Lecture Notes on The Inverse Method

15-836: Substructural Logics
Frank Pfenning

Lecture 18
November 15, 2023

1 Introduction

In this lecture we return to an early theme, namely forward inference. We also switch gears from the proofs-as-programs interpretation of substructural logic in terms of message passing and shared memory to general theorem proving.

Why would we want to prove theorems in substructural logics? First, we have already seen that forward inference (which is a particular form of theorem proving) can be used to model various algorithmic problems, like parsing, subtyping, planning, or graph algorithms. Second, there are a logics for reasoning about programs, specifically separation logic [Reynolds, 2002] and concurrent separation logic [O'Hearn, 2007, Brookes, 2007], both of which substructural. Proving correctness of imperative programs in such logics ultimately comes down to theorem proving in substructural logic. Third, the problem of program synthesis that has recently garnered much attention can be simplified if we know, for example, that the programs we want to synthesize have substructural types because it drastically reduces the search space [Hughes and Orchard, 2020, Melo e Sousa, 2021]. Fourth, a structured form of proof search is the basis for substructural logic programming [Hodas and Miller, 1994, López et al., 2005] that allows yet another class of algorithms to be expressed at a high level of abstraction.

In a way there is an "obvious" method to do theorem proving: we use the rules of the cut-free sequent calculus for bottom-up proof construction. Once theorems become even somewhat complex, this is no longer feasible because there are too many choices and therefore too much backtracking. We can use inversion to reduce the number of choices, but there remains much nondeterminism. Chaining together rules with focusing [Andreoli, 1992, 2001] makes this even better, but proof search continues to suffer from the difficulty of learning from failure on some branches while searching others. As far as I am aware, clause learning for SAT has not yet been understood at a sufficiently fundamental level to effectively apply to nonclassical and substructural logics.

An alternative approach is to use Maslov's *inverse method* [Maslov, 1964]. It is called "inverse" because it proceeds from identity sequents towards the goal instead of from the goal towards identity sequents. At first this might seem a crazy idea because the universe of theorems we generate is infinite and poorly structured, but as we will see it works! The inverse method is quite beautiful because it applies essentially to any logic that admits a cut-free sequent calculus, with particular logic-specific considerations in each case [Voronkov, 1992, Degtyarev and Voronkov, 2001]. Other techniques such as resolution are tied specifically to classical logic, so they don't seem as useful for substructural logics. Applications of the inverse method to substructural logic have also been devised [Chaudhuri and Pfenning, 2005a,b, Chaudhuri, 2006] and generalize the material in these notes further.

2 The Basic Idea

Let's look at the example

$$A \multimap (B \mathbin{\&} C) \vdash (A \multimap B) \mathbin{\&} (A \multimap C)$$

which we should be able to prove. In today's lecture, we use A, B, C, etc. to stand for atomic proposition rather than P, Q, R. It seems clear (more later) that the possible identities at the leaves of a proof tree for this sequent should be

$$\frac{}{A \vdash A}\,\mathsf{id}_A \qquad \frac{}{B \vdash B}\,\mathsf{id}_B \qquad \frac{}{C \vdash C}\,\mathsf{id}_C$$

The space of possible forward inferences here seems *huge*! For example, we might deduce

$$\frac{\dfrac{}{A \vdash A}\,\mathsf{id}_A}{\vdash A \multimap A}\,{\multimap}R$$

It is easy to see that this will not get us anywhere, keep in mind our overall goals. Why is that? Before you read on, think about this and see if you can extend a tentative answer to a more general idea how to make forward inference plausible.

The reason this inference is useless is because $A \multimap A$ is not a subformula that occurs in our goal sequent. In the cut-free sequent calculus, though, all propositions that occur in a proof are subformulas of the final goal sequent.

The idea then is to *specialize* the inference rules so they can be applied in the forward direction but to infer subformulas of the goal sequent! Before we do this, let's examine the subformula property more precisely. By looking at the goal sequent, we can not only predict which subformulas might occur in a proof, but also on which side of a sequent they will be. Except for implication, all the rules keep formulas on the same side of the sequent. And the antecedent of an implication is always on the opposite side of the sequent from the implication itself.

