CHAPTER 5

LARGE-SCALE CONCEPT SPACE GENERATION

5.1 Objectives

Chapter 4 demonstrates the ability of the concept exploration process to help users elaborate information needs by using various manual and automatically generally knowledge sources. This chapter focuses on how large domain-specific knowledge sources (concept spaces) can be generated automatically and investigates their quality compared with that of corresponding domain-specific man-made thesauri. In addition, the chapter addresses the feasibility and scalability issues of large-scale concept space generation in terms of domain knowledge and computing resources.

Concept space generation has been the semantic research component for the Illinois Digital Library Initiative (DLI) project, entitled: "Building the Interspace: Digital Library Infrastructure for a University Engineering Community," one of six projects recently funded by NSF/ARPA/NASA. The goal of the project is to transform the Internet into the *Interspace*, in particular by bringing professional and "intelligent" search and display of structured documents to the Net. To address the scalability issue arising from the continuous growth of digital library

repositories, this chapter presents in detail a parallel computing approach to creating concept spaces for semantic retrieval. The Illinois DLI project provides the large-scale testbed for this research.

5.2 Research Questions and Methodology

The specific research questions to be investigated were:

- **Question 1**: With regard to computing scalability, would the technique of computer generation of concept spaces be applicable to very large textual databases?
- **Question 2**: With regard to domain specific knowledge scalability, would concept space generation by technology create satisfactory domain-specific concept associations from corresponding textual databases?
- **Question 3**: How does the quality of concept associations in concept space generated from very large textual databases compare with that of a man-made domain-specific thesaurus?

I used the systems development methodology to iterate the system development cycle using personal-sized Unix workstations, workgroup-sized Unix servers, and national resources of a high performance and parallel computing facility. I also used the experimental design methodology to collect quantitative measures - concept recall and precision.

5.3 Background and Issues

5.3.1 Alleviating the Vocabulary Problem Using Concept Spaces

The *vocabulary (difference) problem* in human-computer interactions has been studied extensively in recent years (Furnas et al., 1987), who found that in spontaneous word choice for objects in five domains, two people favored the same term with less than 20% probability. This fundamental property of language limits the success of various design methodologies for vocabulary-driven interaction. In information science, indexing and search uncertainty have been recognized as the primary sources of information retrieval problems. Previous research (Bates, 1986) has shown that different indexers, well trained in an indexing scheme, might assign index terms for a given document differently. It has also been observed that an indexer might use different terms for the same document at different times (possibly because of learning or the cognitive state of mind at indexing). A high degree of uncertainty with regard to search terms has also been reported: searchers tend to use different terms for the same information sought. Because of the indeterminism involved in indexing and searching, an exact match between the searcher's terms and those of the indexer is unlikely (Chen and Dhar, 1987). This often results in poor recall and precision in search.

The vocabulary problem affects every domain of human knowledge. Based on research over the past few decades, it has become clear to information scientists

that development of effective online information retrieval systems must consider the cognitive processes and the vocabulary association characteristics of the users.

5.3.2 Concept Association and Thesaurus Work

According to Belkin, users of information retrieval systems bring with them a problem statement which represents an information need. Inherent in all information needs are "anomalous states of knowledge" (ASKs) (Belkin et al., 1982). In a document retrieval system based on ASKs, the searcher's state of knowledge is represented as a network of associations between words. From the structure and characteristics of the network, it is possible to identify anomalies in the state of knowledge. Several models of human memory association wherein knowledge is represented by network-like structures with linked propositions have been suggested. Anderson's work in human memory is particularly pertinent to term associations in retrieval (Anderson, 1985b). According to Anderson, people remember not the exact wording of verbal communication, but the meaning underlying it. The smallest unit of knowledge that can stand as an assertion bearing meaning is the proposition. Memory, then, is represented as a network of such propositions. The strength of the association paths leading to that piece of information contributes to the level of activation being spread. This theory of *spreading activation* has influenced the design of many semantic network based information retrieval systems (Chen et al., 1993).

Many research groups have created vocabulary-based search aids for online information retrieval systems by making use of existing thesauri or dictionaries. Thesauri, in particular, exhibit a structure similar to human word-association networks. While these tools are able to provide the searcher with alternate terms to use in searching, they do not overcome the *knowledge acquisition bottleneck*: the cognitive demand required of humans (indexers or domain experts) to create thesauri or dictionaries in the first place. An alternative approach to creating vocabulary-based search aids is based on *automatic thesaurus generation*.

- **Incorporating existing thesauri:**

  Fox et al. focused on creation of so-called "relational thesauri." For example, CODER adopted the *Handbook of Artificial Intelligence* and *Collin's Dictionary* (Fox et al., 1988). Ahlswede and Evens parsed (Ahlswede and Evens, 1988) *Webster's Seventh New Collegiate Dictionary* to obtain a "lexical database" containing lexical or lexical-semantic relationships from the dictionary definitions. Lesk converted an online version of Murray's *Oxford English Dictionary* into a thesaurus-like tool to facilitate searching of historical manuscripts. These approaches represent attempts to produce "universal lexicons," rather than domain-specific thesauri or dictionaries. Chen et al. conducted a series of experiments which included several large-scale, domain-specific thesauri. In (Chen and Dhar, 1991), Chen and Dhar incorporated a

portion of the *Library of Congress Subject Headings* (LCSH) in the computing area into a system that used a branch-and-bound spreading activation algorithm to assist users in query formulation. More recently, they developed concept-based document retrieval using multiple thesauri: two existing thesauri (LCSH and the ACM Computing Review Classification System) and an automatically-generated computing-specific thesaurus (Chen et al., 1993). The National Library of Medicine's *Unified Medical Language System (UMLS)* project is probably the largest-scale effort adopting existing domain-specific knowledge sources or thesauri in information access. It aims to build an intelligent automated system that understands biomedical terms and their interrelationships and uses this understanding to help users retrieve and organize information from multiple online sources (Lindberg and Humphreys, 1990).

