Lecture Notes on Verifications

15-317: Constructive Logic
Frank Pfenning

Lecture 5
Tuesday, January 31, 2023

1 Introduction

The verificationist point of view, already introduced earlier in the course, is that the meaning of a logical connective should be determined by its introduction rules. From this meaning we derive and then check the soundness and completeness of the elimination rules. These "local" checks pertain only to a single connective at a time.

Under this point of view, what is the meaning of a *proposition*, of course constructed from multiple logical connectives? We say the meaning of a proposition is determined by its *verifications* [Martin-Löf, 1983]. In order to be consistent with the explanation of the connectives, a verification should therefore proceed by introduction rules. However, we also need to take the elimination rules into account because they inevitably appear in the proof of a proposition.

Intuitively, a verification should be a proof that only analyzes the constituents of a proposition. This restriction of the space of all possible proofs is necessary so that the definition is well-founded. For example, if we allowed *all* proofs, then in order to understand the meaning of A, we would have to understand the meaning of $B \supset A$ and B, the whole verificationist approach is in jeopardy because B could be a proposition containing, say, A. But the meaning of A would then in turn depend on the meaning of A, creating a vicious cycle.

In this section we will make the structure of verifications more explicit. We write $A\uparrow$ for the judgment "*A has a verification*". Naturally, this should mean that A is true, and that the evidence for that has a special form. Eventually we will also establish the converse: if A is true then A has a verification. Verifications also play a helpful role in proof search, because $A\uparrow$ limits how a proof of A can be constructed.

From the proof search perspective, the notion of verification is called *intercalation* [Sieg and Byrnes, 1998]. The name suggests that we work toward the middle by only using introduction rules from below and elimination rules from above while construction a derivation.

Besides understanding the meaning of propositions and encoding a search strategy, there is a third important reason to study verifications: from the programming language perspective they give rise to the important algorithm of *bidirectional typechecking* [Dunfield and Krishnaswami, 2022].

It turns out that all of these three: verifications, intercalations, and bidirectional typings, and one and the same!

2 Verifications for the Verificationist

Conjunction is easy to understand. A verification of $A \wedge B$ should consist of a verification of A and a verification of B.

$$\frac{A\!\uparrow \quad B\!\uparrow}{A \wedge B\!\uparrow}\ \wedge I$$

We reuse here the names of the introduction rule, because this rule is strictly analogous to the introduction rule for the truth of a conjunction.

Implication, however, introduces a new hypothesis which is not explicitly justified by an introduction rule but just a new label. For example, in the proof

$$\cfrac{\cfrac{\overline{A \wedge B\ \mathit{true}}\ u}{A\ \mathit{true}}\ \wedge E_1}{(A \wedge B) \supset A\ \mathit{true}}\ \supset I^u$$

the conjunction $A \wedge B$ is not justified by an introduction.

The informal discussion of proof search strategies earlier, namely to use introduction rules from the bottom up and elimination rules from the top down contains the answer. We introduce a second judgment, $A\!\downarrow$ which means "*A may be used*". $A\!\downarrow$ should be the case when either A *true* is a hypothesis, or A is deduced from a hypothesis via elimination rules. Our local soundness arguments provide some evidence that we cannot deduce anything incorrect in this manner.

We now go through the connectives in turn, defining verifications and uses.

Conjunction. In summary of the discussion above, we obtain:

$$\frac{A\!\uparrow \quad B\!\uparrow}{A \wedge B\!\uparrow}\ \wedge I \qquad \frac{A \wedge B\!\downarrow}{A\!\downarrow}\ \wedge E_1 \quad \frac{A \wedge B\!\downarrow}{B\!\downarrow}\ \wedge E_2$$

The first/left elimination rule can be read as: "*If we can use $A \wedge B$ we can use A*", and similarly for the right elimination rule. The directions of the arrows of verifications and uses matches nicely with the direction in which we end up applying the proof rules. The $\wedge I$ rule with all its verifications is applied toward the top: A verification $A \wedge B\!\uparrow$ of $A \wedge B$ will continue to seek a verification $A\!\uparrow$ of A as well as a verification $B\!\uparrow$ of B. In contrast, the elimination rule $\wedge E_1$ with all its uses is applied toward the bottom: If we have license $A \wedge B\!\downarrow$ to use $A \wedge B$, then we also have license $A\!\downarrow$ to use A.