Let's apply this idea, naming the subformulas according to the side of the sequent they may appear on, using L_i for left and R_j for right subformulas.

$$\underbrace{A^R \multimap \underbrace{(B^L \,\&\, C^L)}_{= L_1}}_{= L_0} \vdash \underbrace{\underbrace{(A^L \multimap B^R)}_{= R_1} \,\&\, \underbrace{(A^L \multimap C^R)}_{= R_2}}_{= R_0}$$

First, the possible identities that might be used. We see that all three atoms, A, B, and C may appear on the left as well as on the right in a sequent, so all three identities are possible.

$$\frac{}{A^L \vdash A^R}\ \mathsf{id}_A \qquad \frac{}{B^L \vdash B^R}\ \mathsf{id}_B \qquad \frac{}{C^L \vdash C^R}\ \mathsf{id}_C$$

Next, consider $R_0 = R_1 \& R_2$. Since this is a right formula, there is only one possible specialized rule to infer R_0.

$$\frac{\Delta \vdash R_1 \quad \Delta \vdash R_2}{\Delta \vdash R_0}\ \&R_0$$

For $R_1 = A^L \multimap B^R$ and $R_2 = A^L \vdash C^R$ we also get just a single rule each, that is, two instances of $\multimap R$.

$$\frac{\Delta, A^L \vdash B^R}{\Delta \vdash R_1}\ \multimap R_1 \qquad \frac{\Delta, A^L \vdash C^R}{\Delta \vdash R_2}\ \multimap R_2$$

For L_0 there is a single instance of the $\multimap L$ rule.

$$\frac{\Delta_1 \vdash A^R \quad \Delta_2, L_1 \vdash \delta}{\Delta_1, \Delta_2, L_0 \vdash \delta}\ \multimap L_0$$

The last remaining subformula is $L_1 = B^L \,\&\, C^L$ has two specialized left rules.

$$\frac{\Delta, B^L \vdash \delta}{\Delta, L_1 \vdash \delta}\ \&L_1^1 \qquad \frac{\Delta, C^L \vdash \delta}{\Delta, L_1 \vdash \delta}\ \&L_1^2$$

$$\frac{}{A \vdash A}\ \mathsf{id}_A \qquad \frac{}{B \vdash B}\ \mathsf{id}_B \qquad \frac{}{C \vdash C}\ \mathsf{id}_C$$

$$\frac{\Delta \vdash R_1 \quad \Delta \vdash R_2}{\Delta \vdash R_0}\ \& R_0 \qquad \frac{\Delta, A \vdash B}{\Delta \vdash R_1}\ \multimap R_1 \qquad \frac{\Delta, A \vdash C}{\Delta \vdash R_2}\ \multimap R_2$$

$$\frac{\Delta_1 \vdash A \quad \Delta_2, L_1 \vdash \delta}{\Delta_1, \Delta_2, L_0 \vdash \delta}\ \multimap L_0 \qquad \frac{\Delta, B \vdash \delta}{\Delta, L_1 \vdash \delta}\ \& L_1^1 \qquad \frac{\Delta, C \vdash \delta}{\Delta, L_1 \vdash \delta}\ \& L_1^2$$

Figure 1: Specialized rules for $A \multimap (B \mathbin{\&} C) \vdash (A \multimap B) \mathbin{\&} (A \multimap C)$, goal sequent $L_0 \vdash R_0$

The rules are summarized in Figure 1. We have dropped the superscripts on the atoms since they are determined by their position. An interesting observation is that there are no longer any logical connectives! So during inference we do not consider any of the usual sequent calculus rules, just these specialized ones. Because of the (side-aware) subformula property, there is a proof of our goal sequent if and only if we can infer $L_0 \vdash R_0$ with these rules.

We proceed in a breadth first fashion, always applying all possible rules considering the "facts" already in our database, where the facts are sequents that can be derived. Note that even though our logic is linear, this inference is a structural inference so we may hope that it saturates. We start with the first round, in which only two rules can be applied.