- **Automatic thesaurus generation:**

  Numerous investigators have developed algorithmic approaches to *automatic thesaurus generation*. Most of these approaches employ techniques that compute coefficients of "relatedness" between terms using statistical co-occurrence algorithms (e.g., cosine, Jaccard, Dice similarity functions) (Chen and Lynch, 1992), (Salton, 1988), (Rasmussen, 1992). Some algorithms, however, perform cluster analysis to further group terms of similar meanings (Rasmussen,

1992). Other algorithms, such as latent semantic indexing (Dumais, 1994), perform statistical analysis to identify important semantic descriptors. Stiles (Stiles, 1961) was one of the early researchers who reported improved retrieval performance using a method based on term association (with collections of librarian-applied subject tags). Doyle (Doyle, 1962) further argued that the principles underlying association-based retrieval should apply whether the associations are determined by humans or by machines (programs).

More recently, Crouch and Yang (Crouch and Yang, 1992) automatically generated thesaurus classes from text keywords, which can subsequently be used to index documents and queries. Crouch's approach is based on Salton's vector space model and the term discrimination theory.

5.4 **Concept Space Techniques**

Based on the human information processing theory (Newell and Simon, 1972), (Card et al., 1983), (Anderson, 1985a), Chen et al. argued that creating robust and useful domain-specific thesauri (not universal thesauri) automatically requires a clear understanding of the following seven system development principles: *logarithmic vocabulary growth, completeness, term specificity, asymmetric association, relevance feedback, vocabulary overlapping, and spreading activation* (Chen et al., 1997) Selected algorithms for automatic thesaurus generation have been developed based on these seven principles. The specific steps and algorithms adopted

include: *document and object list collection*, *object filtering*, *automatic indexing*, *co-occurrence analysis*, and *associative retrieval*.

The following presents a brief overview of these techniques in the context of this research experiment. For algorithmic details, readers are referred to (Chen and Lynch, 1992), (Chen et al., 1993), (Chen et al., 1995).

5.4.1 Document and object list collection

In any automatic thesaurus building effort, the first task is to identify complete and recent collections of documents in specific subject domains that can serve as the sources of vocabularies. The proliferation of Internet services and the availability of online bibliographic databases have made document collection much easier.

In (Bates, 1986), Bates proposed a design model for subject access in online catalogs. She stressed the importance of building domain-specific lexicons for online retrieval purposes. A domain-specific, controlled list of keywords can help identify legitimate search vocabularies and help searchers "dock" on to the retrieval system. For most domain-specific databases, there appear always to be some existing lists of subject descriptors (e.g., the subject indexes at the back of a textbook), researchers' names (e.g., author indexes or researchers' directories), and other domain-specific objects (e.g., genes, experimental methods, organizational names, etc.) which exist online or can be obtained through OCR scanning.

5.4.2 Object Filtering

For each online document, terms are first identified and then matched with terms in known vocabularies, a process referred to as *object filtering*. Because the texts remaining after object filtering may still contain many important concepts, an automatic indexing procedure, which includes dictionary look-up, stop-wording, word stemming, and term phrase formation, then follows (Salton, 1988).

5.4.3 Automatic Indexing

Automatic indexing begins with stop-wording and continues with term phrase formation. The algorithm first identifies individual words and non-word tokens from free text. Next, a stop-word list is used to remove non-semantic bearing words (e.g., "the", "a", "on", "in"). After removing stop words, a similar procedure is adopted to remove verbs and adverbs. Finally, term phrase formation is accomplished by combining adjacent words. This approach is different from Salton's (Salton, 1988), which forms term phrases with non-stop words from various word positions in a sentence. The main objective of the process of *automatic indexing* is to extract useful *concepts* (*semantics*) from unstructured (free) text of document records.

5.4.4 Co-occurrence Analysis

After terms have been identified in each document, The analysis process computes the term frequency and the document frequency for each term in a document. Term frequency, tf_{ij}, represents the number of occurrences of term j in document i. Document frequency, df_j, represents the number of documents in a collection of n documents in which term j occurs. A few changes in the standard *term frequency* and *inverse document frequency* measures have been made based on the experience gained from the system development process.

Usually terms identified from the title of a document are more descriptive than terms identified from the abstract of the document. In addition, terms identified by the object filters are usually more accurate than terms generated by automatic indexing. Adjustment in determining the weights of importance can be made according to the requirements and needs of different domains and applications.

The combined weight of term j in document i, d_{ij}, is computed based on the product of "term frequency" and "inverse document frequency" as follows:

$$d_{ij} = tf_{ij} \times \log(\frac{N}{df_j} \times w_j)$$

where N represents the total number of documents in a collection and w_j represents the number of words in descriptor j. Multiple-word terms are assigned heavier

weights than single-word terms because multiple-word terms usually convey more precise semantic meaning than single-word terms.

Term co-occurrence analysis is performed based on the asymmetric "Cluster Function" developed by Chen and Lynch (Chen and Lynch, 1992).

$$W_{jk} = \frac{\sum_{i=1}^{n} d_{ijk}}{\sum_{i=1}^{n} d_{ij}} \times WeightingFactor(k)$$

$$W_{kj} = \frac{\sum_{i=1}^{n} d_{ikj}}{\sum_{i=1}^{n} d_{ik}} \times WeightingFactor(j)$$

W_{jk} indicates the similarity weights from term j to term k and W_{kj} indicates the similarity weights from term k to term j. d_{ij} and d_{ik} were calculated based on the equation in the previous step. d_{ijk} and d_{ikj} represent the combined weight of both descriptors j and k in document i. However, they are computed slightly differently due to their different starting terms. They are defined as follows:

$$d_{ijk} = tf_{ijk} \times \log\left(\frac{N}{df_{jk}} \times w_j\right)$$

$$d_{ikj} = tf_{ijk} \times \log\left(\frac{N}{df_{jk}} \times w_k\right)$$

where tf_{ijk} represents the number of occurrences of both term j and term k in document i (the smaller number of occurrences between the terms is chosen). df_{jk} represents the number of documents (in a collection of N documents) in which terms j and k occur together. w_j represents the number of words of descriptor j and w_k represents the number of words of descriptor k (thus descriptors with multiple words receive higher weights).