Implication. The introduction rule creates a new hypothesis, which we may use in a proof. The assumption is therefore of the judgment $A{\downarrow}$

$$\dfrac{\begin{array}{c}\overline{A{\downarrow}}\ u\\ \vdots\\ B{\uparrow}\end{array}}{A \supset B{\uparrow}}\ {\supset}I^u$$

In order to use an implication $A \supset B$ we first require a verification of A. Just requiring that A may be used would be too weak, as can be seen when trying to prove $((A \supset A) \supset B) \supset B{\uparrow}$ (see Section 3) It should also be clear from the fact that we are not eliminating a connective from A.

$$\dfrac{A \supset B{\downarrow} \quad A{\uparrow}}{B{\downarrow}}\ {\supset}E$$

Verifications and uses meet in ${\supset}I^u$ and ${\supset}E$ due to the direction of the implication. A verification $A \supset B{\uparrow}$ of $A \supset B$ consists of a verification $B{\uparrow}$ of B that has license $A{\downarrow}$ to use the additional hypothesis A. A use $A \supset B{\downarrow}$ of $A \supset B$ gives license to use $B{\downarrow}$ but only after launching a verification $A{\uparrow}$ to verify that A actually holds.

Disjunction. The verifications of a disjunction immediately follow from their introduction rules.

$$\dfrac{A{\uparrow}}{A \vee B{\uparrow}}\ {\vee}I_L \qquad \dfrac{B{\uparrow}}{A \vee B{\uparrow}}\ {\vee}I_R$$

A disjunction is used in a proof by cases, that is, $\vee E$. This rule introduces two new hypotheses, and each of them may be used in the corresponding subproof. Whenever we set up a hypothetical judgment we are trying to find a verification of the conclusion, possibly with uses of hypotheses. So the conclusion of $\vee E$ should be a verification.

$$\dfrac{A \vee B{\downarrow} \quad \begin{array}{c}\overline{A{\downarrow}}\ u\\ \vdots\\ C{\uparrow}\end{array} \quad \begin{array}{c}\overline{B{\downarrow}}\ w\\ \vdots\\ C{\uparrow}\end{array}}{C{\uparrow}}\ {\vee}E^{u,w}$$

Truth. The only verification of truth is the trival one.

$$\dfrac{}{\top{\uparrow}}\ {\top}I$$

A hypothesis $\top{\downarrow}$ cannot be used because there is no elimination rule for $\top$.

Falsehood. There is no verification of falsehood because we have no introduction rule. We can use falsehood, signifying a contradiction from our current hypotheses, to verify any conclusion. This is the nullary case of a disjunction.

$$\frac{\bot{\downarrow}}{C{\uparrow}}\ \bot E$$

One might argue that a license to use $\bot$ should give us a license to use any arbitrary other C. But the $\bot E$ rule restricts this such that $\bot{\downarrow}$ is only used to show the C we are actually looking to verify, as in conclusion $C{\uparrow}$ of $\vee E$.

Propositional Variables. How do we construct a verification of a schematic variable like A? We cannot break down the structure of A because there is none, so we can only proceed if we already know A is true. This can only come from a hypothesis, so we have a rule that lets us use the knowledge of an of a propositional variable to construct a verification.

$$\frac{A{\downarrow}}{A{\uparrow}}\ {\downarrow}{\uparrow}^*$$

We annotate the rule here to remind ourselves that A must be a propositional variable, rather than an arbitrary proposition. This rule has a special status in that it represents a change in judgments but is not tied to a particular connective. We call this a *judgmental rule* in order to distinguish it from the usual introduction and elimination rules that characterize the connectives.

