workshop on Case-Based Reasoning*, David Aha, editor, 164-171, 1994.

[Bareiss, 1989] E. Ray Bareiss. *Exemplar-Based Knowledge Acquisition - A Unified Approach to Concept Representation, Classification, and Learning*. San Diego, CA: Academic Press, 1989.

[Beauchamp and Childress, 1978] T. L. Beauchamp and J. F. Childress. *Principles of Biomedical Ethics*. New York: Oxford University Press, First edition, 1978.

[Berman and Hafner, 1986] Donald H. Berman and Carole D. Hafner. Obstacles to the Development of Logic-Based Models of Legal Reasoning. In *Computer Power and Legal Language*. C. Walter, editor, 185-214, Greenwood Press, 1986

[Berman and Greiner, 1980] Harold J. Berman and William R. Greiner. *The Nature and Functions of Law*. Mineola, NY: Foundation Press, 1980.

[Bok, 1989] Sissela Bok. *Lying*: *Moral Choice in Public and Private Life*. New York: Random House, Inc. Vintage Books, 1989.

[Branting, 1994] L. Karl Branting. A Computational Model of Ratio Decidendi. *Artificial Intelligence and Law* 2:1-31. The Netherlands: Kluwer Academic Publishers, 1994.

[Branting, 1993] L. Karl Branting. A Reduction-Graph Model of Ratio Decidendi. In the *Proceedings of the Fourth International Conference on Artificial Intelligence and Law (ICAIL)*, Amsterdam, The Netherlands, 1993.

[Branting, 1992] L. Karl Branting. A Model of the Role of Expertise in Analog Retrieval. In the *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society.* Bloomington, Indiana, 1992.

[Branting, 1991] L. Karl Branting. Building Explanations from Rules and Structured Cases. *International Journal of Man-Machine Studies*, 34 (6): 797-837, 1991.

[Brill, 1993] David Brill. *Loom Reference Manual, Version 2.0.* University of Southern California, 1993.

[Brüninghaus and Ashley, 1999] Stefanie Brüninghaus and Kevin D. Ashley. Bootstrapping Case Base Development with Annotated Case Summaries. In the *Proceedings of the Third International Conference on Case-Based Reasoning* (ICCBR-99). Seeon Monastery, Germany, 1999.

[Brüninghaus and Ashley, 1998] Stefanie Brüninghaus and Kevin D. Ashley. Developing Mapping and Evaluation Techniques for Textual CBR. In the *Proceedings of the AAAI-98 Workshop on Case-Based Reasoning*, 1998.

[Bunke and Messmer, 1993] H. Bunke, H. and B. T. Messmer. Similarity Measures for Structured Representations. In the *Proceedings of the First European Workshop on Case-Based Reasoning*. Kaiserslautern, Germany, 1993.

[Burke *et al.*, 1997] Robin D. Burke, Kristian J. Hammond, Vladimir Kulyukin, Steven L. Lytinen, Noriko Tomuro, and Scott Schoenberg. Question Answering from Frequently Asked Question Files: Experiences with the FAQ Finder System. *AI Magazine*, Volume 18, No. 2, Summer, 1997.

[Cavalier and Covey, 1996] Robert Cavalier and Preston K. Covey. *A Right to Die? The Dax Cowart Case* CD-ROM Teacher's Guide, Version 1.0, Pittsburgh, PA: Center for the Advancement of Applied Ethics, Carnegie Mellon University, 1996

[Cheng *et al.*, 1996] Chun Hung Cheng, Clyde W. Holsapple, and Anita Lee. Citation-Based Journal Rankings for AI Research: A Business Perspective. *AI Magazine*, Volume 17, No. 2, Summer, 1996.

[Colby and Kohlberg, 1987a] A. Colby and L. Kohlberg. *The Measurement of Moral Judgment. Vol. 1: Theoretical Foundations and Research Validation*. New York: Cambridge University Press, 1987.

[Colby and Kohlberg, 1987b] A. Colby and L. Kohlberg. *The Measurement of Moral Judgment. Vol. 2: Standard Issue Scoring Manual*. New York: Cambridge University Press, 1987.

[Davis *et al.*, 1977] R. Davis, B. G. Buchanan, and E. H. Shortliffe. Production Rules as a Representation for a Knowledge-Based Consultation System. *Artificial Intelligence* 8:15-45, 1977.

[Davison and Hinkley, 1997] A. C. Davison and D. V. Hinkley. *Bootstrap Methods and Their Application*. Cambridge University Press, 1997.