In order to *penalize* general terms (terms which appeared in many places) in the co-occurrence analysis, the following weighting scheme, which is similar to the *inverse document frequency* function, is applied:

$$WeightingFactor(k) = \frac{\log \frac{N}{df_k}}{\log N}$$

$$WeightingFactor(j) = \frac{\log \frac{N}{df_j}}{\log N}$$

Terms with a higher df_k or df_j value (more general terms) have a smaller weighting factor value, which causes the co-occurrence probability to become smaller. In effect, general terms are *pushed* down in the co-occurrence table (terms in the co-occurrence table are presented in reverse probabilistic order, with more relevant terms appearing first).

5.4.5 Associative Retrieval

In addition to the user-controlled thesaurus browsing process, searchers can also invoke selected spreading activation algorithms for multiple-term, multiple-link term suggestions. The previous chapter presents in detail the use of the serial branch-and-bound algorithm and the parallel Hopfield net algorithm for the spreading activation process. The Hopfield algorithm, in particular, has been shown to be ideal for concept-based information retrieval.

5.5 **Parallel Computing Approach**

Over the past decade, use of parallel computing for information retrieval gradually has progressed from active research to commercial applications (Rasmussen, 1991). Many new classes of algorithms and applications have emerged and created unique opportunities and challenges, especially in the context of Grand Challenge Applications and National Information Infrastructures (Wah, 1993).

5.5.1 Parallel Computing for Information Retrieval

Parallel computing is defined as information processing that emphasizes concurrent manipulation of data belonging to one or more processes solving a single problem. Two classes of architecture have been used to distinguish between different parallel supercomputers: SIMD (single instruction stream, multiple data stream) and MIMD (multiple instruction stream, multiple data stream). In SIMD

machines (e.g., MasPar MP-1), one control processor broadcasts a single instruction stream to all the other processors simultaneously for execution on different data streams. In MIMD machines (e.g., Thinking Machine's CM-5), each processor has its own independent program instruction stream. While this classification has been useful in distinguishing between supercomputing architectures, many other classification schemes have recently been proposed.

In (Rasmussen, 1991), Rasmussen suggested three approaches to parallel computing in information retrieval (IR): development and testing of parallel IR algorithms, design of special-purpose parallel hardware for IR applications (e.g., database machines), and development of distributed systems for database access. The first approach, in particular, is of most relevance to this research.

A major focus of research in the 1980s was on the adaptation and refinement of existing popular IR algorithms to parallel processors. Algorithms based on pattern matching (string matching) algorithms or text signatures (superimposed coding) and inverted index file algorithms have attracted most attention (Rasmussen, 1991). Text signatures provide an efficient fixed-length document representation which is ideal for parallel processors (Stanfill and Thau, 1991). However, Salton and Buckley (Salton and Buckley, 1988) have shown that the limited memory units attached to the small processing units on Connection Machine cannot accommodate sophisticated term weights, resulting in significantly degraded performance. In (Couvreur et al., 1994), Couvreur et al. reported the results of modeling the

performance of searching large text databases (10+ GBs of *Chemical Abstracts*) via various parallel hardware architectures and search algorithms. They found that a multiprocessor mainframe with parallel inverted index file algorithms and the TRW Fast Data Finder (FDF, special-purpose parallel IR hardware) with "on-the-fly" pattern matching capability out-performed loosely-coupled RISC processors with a text signature algorithm.

While parallelizing existing IR algorithms accounted for a majority of previous research efforts, a significant number of diverse and promising information processing and analysis algorithms have emerged and are believed to be ideal for parallel computation. Co-occurrence analysis and clustering algorithms have traditionally been the most computationally intensive algorithms in information science. As discussed previously, this class of algorithms often aims to compute "relationships" between term-pairs and/or document-pairs and cluster terms/documents of similar nature. As Rasmussen commented (Rasmussen, 1991), "An IR application that is particularly computationally intensive (usually $O(N^2)$ to $O(N^3)$, while offering a high degree of parallelism, is document clustering." He cautioned that the successful implementation of a parallel solution in IR requires an appropriate match of task, algorithm, and architecture.

5.5.2 Parallel Computing for Knowledge Discovery

Many new algorithms developed in the area of machine learning, in particular neural networks and genetic algorithms, are parallel in nature and have become prime candidates for parallel computation. Unlike the IR tasks performed by the previous pattern matching, signature files, and inverted index algorithms, most of these machine learning algorithms require extensive pre-processing and analysis of a significant number of textual documents. (Chen provides a complete and up-to-date review and discussion of machine learning techniques for IR in (Chen, 1995b).)

Neural networks computing, in particular, seems to fit well with conventional retrieval models such as the vector space model (Salton, 1988) and the probabilistic model (Maron and Kuhns, 1960). Oddy and Balakrishnan (Oddy and Balakrishnan, 1991) described a parallel IR system in which a document collection is represented as a network mapped over a Connection Machine. The *PThomas* system is similar to the neural networks model conceptually. However, these researchers noted the practical problem of CM size limits (32K processors), which may render the approach to be infeasible for large-scale databases. They suggested a network partition approach to solving this problem.

Yang and his coworkers (Yang and Korfhage, 1993) have developed adaptive retrieval methods based on genetic algorithms and the vector space model using

relevance feedback. They reported the effect of adopting genetic algorithms in large databases, the impact of genetic operators, and GA's parallel searching capability. Frieder and Siegelmann (Frieder and Siegelmann, 1991) also reported a data placement strategy for parallel information retrieval systems using a genetic algorithms approach. Their results compared favorably with pseudo-optimal document allocations.

This emerging machine learning and information analysis paradigm for IR was also echoed in the recent "Report on Workshop on High Performance Computing and Communications for Grand Challenge Applications: Computer Vision, Speech and Natural Language Processing, and Artificial Intelligence" (Wah, 1993). Automatic analysis of co-occurrence patterns in a text corpus, development of electronic librarians to locate information, and discovering new knowledge from existing sources are among the main areas of future research identified by the workshop participants. As the report stated: "High performance AI systems will undoubtedly require very large knowledge bases. Today, the construction of even small to medium knowledge bases is very time-consuming and often prevents the application of AI to real-world problems. To overcome this deficiency, automation of knowledge-base construction is needed. Knowledge may be acquired from the vast amount of information stored as texts. Patterns of concepts and their semantic properties may then be extracted from text via natural language parsing and learning techniques." The research work in the areas of automatic thesaurus

generation, spreading activation, and machine learning based IR is a good example of adopting this new "knowledge discovery" paradigm for digital libraries.