This is also the first time we restrict a rule to apply only to schematic variables. In lecture, we did not make this point explicitly—in fact this rule would not allow you to derive anything that is not true even if A may have some propositional structure, but one may question if in that case it would still be a verification of A since we have not broken down its structure by introductions to the greatest possible extent.

Global soundness. Local soundness is an intrinsic property of each connective, asserting that the elimination rules for it are not too strong given the introduction rules. Global soundness is its counterpart for the whole system of inference rules. It says that if an arbitrary proposition A has a verification then we may use A without gaining any information. That is, for arbitrary propositions A and C:

$$\textit{If}\quad A{\uparrow}\quad \textit{and}\quad \begin{array}{c} A{\downarrow} \\ \vdots \\ C{\uparrow} \end{array}\quad \textit{then}\quad C{\uparrow}.$$

We would want to prove this using a substitution principle, except that the judgment $A{\uparrow}$ and $A{\downarrow}$ do not match. In the end, the arguments for local soundness will help us carry out this proof later in this course when we have progressed to sequent calculus.

Global completeness. Local completeness is also an intrinsic property of each connective. It asserts that the elimination rules are not too weak, given the introduction rule. Global completeness is its counterpart for the whole system of inference rules. It says that if we may use A then we can construct from this a verification of A. That is, for arbitrary propositions A:

$$\begin{array}{c} A{\downarrow} \\ \vdots \\ A{\uparrow} \end{array}$$

Global completeness follows from local completeness rather directly by induction on the structure of A. Note how crucial it is to distinguish the verification judgment $A{\uparrow}$ from the use judgment $A{\downarrow}$ to be able to clearly state the goal of global completeness.

Global soundness and completeness are properties of whole deductive systems. Their proof must be carried out in a mathematical *metalanguage* which makes them a bit different than the formal proofs that we have done so far within natural deduction. Of course, we would like them to be correct as well, which means they should follow the same principles of valid inference that we have laid out so far.

There are two further properties we would like, relating truth, verifications, and uses. The first is that if A has a verification or A may be used, then A is true. This is rather evident since we have just specialized the introduction and elimination rules, except for the judgmental rule ${\downarrow}{\uparrow}$. But under the interpretation of verification and use as truth, this inference becomes redundant.

Significantly more difficult is the property that if A is true then A has a verification. Since we justified the meaning of the connectives from their verifications, a failure of this property would be devastating to the verificationist program. Fortunately it holds and can be proved by exhibiting a process of *proof normalization* that takes an arbitrary proof of A true and constructs a verification of A.

All these properties in concert show that our rules are well constructed, locally as well as globally. Experience with many other logical systems indicates that this is not an isolated phenomenon: we can employ the verificationist point of view to give coherent sets of rules not just for constructive logic, but for classical logic, temporal logic, spatial logic, modal logic, and many other logics that are of interest in computer science. Taken together, these constitute strong evidence that separating judgments from propositions and taking a verificationist point of view in the definition of the logical connectives is indeed a proper and useful foundation for logic.

Finally observe how verifications play a role in informing proof search by reducing the proof search space. The direction of the arrows indicates in which direction a judgment should be expanded during proof search. A verification $A{\uparrow}$ needs to be verified upwards by applying its appropriate introduction rule. A license to use $A{\downarrow}$ can be used downwards by applying its appropriate elimination rule. Verifications and uses meet in the judgmental rule ${\downarrow}{\uparrow}$. In fact, when you carefully examine the example deductions we have conducted so far, you will see that they all already ended up following the proof search order that verifications and uses mandate (except the one we constructed to illustrate a local reduction in the proof of $A \supset (B \supset B)$. What needed our creativity in proof search so far has no become systematic thanks to a distinction of whether A needs to be verified or whether A can be assumed to hold.