[Efron and Tibshirani, 1993] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. San Francisco: Chapman and Hall, 1993.

[Evans, 1968] Thomas Evans. A Program for the Solution of a Class of Geometric Analogy Intelligence Test Questions. In *Semantic Information Processing*, M. Minsky, editor, 271-353. Cambridge, MA: MIT Press, 1968.

[Forbus *et al*., 1994] Kenneth D. Forbus, Dedre Gentner, and Keith Law. MAC/FAC: A Model of Similarity-based Retrieval. *Cognitive Science* 19, 141-205, Norwood, NJ: Ablex Publishing Corporation, 1994.

[Frakes and Baeza-Yates, 1992] W. Frakes and R. Baeza-Yates, Editors. *Information Retrieval: Data Structures and Algorithms*. Englewood Cliffs, NJ: Prentice- Hall, 1992.

[Gardner, 1987] Anne Gardner. *An Artificial Intelligence Approach to Legal Reasoning*. Cambridge, MA: MIT Press, 1987.

[Gentner, 1983] Dedre Gentner. Structure-mapping: A Theoretical Framework for Analogy. *Cognitive Science*, 7, 155-170, 1983.

[Gilligan, 1982] Carol Gilligan. *In a Different Voice*. Cambridge, MA: Harvard University Press, 1982.

[Ginsberg, 1993] Matt Ginsberg. *Essentials of Artificial Intelligence*. San Francisco, CA: Morgan Kaufmann Publishers, Inc., 1993.

[Hammond, 1989] Kristian J. Hammond. *Case-Based Planning: Viewing Planning as a Memory Task*. San Diego, CA: Academic Press, Inc., 1989.

[Harris *et al*., 1999] Charles E. Harris, Michael S. Pritchard, and Michael J. Rabins. *Engineering Ethics: Concepts and Cases.* Belmont, CA: Wadsworth, a division of Thomson Learning, Second edition, 1999.

[Hinrichs, 1992] T. R. Hinrichs. *Problem Solving in Open Worlds: A Case Study in Design*. Northvale, NJ: Erlbaum, 1992.

[Holyoak and Thagard, 1995] Keith J. Holyoak and Paul Thagard. *Mental Leaps: Analogy in Creative Thought*. Cambridge, MA: The MIT Press, 1995.

[Jaczynski and Trousse, 1998] Michel Jaczynski and Brigitte Trousse. BROADWAY: A World Wide Web Browsing Advisor Reusing Past Navigations from a Group of Users. In the *Proceedings of the Fourth European Workshop on Case-Based Reasoning*. Dublin, Ireland, 1998.

[Jaczynski, 1997] Michel Jaczynski. Framework for the Management of Past Experiences with Time-Extended Situations. In the *Proceedings of the Sixth ACM Conference on Information and Knowledge Management* (CIKM ’97), Las Vegas, Nevada, 1997.

[Jonsen and Toulmin, 1988] Albert R. Jonsen and Stephen Toulmin. *The Abuse of Casuistry: A History of Moral Reasoning*. Berkeley, CA: University of California Press, 1988.

[Kant, edited and published in 1969] Immanuel Kant. *Foundations of the Metaphysics of Morals with Critical Essays*. Robert Paul Wolff, editor, Indianapolis, IN: Bobbs-Merrill, 1969.

[Kass *et al*., 1986] Alex M. Kass, David B. Leake, and Christopher C. Owens. Swale: A Program that Explains. In *Explanation Patterns: Understanding Mechanically and Creatively*. Hillsdale, NJ: Erlbaum, 1986.

[Keane *et al*., 1994] Mark T. Keane, Tim Ledgeway, and Stuart Duff. Constraints on Analogical Mapping: A Comparison of Three Models. *Cognitive Science* 18, 387-438, 1994.

[Kolodner, 1993] Janet Kolodner. *Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1993.

[Korfhage, 1997] R. Korfhage. *Information Storage and Retrieval*. New York: Wiley, 1997.

[Koton, 1989] Phyllis Koton. *Using Experience in Learning and Problem Solving*. Ph.D. Dissertation, Massachusetts Institute of Technology, 1988, Technical Report 441, MIT Laboratory for Computer Science, Cambridge, MA, 1989.

[Koomen, 1989] Johannes A. G. M. Koomen. *The TIMELOGIC Temporal Reasoning System*. Technical Report 231, Computer Science Department, the University of Rochester, NY, 1989.

[Leake, 1991] David B. Leake. An Indexing Vocabulary for Case-Based Explanation. In the *Proceedings of AAAI-91*, 1991.