5.6 Parellel Computing: Concept Space Generation

In addition to discussing supercomputers based on the SIMD and MIMD architectures, some researchers also classify supercomputers in terms of their processor and memory requirements. According to Larry Smarr, director of NCSA, the first era of supercomputing belongs mainly to "shared memory vector processors" such as Cray X-MP, Cray Y-MP, Cray 2, and Convex C3. The second era of "distributed memory systems" include systems like CM-2, CM-5, IBM clusters, HP clusters, etc. More recently, "shared memory multiprocessors" (SMP) have emerged as the dominant force for the third and current era, e.g., SGI Power Challenge, Convex Exemplar, etc. The parallel computation implementation of the Illinois DLI semantic retrieval research has coincided with the availability of supercomputers of the second and third eras.

As one of the Artificial Intelligent (AI) Lab members at the University of Arizona I have had an opportunity to use all NCSA's supercomputing facility. The rest of this section describes a series of parallel computing implementations and experiments on concept space generation that we have performed from 1994 to 1999. The parallel implementation since Fall 1994 has been on a 512-node CM-5, a 16-processor SGI Power Challenge, a 48-processor SGI Power Challenge Array

(SGIs), a 16-processor Cray CS6400 (SPARCs), a 64-processor Convex Exemplar (HPs), and a 1024-processor SGI Origin2000.

5.6.1 Worm and Fly Concept Spaces Generation Using CM-5 and SGI Power Challenge

In our previous NSF-funded "Worm Community System" project, we adopted the concept space approach to cross-domain scientific information retrieval. By working closely with worm and fly biologists in the Molecular and Cellular Biology Department at the University of Arizona for about two years, we generated a worm thesaurus in Fall 1993 (Chen et al., 1995) and a fly thesaurus in Summer 1994. Both thesauri were independently tested by the biologists and are available for Internet WWW access at: http://bpaosf.bpa.arizona.edu:8000/cgi-bin/BioQuest.

The resulting worm thesaurus consisted of 7,657 terms and 547,810 links and the fly thesaurus contained 15,626 terms and 750,314 links (after applying various thresholds). Most of these terms were author names or subject descriptors. The document collections were mainly Medline and Biosis abstracts acquired from online sources. Each collection contained about 7,000 abstracts, with 10 MBs of text. It took 50 and 70 minutes, respectively, to generate the two thesauri on a DEC Alpha 2100 workstation (200 MHz, 128-MB RAM). Automatic indexing, which is less computationally intensive, took about 2.5 minutes; while the rest of

the computation was for co-occurrence analysis. The resulting thesauri were about the same size as the initial document collections (i.e., 1 : 1 storage overhead).

In order to address the scalability issue for concept space generation for large collections, i.e., GB-scale collections, which are common in scientific and engineering domains (the topic domains of the Illinois DLI project), we proceeded to test all concept space algorithms, in particular those of automatic indexing and co-occurrence analysis on supercomputers at NCSA. Our initial platform was the CM-5.

The 7000+ abstracts of the fly collection were used to generate the same fly thesaurus using a 512-node CM-5. CM-5 was mainly based on a massively parallel architecture, with 512 SPARC Cypress processors on distributed memory. Its total memory size is 16 GBs (32 MBs/node on 512 nodes). In automatic indexing, we took the *data parallel* approach by breaking the data file into 100 documents (each 0.5 MB in size) and assigned each 100-document set to one node. Our existing C code was modified to C*, CM's data parallel programming language. In addition, each node needed to access a 2-MB *object filtering* file (with list of known biology terms). Due to the memory size limit of the CM-5 processing unit (node), the automatic indexing process on the same collection took about 19 minutes. However, the same *data parallel* approach worked very well for co-occurrence analysis. By sorting all unique terms in alphabetical order and partitioning them into 22-by-22 (484) chunks of ordered terms, we were able to assign each chunk (0.25 MB) to

one CM-5 node. The small file size and processor scale-up greatly speeded up the processing time from 67 minutes (on the DEC Alpha) to 4 minutes, a 17-fold improvement (according to Thomborson (Thomborson, 1993), a more than 20-fold speedup on supercomputers is unlikely without expensive and time-consuming re-coding). Although the testing was encouraging, CM-5 did not appear to be able to alleviate the automatic indexing bottleneck for large-scale test collections.

Further testing was conducted in Spring 1995 on the new NCSA Power Challenge (installed in Fall 1994), which is based on a shared memory multiprocessor architecture. It contains 16 MIPS R8000 processors, with a total shared memory size of 4 GBs. Using the same *data parallel* approach, we fully utilized the 16 processors. The resulting processing time was 24 seconds for automatic indexing and 21.5 minutes for co-occurrence analysis. The shared memory architecture alleviated the automatic indexing bottleneck experienced in CM-5, but the co-occurrence analysis time was longer than that on CM-5. All programs were written in C. Table 1 summarizes the CPU time for the 10-MB fly collection on automatic indexing and co-occurrence analysis, respectively, on a DEC Alpha 2100, 512-node CM-5, and 16-processor SGI Power Challenge.

In summary, due to the (distributed) memory size limit of the CM-5 processing unit (32 MBs/node), the automatic indexing process, which involved a large object filtering file, became the bottleneck. However, the 512-node CM-5 was able to perform co-occurrence analysis (a similarity matrix computation by nature, an

Algorithm/Platform	DEC Alpha 2100	CM-5	SGI Power Challenge
Automatic Indexing	2.5 mins	19 mins	24 secs
Co-occurrence Analysis	67 mins	4 mins	21.5 mins
Total	69.5 mins	23 mins	22 mins

Table 5.1: A benchmark comparison summary on DEC Alpha, CM-5, and SGI Power Challenge

$O(N^2)$ process, where N is the number of terms) efficiently after we adopted a data parallel approach, in which each node received a small block (in size) of the matrix to compute. Due to the large shared memory space (4 GBs), automatic indexing was no longer the bottleneck for SGI Power Challenge. However, co-occurrence analysis remained time-consuming, especially for the specific 16-node NCSA SGI Power Challenge we tested (due to a smaller number of processors and a slower clock rate).