3 A Counterexample

In this section we illustrate how things may go wrong if we do not define the notation of verification correctly.

If the $\supset\!E$ elimination rule would be modified to have second premise use $A{\downarrow}$ instead of verification $A{\uparrow}$:

$$\dfrac{A \supset B{\downarrow} \quad A{\downarrow}}{B{\downarrow}} \supset\!E?$$

Then the verification of $((A \supset A) \supset B) \supset B{\uparrow}$ would be stuck:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{(A \supset A) \supset B{\downarrow}}\,u \quad A \supset A{\downarrow}}{B{\downarrow}} \supset\!E?}{B{\uparrow}} \uparrow\downarrow}{((A \supset A) \supset B) \supset B{\uparrow}} \supset\!I^u$$

because there is no way to extract $A \supset A{\downarrow}$ from the only hypothesis we have. In contrast, here is the correct verification:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{(A \supset A) \supset B{\downarrow}}\,u \quad \dfrac{\dfrac{\dfrac{}{A{\downarrow}}\,w}{A{\uparrow}} \uparrow\downarrow}{A \supset A{\uparrow}} \supset\!I^w}{B{\downarrow}} \supset\!E}{B{\uparrow}} \uparrow\downarrow}{((A \supset A) \supset B) \supset B{\uparrow}} \supset\!I^u$$

4 Proof Terms Revisited

Any verification is a proof. This very easy to see, because we can traverse a verifcation and replace both $A{\uparrow}$ and $A{\downarrow}$ by A *true* and obtain a proof. The minimal required change is to collapse instances of the rule

$$\dfrac{A{\downarrow}}{A{\uparrow}} \downarrow\uparrow$$

into simply A *true*, because otherwise premise and conclusion of the rule would be identical.

These observations suggest that we should not need to devise a new notation for *proof terms*, just reuse them and distinguish those that constitute verifications. Indeed, we can just annotate the rules with the usual proof terms and we obtain a system which we might right as $M : A{\uparrow}$ (*M is a verification of A*) and $M : A{\downarrow}$ (*M is a justification for the use of A*).

In programming languages, a slightly different notation has taken hold, so we introduce it here. We write

$M \Leftarrow A$	M checks against A
$M \Rightarrow A$	M synthesizes A

The intuition (yet to be tested) you should have is that the judgment $M \Leftarrow A$ embodies an algorithm for *checking* if the term M has type A. The judgment $M \Rightarrow A$ by contrast embodies an algorithm that, when given M can synthesize an A (if the term is indeed well-typed, of course).

Furthermore, since we keep the proof terms, the propositions, and even the inference rules intact and just restrict their use, the fact that both $M \Leftarrow A$ and $M \Rightarrow A$ imply $M : A$ should be immediately evident.

Let's rewrite the rules for verifications with the proof terms and examine this intuition.

Conjunction. As usual, we start with conjunction.

$$\frac{M \Leftarrow A \quad N \Leftarrow B}{\langle M, N\rangle \Leftarrow A \wedge B}\ \wedge I$$

We imagine we are given the task or *checking* whether $\langle M, N\rangle \Leftarrow A \wedge B$. We reduce this task to checking if M checks against A and N checks against B. Fortunately, we have all the information needed to ask these questions and the construction of a derivation proceeds.

Conversion, consider we are asked to *synthesize* a type for $\mathbf{fst}\, M$. That is:

$$\mathbf{fst}\, M \Rightarrow ?$$

We can accomplish this if we can ask for the type synthesized by M. Since **fst** is trying to take the first component of a pair, M better synthesize a conjunction (= product type) using the $\wedge E_1$ rule.

$$\frac{M \Rightarrow A \wedge B}{\mathbf{fst}\, M \Rightarrow ?}\ \wedge E_1$$

Now we can fill in the "?" by extracting the first conjunct in the premise.