[Lenz, 1998] Mario Lenz. Defining Knowledge Layers for Textual Case-Based Reasoning. In the *Proceedings of the Fourth European Workshop on Case-Based Reasoning*. Dublin, Ireland, 1998.

[Lewis et al., 1996] David D. Lewis, Robert E. Schapire, James P. Callan, and Ron Papka. Training Algorithms for Linear Text Classifiers. In the *Proceedings of the 19th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*. Zurich, Switzerland, 1996.

[Llewellyn, 1930] K. N. Llewellyn. *The Bramble Bush. On Our Law and its Study*. New York: Oceana Publications, 1930.

[Lytinen, 1992] Steven L. Lytinen. Conceptual Dependency and its Descendants. In *Semantic Networks in Artificial Intelligence*, Fritz Lehmann, editor, Great Britain: Pergamon Press, 1992.

[MacGregor, 1990] Robert MacGregor. The Evolving Technology of Classification-Based Knowledge Representation Systems. In *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, Chapter 13, John Sowa, editor, San Mateo, CA: Morgan Kaufmann, 1990.

[Martin, 1998] Judith Martin. *Miss Manners' Basic Training: The Right Thing to Say*. Crown Publishing, 1998.

[McLaren and Ashley, 1999] Bruce M. McLaren and Kevin D. Ashley. Case Representation, Acquisition, and Retrieval in SIROCCO. In the *Proceedings of the Third International Conference on Case-Based Reasoning* (ICCBR-99). Seeon Monastery, Germany, 1999.

[McLaren and Ashley, 1998] Bruce M. McLaren and Kevin D. Ashley. Exploring the Dialectic Between Abstract Rules and Concrete Facts: Operationalizing Principles and Cases in Engineering Ethics. In the *Proceedings of the Fourth European Workshop on Case-Based Reasoning*. Dublin, Ireland, 1998.

[McLaren and Ashley, 1995a] Bruce M. McLaren and Kevin D. Ashley. Case-Based Comparative Evaluation in TRUTH-TELLER. In the *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*. Pittsburgh, PA, 1995.

[McLaren and Ashley, 1995b] Bruce M. McLaren and Kevin D. Ashley. Context Sensitive Case Comparisons in Practical Ethics: Reasoning about Reasons. In the *Proceedings of the Fifth International Conference on Artificial Intelligence and Law*, College Park, MD, 1995.

[Mill, edited and published in 1979] John Stuart Mill. *Utilitarianism*. G. Sher, editor, Indianapolis, IN: Hackett, 1979.

[Miller *et al*., 1990] G. Miller, R. Beckwith, C. Fellbaum, G. Gross, and K. Miller. Introduction to WordNet: An On-Line Lexical Database. *International Journal of Lexicography* 3:235-244, 1990.

[Minton *et al*., 1990] Steven Minton, Jaime G. Carbonell, Craig A. Knoblock, Daniel R. Kuokka, Oren Etzioni, and Yolanda Gil. Explanation-Based Learning: A Problem Solving Perspective. In *Machine Learning: Paradigms and Methods*. 63-118. J. G. Carbonell, editor, Cambridge, MA: The MIT Press, 1990.

[Minton, 1988] Steven Minton. *Learning Effective Search Control Knowledge: An Explanation-Based Approach*. Ph.D. Dissertation, Carnegie Mellon University, 1988.

[Mostow, 1983] J. Mostow. Machine transformation of advice into a heuristic search procedure. In *Machine Learning*, vol. 1, 1983.

[Muñoz-Avila and Huellen, 1995] Héctor Muñoz-Avila and Jochem Huellen. Retrieving Cases in Structured Domains by Using Goal Dependencies. In the *Proceedings of the First International Conference on Case-Based Reasoning*, Sesimbra, Portugal, 1995.

[Neustadt and May, 1986] R. E. Neustadt and E. R. May. *Thinking in Time*. New York: Free Press, 1986.

[NSPE, 1958-1998] National Society of Professional Engineers. Opinions of the Board of Ethical Review, Volumes I through VIII. Alexandria, VA: the National Society of Professional Engineers, 1958-1998.

[NSPE, 1996] National Society of Professional Engineers. *The NSPE Ethics Reference Guide*. Alexandria, VA: the National Society of Professional Engineers, 1996.

[Pinkus *et al.*, 1997] R. L. Pinkus, L. J. Shuman, and N. P. Hummon. *Engineering Ethics: Balancing Cost, Schedule and Risk-Lessons Learned from the Space Shuttle*. New York: Cambridge University Press, 1997.