The programming learning curve on SGI Power Challenge was significantly smoother than that on CM-5 (roughly 2 weeks vs. 2 months). After some careful consideration and discussions with researchers and staff at NCSA, SGI Power Challenge appeared most promising because of its ease of programming, large shared memory, and expandability. With the planned addition of the 48-processor SGI Power Challenge Array (and faster processors) and supercomputers of similar architecture (e.g., 64-processor Convex Exemplar, and 16-processor Cray CS6400), at NCSA, SMP-based parallel computers were our choice for further experiments.

Our most recent experiment involved a 24-node Convex Exemplar, also provided by NCSA. The NCSA Convex Exemplar employed in September 1995 was a 3-hypernode SPP-1200 system, with 24 HP PA-RISC 7200 chips (processors), 4 GBs of physical memory, and 88 of GBs disk space with peak performance 240 MFLOPS per processor and 1.9 GFLOPS per hypernode. Given more processors and a higher clock rate per processor (compared with the 16-node SGI Power Challenge), automatic indexing for the same fly collection took 0.39 minutes and co-occurrence analysis took 1.46 minutes, both better performances than those of CM-5 and the 16-node SGI Power Challenge.

For large-scale analysis of textual collections, we believe the shared memory multiprocessors (SMP) such as SGI Power Challenge (the NCSA SGI Power Challenge was also recently upgraded) and Convex Exemplar are extremely promising. In a recent issue of *Science* (Pool, 1995), two U.S. supercomputing center directors have commended SMP highly for its architectural fit with the emerging data mining (knowledge discovery) and digital library applications.

5.6.2 Computer Engineering Concept Space Generation Using SGI Power Challenge

In the Illinois DLI project, we obtained an INSPEC test collection of 400,000+ (computer science and electrical engineering abstracts from the 1992-1994 INSPEC

database in Spring 1995. This 2-GB testbed recently was used to generate a computer engineering concept space using the 16-processor SGI Power Challenge. The automatic indexing process for this gigabyte collection (about a 200-fold increase in size over the fly/worm collection) took 1.5 hours. The most computationally intensive co-occurrence analysis took 23 hours. It used about 25% of the available CPU cycles on the NCSA Power Challenge for a three-week period (in fact it was the largest single user of the NCSA supercomputing resources at that time). The computer engineering concept space contained about 270,000 terms and 4,000,000+ links. We estimated that running the same program on our DEC Alpha 2100 workstation for the same INSPEC collection would take about 20-30 days of CPU time. This long turn-around time was considered infeasible, due to the iterative nature of our system development and testing effort and the need for testing other even larger collections, e.g., 5 million Compendex engineering abstracts.

We also obtained an online version of the INSPEC thesaurus, which contains 7,000+ terms in a classification hierarchy (mostly narrower-term, broader-term, related-term relationships). Our initial analysis showed that the computer engineering concept space appears to contain finer-grained and newer concepts (terms) than the INSPEC thesaurus. On the other hand, the INSPEC thesaurus provides a richer classification structure of the conceptual relationships than the concept space because of its symbolic links.

The computer engineering concept space was indexed (using WAIS indexing) and recently placed on WWW as an Internet server at: http://ai.bpa.arizona.edu/cgi-bin/csquest. Figure 5.1 shows the search results (related terms) using this server, called *CSQuest*, with a search term, "context analysis." We recommend that this server be used as an interactive computer term suggester for searching any computer-related bibliographical databases (e.g., INSPEC database) or Internet services (e.g., the CS Technical Report projects).

5.6.3 CancerLit Concept Space Generation Using SGI Origin2000

In a medical informatics project, we obtained a CancerLit collection from National Cancer Institute (NCI) that includes close to a million CancerLit document records from January 1992 to June 1999. This 4-GB testbed also represents close to two-thirds of recent cancer research in the entire CancerLit collection. We have created several CancerLit concept spaces exclusively on SGI Origin2000, a new shared memory supercomputing machine in NCSA, since 1997.

Because of the availability of faster processors than those of SGI Power Challenge, the automatic indexing process took less than an hour of processing time on the SGI Origin2000. We tested various processors using the data parallelization strategy. The operating results showed that the wall-clock time (real time elapsed) decreased linearly proportional to the number of processors being used. That is, if 16 processors were used, the wall-clock time to finish the automatic indexing