$$\frac{M \Rightarrow A \wedge B}{\mathbf{fst}\, M \Rightarrow A}\ \wedge E_1$$

Symmetric reasoning leads us to the rule

$$\frac{M \Rightarrow A \wedge B}{\mathbf{snd}\, M \Rightarrow B}\ \wedge E_2$$

Implication. Implication is already slightly more complicated. The introduction rules creates a new hypothesis, which is something we can *use* so it synthesizes a type.

$$\frac{\begin{array}{c}\overline{x \Rightarrow A}^{\;x} \\ \vdots \\ M \Leftarrow B\end{array}}{\lambda x.\, M \Leftarrow A \supset B}\ \supset I^x$$

Does it make sense to type-check in this way? Yes! We are given $\lambda x.\, M$ to check against $A \supset B$. This means that we know A, so it makes sense that x would *synthesize* the type A. Further, it makes sense to check M against B, since we know both of them.

On to the elimination rule:

$$\frac{M \Rightarrow A \supset B \quad N \Leftarrow A}{M\, N \Rightarrow B} \supset E$$

To start with, we are given only $M\, N$ and have to synthesize B. Since we know M, it is okay to ask the type synthesized by M. At this point in the construction of a derivation, it looks like this:

$$\frac{M \Rightarrow ? \quad N \Leftarrow ?}{M\, N \Rightarrow ?} \supset E$$

The synthesis of a type for M will either fail, or it will give us something other than an implication (= function type) (in which case we fail), or it will give us an implication.

$$\frac{M \Rightarrow A \supset B \quad M \Leftarrow ?}{M\, N \Rightarrow ?} \supset E$$

But at this point we know A, so we can check A and also we know B, the type synthesized for $M\, N$.

$$\frac{M \Rightarrow A \supset B \quad M \Leftarrow A}{M\, N \Rightarrow B} \supset E$$

So, everything works out regarding how we view the rules as an algorithm for proof construction.

Disjunction. We go a bit faster now, just presenting the outcome, and leave it to the reader to check that the rules respect our algorithmic interpretation. First, the checking

$$\frac{M \Leftarrow A}{M \Leftarrow A \vee B} \vee I_1 \qquad \frac{M \Leftarrow B}{M \Leftarrow A \vee B} \vee I_2$$

and then the synthesis rules

$$\frac{M \Rightarrow A \vee B \quad \begin{array}{c} \overline{u \Rightarrow A}\ u \\ \vdots \\ N \Leftarrow C \end{array} \quad \begin{array}{c} \overline{w \Rightarrow B}\ w \\ \vdots \\ P \Leftarrow C \end{array}}{\textbf{case}(M, u.\, N, w.\, P) \Leftarrow C} \vee E^{u,w}$$

Falsehood. There is only one rule, and it arises as the nullary version of $\vee E$.

$$\frac{M \Rightarrow \bot}{\textbf{abort}\, M \Leftarrow C} \bot E$$

Truth. Symmetrically, truth has only a checking rule

$$\frac{}{\langle\,\rangle \Leftarrow \top}\ \top I$$

Judgmental Rule. There is one rule that just manipulates judgments, namely:

$$\frac{A{\downarrow}}{A{\uparrow}}\ {\downarrow}{\uparrow}$$

This rule doesn't actually change the proof term, but there is still something interesting going on. To start with, from the conclusion, we like to the M against A.

$$\frac{M \Rightarrow\ ?}{M \Leftarrow A}\ {\downarrow}{\uparrow}$$

Since we know M, is it okay to ask in the premise for the synthesis of a type of M. We then have to compare whatever is synthesized with A.

$$\frac{M \Rightarrow A' \quad A' = A}{M \Leftarrow A}\ {\downarrow}{\uparrow}$$

Here, $A = A'$ is simply ordinary equality of the two propositions. If we want to extend our language to admit subtyping, this would be the only place we needed to change, checking that A' is a subtype of A.