[Rissland *et al.*, 1997] Edwina L. Rissland, David B. Skalak, and M. Timur Friedman. Evaluating a Legal Argument Program: The BankXX Experiments. *Artificial Intelligence and Law* Volume 5. The Netherlands: Kluwer Academic Publishers, 1997.

[Rissland *et al.*, 1996] Edwina L. Rissland, David B. Skalak, and M. Timur Friedman. BankXX: Supporting Legal Arguments through Heuristic Retrieval. *Artificial Intelligence and Law* Volume 4. The Netherlands: Kluwer Academic Publishers, 1996.

[Rissland *et al.*, 1993] Edwina L. Rissland, David B. Skalak, and M. Timur Friedman. BankXX: A Program to Generate Argument through Case-Based Search. In the *Proceedings of the Fourth International Conference on Artificial Intelligence and Law (ICAIL)*, Amsterdam, The Netherlands, 1993.

[Rissland and Skalak, 1991] Edwina L. Rissland and David B. Skalak. CABARET: Statutory Interpretation in a Hybrid Architecture. In *International Journal of Man-Machine Studies*, 34 (6): 839-887, 1991.

[Schank, 1972] Roger C. Schank. Conceptual Dependency: A Theory of Natural Language Understanding. *Cognitive Psychology* 3, 552-631, 1972.

[Searing, 1998] Donald R. Searing. *HARPS Ethical Analysis Methodology, Method Description*. Version 2.0.0., Lake Zurich, IL: Taknosys Software Corporation, 1998.

[Sergot *et al.*, 1986] M. J. Sergot, F. Sadri, R. A. Kowalski, F. Kriwaczek, P. Hammond, and H. T. Cory. The British Nationality Act as a Logic Program. In the *Communications of the ACM*, 29(5): 370-386, 1986

[Smyth and McKenna, 1999] Barry Smyth and Elizabeth McKenna. Building Compact Competent Case-Bases. In the *Proceedings of the Third International Conference on Case-Based Reasoning* (ICCBR-99). Seeon Monastery, Germany, 1999.

[Spellman and Holyoak, 1993] Barbara A. Spellman and Keith J. Holyoak. An Inhibitory Mechanism for Goal-Directed Analogical Mapping. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, 947-952. Hillsdale, NJ: Erlbaum, 1993.

[Strong, 1988] C. Strong. Justification in Ethics. In Baruch A. Brody, editor, *Moral Theory and Moral Judgments in Medical Ethics*, 193-211. Dordrecht: Kluwer Academic Publishers, 1988.

[Thagard *et al.*, 1990] P. Thagard, K. J. Holyoak, G. Nelson, and D. Gochfeld. *Analog Retrieval by Constraint Satisfaction*. Technical Report CSL-Report 41, Princeton University, 1990.

[Toulmin, 1958] Stephen E. Toulmin. *The Uses of Argument*. Cambridge, England: Cambridge University Press, 1958.

[Twining and Miers, 1976] William Twining and David Miers. *How to Do Things With Rules*. London: Cox and Wyman, Ltd, 1976.

[Van Buren, 1989] Abigal Van Buren. *The Best of Dear Abby*. Andrews McMeel Publishing, Second edition, 1989.

[van Rijsbergen, 1979] C.J. van Rijsbergen. *Information Retrieval*. London: Butterworths, Second edition, 1979.

[Veloso, 1992] Manuela M. Veloso. *Learning by Analogical Reasoning in General Problem Solving*. Ph.D. Dissertation, Carnegie Mellon University, 1992.

[Vila and Yoshino, 1998] Lluís Vila and Hajime Yoshino. Time in Automated Legal Reasoning. *Information and Communications Technology Law*, Vol. 7, No. 3, 1998.

[Warnock, 1971] G. J. Warnock. *The Object of Morality*. London: Methuen and Co., 1971.

[Winston, 1980] Patrick H. Winston. Learning and Reasoning by Analogy. *Communications of the ACM* 23:689-703, 1980.

[Witten *et al*., 1999] Ian H. Witten, Alistair Moffat, and Timothy C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. San Francisco, CA: Morgan Kaufmann Publishers, Inc., Second edition, 1999.

[Yen *et al*., 1991] John Yen, Robert Neches, and Robert MacGregor. CLASP: Integrating Term Subsumption Systems and Production Systems. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 3, No. 1, March, 1991.

[Zobel and Moffat, 1998] Justin Zobel and Alistair Moffat. Exploring the Similarity Space. *SIGIR Forum*, Vol. 32, No. 1, 18-34. New York: ACM Press, 1998.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

Y

Z