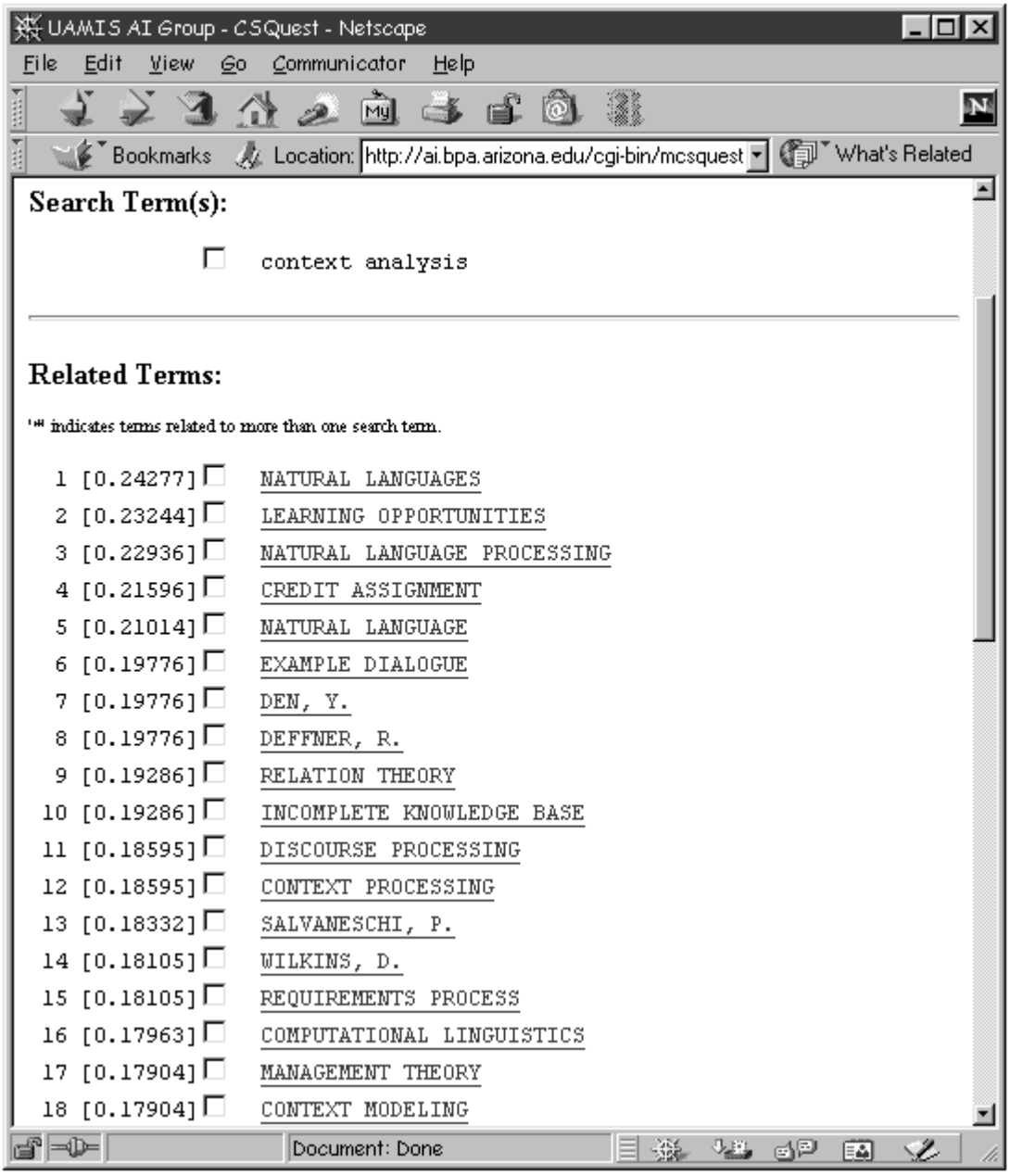

Figure 5.1: CSQuest-suggested terms for "context analysis"

process was about 4 minutes. Memory usage for this automatic indexing process was insignificant. Each processor used about 2 MB of memory to handle input, output and working memory needed. The required lookup memories for stop words and stemming were initialized at less than 0.1 MB of shared memory in less than 1 second of processing time.

Intrinsically, while the data parallelization strategy works well with the automatic indexing process, which is merely an independent analysis on each individual record, parallelization strategy at an algorithmic level is needed for co-occurrence analysis on the entire CancerLit collection. However, we found that a coarse-grained algorithmic parallelization strategy was adequate for co-occurrence analysis when the tremendous memory of up to 64 GB RAM was available and shared simultaneously by multiple processors to perform parallel and independent tasks. We also found that using more processors for co-occurrence analysis might not be beneficial to the overall run-time performance because the use of more processors (up to 32) led to longer total waiting time for peer processes to reach corresponding synchronized points. The coarse-grained strategy allowed longer individual processing cycles between synchronized points to maximize the advantage of having large physical memory. The co-occurrence analysis took 3.5 hours of computing cycles in the SGI Origin2000 running with 8 processors and 4 GB RAM. The improvement in performance came from both better hardware and software optimization.

A revision of CancerLit concept space as of August 1999 contained 1.3 million terms and 27 million links, which covered the underlying CancerLit records from January 1992 to June 1999. A simple CGI-enabled search engine was built to access both CancerLit concept space and document space interchangeably. The server size for the corresponding revision was 3.5 GB. The CancerLit concept space server is accessible at http://ai20.bpa.arizona.edu/cgi-bin/cancerlit/cn.

5.6.4 Lessons Learned: Matching Concept Space Techniques with Parallel Computer Architecture

Both evolution of supercomputer architectures and the maturation of the concept space techniques for very large applications since 1993 have provided tremendous dynamics impetus to understand how to optimize software design and implementation by leveraging on different characteristics of various hardwares and architectures. The strengths and weaknesses of different supercomputer architectures play a significant role in directing the software optimizing effort. Simultaneously, optimization efforts need to fit the nature of concept space techniques at various stages.

Both data and algorithmic parallelization strategies fit well with concept space techniques and are fully supported by the lastest shared memory supercomputer architecture represented in SGI Origin2000. In addition to dicussing these two

strategies, the following examines the scalability issues related to the overall concept space generation process.

5.6.4.1 Data Parallelization

Acknowledging that concept space techniques are applicable to information analysis at the single-document as well as the entire-collection level, it nevertheless is natural to separate these two levels of analysis cleanly. The automatic indexing process is single-document analysis while the co-occurrence process is an aggregated analysis at collection level. The data parallelization strategy fits well with the automatic indexing process because the analysis involves taking an input stream, using a small amount of working memory (usually less than 0.1 MB), and producing an output stream. If input data are equally distributed to N processors, a linear reduction in wall-clock run-time will be achieved. We can use as many processors as are available in a computing environment to speed up the automatic indexing process using data parallelization strategy. Indeed, such a strategy will apply equally well on a network of personal computers.

5.6.4.2 Algorithmic Parallelization

The co-occurrence analysis requires simultaneous availability of all relevant data to compute each co-occurring weight between two terms. This requirement demands enormous memory for operation. The shared memory architecture in SGI

Origin2000 fits this requirement. In fact, the size of all necessary data for co-occurrence analysis after pre-processing is under 0.5 GB for a collection of 1 million documents.