But as we see in the next section, there is a caveat.

5 Normal Deductions

We call a deduction *normal* if there is no subdeduction that is subject to a local reduction, that is, an introduction followed immediately by an elimination of the connective just introduced.

We can inspect all the possible reductions and we realize that all verifications are normal. Let's look at a particular derivation that can be reduced:

$$\cfrac{\cfrac{\begin{array}{cc}\mathcal{D} & \mathcal{D}\\ A\ \mathit{true} & B\ \mathit{true}\end{array}}{A \wedge B\ \mathit{true}}\ \wedge I}{A\ \mathit{true}}\ \wedge E_1$$

Annotating the rules with directions:

$$\cfrac{\cfrac{\begin{array}{cc}\mathcal{D} & \mathcal{E}\\ A{\uparrow} & B{\uparrow}\end{array}}{A \wedge B\ ?}\ \wedge I}{A{\downarrow}}\ \wedge E_1$$

We see there is a clash: the introduction forces $A \wedge B{\uparrow}$ while the elimination forces $A \wedge B{\downarrow}$. But we do *not* have a rule

$$\frac{A{\uparrow}}{A{\downarrow}}\ {\uparrow}{\downarrow}\ ??$$

because such a rule together with ${\downarrow}{\uparrow}$ would allow us to go freely between $\uparrow$ and $\downarrow$ leading us to the situation where *any* proof is allowed (not just verifications).

We also see that the all-important subformula property fails here, because the proof of $A{\downarrow}$ takes a detour through $A \wedge B$ which is not a subformula of the conclusion or either premise.

It turns out that we do not quite have enough reductions to ensure the converse, which would be "*every proof that cannot be reduced is a verification*". We may return to this point in a future lecture.

6 Counting Verifications

Recall that under the verificationist point of view, the meaning of the proposition is determined by its verifications.

First, we observe that there is no introduction rule for $\bot$ and therefore no verification of $\bot$. In other words, not every proposition has a verification. If we assume global soundness (yet to be proved), then this implies the consistency of the logic.

As a second example, how many verifications are there of $A \supset A$, for a propositional variable A? A minute of doodling will tell you there can be only one, namely:

$$\dfrac{\dfrac{\dfrac{}{A{\downarrow}}\ u}{A{\uparrow}}\ {\downarrow}{\uparrow}}{A \supset A{\uparrow}}\ {\supset}I^u$$

This also means there is exactly one normal term of type $A \supset A$:

$$\lambda\, u.\, u \Leftarrow A \supset A$$

Similarly, there are exactly two verification sof $A \supset (A \supset A)$, as we checked in lecture, and therefore also only two normal proof terms

$$\begin{array}{c} \lambda u.\, \lambda w.\, u \\ \lambda u.\, \lambda w.\, w \end{array}$$

Taking things a step further, we see that the normal proofs of type $(A \supset A) \supset (A \supset A)$ are in bijection with the natural numbers:

$$\begin{array}{lcl} \mathsf{zero} & = & \lambda z.\, \lambda s.\, z \\ \mathsf{one} & = & \lambda z.\, \lambda s.\, s(z) \\ \mathsf{two} & = & \lambda z.\, \lambda s.\, s(s(z)) \\ \ldots & & \end{array}$$

References

Jana Dunfield and Neel Krishnaswami. Bidirectional typing. *ACM Computing Surveys*, 98 (5):1–38, 2022.

Per Martin-Löf. On the meanings of the logical constants and the justifications of the logical laws. Notes for three lectures given in Siena, Italy. Published in *Nordic Journal of Philosophical Logic*, 1(1):11-60, 1996, April 1983. URL `http://www.hf.uio.no/ifikk/forskning/publikasjoner/tidsskrifter/njpl/vol1no1/meaning.pdf`.

Wilfried Sieg and John Byrnes. Normal natural deduction proofs (in classical logic). *Studia Logica*, 60(1):67–106, January 1998.