The design of the algorithmic parallelization strategy for co-occurrence analysis depends on how much working memory is available and the characteristics of the co-occurrence matrix. Generally, the available memory, other than that for input data, is divided equally among a set of processors for parallel execution. The co-occurrence matrix is very sparse; that is, a term rarely co-occurs with more than 10% of all terms in an analysis. Many terms co-occur only hundreds or thousands of times in a collection. Based on this observation, we used an optimal method to accumulate all co-occurring pairs of terms when processing each document. However, such a co-occurrence process may not have sufficient working memory to hold all pairs of terms. When going through all documents, we therefore mixed a data parallelling strategy to create a portion of term pairs having a different set of starting terms. That is, all documents were parsed multiple times. Using of up to 2 to 4 GB of memory, the number of times to parse all documents was less than 100. Therefore, the performance efficiency of the co-occurrence analysis was directly proportional to the size of memory available to the process. From a series of experiments on executing the co-occurrence analysis with various number of processors ranging from 1 to 32 (not continuously), the increase in number of processors used demonstrated diminishing overall performance efficiency for a

given size of memory. This is due to the fact that having more processors reduces the effective size of each one's working memory. The optimal combination of memory and number of processors were found to be 2-4 GB and 8-16 processors, respectively.

5.6.4.3 Synchronization: How Often?

The frequency of synchronization was directly related to whether an algorithmic parallelation strategy applies to a particular process. We chose to use a coarse-grained approach to maximize the ratio of run time to idle time for each processor in the co-occurrence analysis. This also was one of the factors in reaching the point of diminishing return on the performance efficiency when adding more processors. As long as one processor was still working, all other peer processors were forced to wait. The use of more processors simply cumulates more idle time in each synchronization.

5.7 System Evaluation: A Concept Association Experiment

In order to examine the performance of the computer engineering concept space in capturing meaningful conceptual associations between terms, a concept association experiment was conducted using the INSPEC thesaurus as the benchmark for comparison. The experimental design was similar to that of those adopted in memory association (Anderson, 1985a), (Chen and Lynch, 1992) and information

retrieval experiments (Chen et al., 1995). The experimental design and results generated are presented in this section.

5.7.1 Experimental Design

A two-phase experiment was performed involving 12 subjects affiliated with the University of Arizona Management Information Systems Department, including two faculty members, and ten graduate degree candidates who had successfully completed course work in Artificial Intelligence, Databases, or Telecommunications/Networking. Prior to Phase 1, experimenters solicited from each faculty subject a list of 16 candidate terms from his domain that could be used as test descriptors. For each domain, eight terms found in both the engineering concept space and the INSPEC thesaurus were selected. One term was discarded because subjects objected to it, leaving a total of 23 test descriptors.

In Phase 1 (*Recall Phase*), each subject (both student and faculty) was asked to generate through a free association process as many related terms as possible in response to each test descriptor presented. This phase of the experiment called upon subjects' memory recall.

In Phase 2 (*Recognition Phase*), experimenters created randomized lists of associated terms for subjects to evaluate with regard to their relevance to the test descriptor, including 40 associated terms suggested by the Computer Engineering Concept Space, and all terms suggested by the INSPEC thesaurus. The concept

space offered significantly more than 40 terms; the highest weighted 40 terms were selected for evaluation (about 2 screenfuls of terms). The 12 subjects were then asked to evaluate each suggested term according to the Likert-like scale: "Irrelevant," "Somewhat Relevant," "Very Relevant." Terms considered too general were to be ranked as "Irrelevant." This phase of the experiment called upon the subjects' ability to recognize relevant terms. Human beings are more likely to recognize than to recall. The complete experiment lasted between 1.25 hours and 2.5 hours for each subject.

5.7.2 Experimental Results: Concept Recall and Concept Precision

In contrast to the *document* recall and precision measures typically used in information science research, this experiment adopted *concept recall* and *concept precision* for evaluation. Instead of examining the number of relevant documents retrieved, the number of relevant terms (concepts) identified by the concept space and the INSPEC thesaurus was counted. They were computed as follows:

$$ConceptRecall = \frac{\text{Number of Retrieved Relevant Concepts}}{\text{Number of Total Relevant Concepts}}$$

$$ConceptPrecision = \frac{\text{Number of Retrieved Relevant Concepts}}{\text{Number of Total Retrieved Concepts}}$$

The number of *Retrieved Relevant Concepts* represented the number of concepts judged "Very Relevant" or "Somewhat Relevant" for each thesaurus. *Total Relevant Concepts* represented the target set of concepts that can be obtained through user-thesaurus interaction, and included all concepts generated by the subjects in Phase 1, as well as those additional unique concepts judged relevant from the computer engineering concept space and the INSPEC thesaurus from Phase 2. Graduate student subjects generated between 0 and 49 terms, with a mean of 7.83 terms. Faculty subjects generated between 5 and 30 terms, with a mean of 16.47 terms. Based on this target set of concepts, we examined the relevant concepts generated by each thesaurus to determine the *concept recall*. *Total Retrieved Concepts*, representing the total number of terms suggested by either thesaurus, was used to calculate *concept precision* levels. For the concept space, this value was always 40. The number of retrieved terms offered by the INSPEC thesaurus ranged from 2 to 38, with a mean of 10.391 terms. Two-sample t-tests were performed for *concept recall* and *concept precision*. Separate comparisons were made for each group of subjects (graduate students and experts).

The ten graduate student subjects, responding to each of the 23 test descriptors, generated a total of 230 data sets. The results for *concept recall* and *concept precision* are shown in Figure 5.2. The concept recall results represented a sample size of 218 and included 12 data sets in which subjects did not respond to the test descriptor presented and thus were assigned a *Retrieved Relevant Concepts*

value of zero. Concept recall for the automatically generated concept space was 69%, significantly greater than the 17.7% recall value resulting from the INSPEC thesaurus. This result can be attributed to the fact that the concept space offered the subjects a greater number of potentially useful and relevant terms. Of the total set of relevant terms for each test descriptor, approximately 70% came from the automatic concept space. This points to a major advantage of the automatically generated concept space – that it can offer the searcher a richer and more meaningful space for concept association and term suggestion.

Concept precision for the concept space was less than that of the INSPEC thesaurus (59.5% vs. 68.2%), a difference that was statistically significant (at a 10% significance level). That the precision for INSPEC thesaurus was not 100% can be explained by the fact that although terms in a manually generated thesaurus are carefully selected to represent a limited number of highly relevant terms, subjects typically considered broader or parent terms to be irrelevant, which lessened the number of potentially relevant terms within the set suggested. It was not unexpected, then, that the INSPEC thesaurus fared better on precision than did the concept space. Automatic indexing, the technique used in automatic thesaurus generation, not only generates useful, but also noisy terms. Thresholds can be applied to limit this effect, but cannot eliminate it. Therefore the concept space would be expected to contain more potentially irrelevant terms.

Similar results were obtained from the faculty subjects. These subjects responded to those terms relevant to their area of expertise; one responded to Artificial Intelligence terms, the other to Database and Telecommunication/Networking terms. All data sets were completed by the faculty subjects, resulting in a sample size of 23. The experts' concept recall for the automatic thesaurus was somewhat lower than that of student subjects, indicating that the faculty members' criteria for relevance was more stringent than that of the students. In addition, experts tended to have a much higher rate of matching thesaurus and concept space suggested terms. So, while they identified fewer terms as being relevant than did the student subjects, the presence of numerous matching terms from Phase 1 resulted in lower recall. Experts' concept precision was higher than that for the students for both the concept space and the INSPEC thesaurus, primarily because experts failed to respond to far fewer concepts than students. That the difference in precision performance was not statistically significant for the experts is probably attributable to the relatively small sample size.

In conclusion, the automatically generated computer engineering concept space performed much better than the INSPEC thesaurus with regard to concept recall, whereas the INSPEC thesaurus performed better than the concept space with regard to concept precision. The implications of these findings are that the concept space appears to be robust and useful, that the automatically-generated concept space and the manually-created INSPEC thesaurus complement one another, and

```
                                    INDIVIDUAL 95 PCT CI'S FOR MEAN
 LEVEL           N      MEAN     STDEV  -+---------+---------+---------+-----
Rec Auto Stud  218    0.6908    0.1852                                    (*-)
Rec Insp Stud  218    0.1771    0.1381   (*)
                                        -+---------+---------+---------+-----
   POOLED STDEV =     0.1633           0.16      0.32      0.48      0.64

                                    INDIVIDUAL 95 PCT CI'S FOR MEAN
  LEVEL          N      MEAN     STDEV  -+---------+---------+---------+-----
Pre Auto Stud  230    0.5950    0.2822   (--------*--------)
Pre Insp Stud  230    0.6822    0.4153                     (--------*---------)
                                        -+---------+---------+---------+-----
   POOLED STDEV =     0.3551          0.550     0.600     0.650     0.700
```

Figure 5.2: ANOVA analysis for recall and precision with graduate student subjects

```
                                    INDIVIDUAL 95 PCT CI'S FOR MEAN
 LEVEL           N      MEAN     STDEV  ----+---------+---------+---------+--
Rec Auto Fac   23    0.5923    0.1294                                  (--*---)
Rec Insp Fac   23    0.1502    0.1064   (--*--)
                                        ----+---------+---------+---------+--
  POOLED STDEV =     0.1185             0.15      0.30      0.45      0.60

                                    INDIVIDUAL 95 PCT CI'S FOR MEAN
 LEVEL           N      MEAN     STDEV  --------+---------+---------+--------
Pre Auto Fac   23    0.6772    0.2004   (-------------*-------------)
Pre Insp Fac   23    0.7366    0.2665            (-------------*-------------)
                                        --------+---------+---------+--------
POOLED STDEV =       0.2358                 0.630     0.700     0.770
```

Figure 5.3: ANOVA analysis for recall and precision with faculty subjects

that the greatest assistance to the searcher can be provided when both are available for query enhancement.

Although the concept association experimental result was encouraging, the usefulness of the concept space for improving user query results (i.e., document recall and precision) was not tested. In a recent AI Lab experiment involving worm (nematode) and fly (Drosophila) concept spaces and a worm database, we have shown that concept spaces helped improve the document recall level (from 32% to 65%), although the document precision level did not improve significantly (Chen et al., 1997). As the next step of our project, we plan to incorporate the engineering concept space in the Illinois engineering literature testbed and examine its usefulness in improving document recall and precision. Future experiments will also include several spreading activation techniques for consulting concept spaces algorithmically for document retrieval, as shown in previous chapter.

5.8 Discussion and Conclusion

This research presents preliminary results generated from the semantic retrieval research component of the NSF/ARPA/NASA-funded Illinois DLI project. Using a variation of the automatic thesaurus generation technique - the *concept space* approach, the goal was to create graphs of domain-specific concepts (terms) and their weighted co-occurrence relationships for all major engineering domains. Merging these concept spaces and providing traversal paths across different concept spaces

could potentially help alleviate the *vocabulary (difference) problem* evident in large-scale information retrieval. The AI Lab previously has successfully adopted such a technique for a smaller molecular biology domain (Worm Community System, with 10+ MBs of document collection) with encouraging results.

In order to address the scalability issue related to large-scale information retrieval and analysis for the current Illinois DLI project, we recently proceeded to experiment using the concept space approach on parallel supercomputers. Our test collection was 2+ GBs of computer science and electrical engineering abstracts and the concept space approach called for extensive textual and statistical analysis (a form of knowledge discovery) based on *automatic indexing* and *co-occurrence analysis* algorithms, both previously tested in the biology domain. Initial testing results using a 512-node CM-5 and a 16-processor SGI Power Challenge were promising. Power Challenge was later selected to automatically create a comprehensive computer engineering concept space of about 270,000 terms and 4,000,000+ links using 24.5 hours of CPU time. The user evaluation involving 12 knowledgable subjects revealed that the automatically-created computer engineering concept space generated significantly higher *concept recall* than the human-generated INSPEC thesaurus (concept space : INSPEC thesaurus = 69.08% : 17.71%). However, the INSPEC was more precise than the automatic concept space (concept space : INSPEC thesaurus = 59.50% : 68.22%). Using the INSPEC thesaurus as the benchmark for comparison, I believe the computer engineering concept space has

demonstrated its robustness and potential usefulness for suggesting relevant terms for search. However, multiple interfaces and multiple vocabulary search aids will be necessary for effective concept-based search across multiple large-scale repositories and domains.