(1)	$A \vdash A$	(id_A)
(2)	$B \vdash B$	(id_B)
(3)	$C \vdash C$	(id_C)
(4)	$L_1 \vdash B$	$(\& L_1^1\ 1)$
(5)	$L_1 \vdash C$	$(\& L_1^2\ 2)$

Since $L_1 = B \mathbin{\&} C$, we see that sequents (4) and (5) make sense after we expand the definitions. Besides inferences that only give us sequents we already know, the only new ones are two applications of $\multimap L_0$.

(1)	$A \vdash A$	(id_A)
(2)	$B \vdash B$	(id_B)
(3)	$C \vdash C$	(id_C)
(4)	$L_1 \vdash B$	$(\& L_1^1\ 1)$
(5)	$L_1 \vdash C$	$(\& L_1^2\ 2)$
(6)	$A, L_0 \vdash B$	$(\multimap L_0\ 1\ 4)$
(7)	$A, L_0 \vdash C$	$(\multimap L_0\ 1\ 5)$

Now we can apply $\multimap R_1$ and $\multimap R_2$, followed by $\& R_0$.

(1)	$A \vdash A$	(id_A)
(2)	$B \vdash B$	(id_B)
(3)	$C \vdash C$	(id_C)
(4)	$L_1 \vdash B$	($\& L_1^1$ 1)
(5)	$L_1 \vdash C$	($\& L_1^2$ 2)
(6)	$A, L_0 \vdash B$	($\multimap L_0$ 1 4)
(7)	$A, L_0 \vdash C$	($\multimap L_0$ 1 5)
(8)	$L_0 \vdash R_1$	($\multimap R_1$ 6)
(9)	$L_0 \vdash R_2$	($\multimap R_2$ 7)
(10)	$L_0 \vdash R_0$	($\& R_0$ 8 9)

We have to be careful about the final inference because both premises must have the same antecedent Δ. Fortunately, that is the case with $\Delta = L_0$.

The sequent (10) is also our goal sequent, but we also have reached a point of saturation: any further inferences would only yield sequents we already have.

Next we do an example that is not provable:

$$A \multimap (B \otimes C) \vdash (A \multimap B) \otimes (A \multimap C)$$

As before, we label subformulas.

$$\underbrace{A^R \multimap \underbrace{(B^L \otimes C^L)}_{= L_1}}_{= L_0} \vdash \underbrace{\underbrace{(A^L \multimap B^R)}_{= R_1} \otimes \underbrace{(A^L \multimap C^R)}_{= R_2}}_{= R_0}$$

Instances of the identity as the same as before.

$$\frac{}{A^L \vdash A^R}\ \mathsf{id}_A \qquad \frac{}{B^L \vdash B^R}\ \mathsf{id}_B \qquad \frac{}{C^L \vdash C^R}\ \mathsf{id}_C$$

For the left propositions we generate:

$$\frac{\Delta_1 \vdash A \quad \Delta_2, L_1 \vdash \delta}{\Delta_1, \Delta_2, L_0 \vdash \delta}\ {\multimap}L_0 \qquad \frac{\Delta, A, B \vdash \delta}{\Delta, L_1 \vdash \delta}\ {\otimes}L_1$$

And for the right propositions:

$$\frac{\Delta_1 \vdash R_1 \quad \Delta_2 \vdash R_2}{\Delta_1, \Delta_2 \vdash R_0}\ {\otimes}R_0 \qquad \frac{\Delta, A \vdash B}{\Delta \vdash R_1}\ {\multimap}R_1 \quad \frac{\Delta, A \vdash C}{\Delta \vdash R_2}\ {\multimap}R_2$$

Now we throw away the general rules and start with

(1)	$A \vdash A$	(id_A)
(2)	$B \vdash B$	(id_B)
(3)	$C \vdash C$	(id_C)

At this point we realize that *no rule is applicable*! Therefore we know the goal sequent is not provable.

The same style of rule generations applies to most of the connectives of linear logic ($A \multimap B$, $A \mathbin{\&} B$, $A \otimes B$, $\mathbf{1}$, $A \oplus B$) but has to be modified when we consider the exponential $!A = {\downarrow}{\uparrow}A$ or the additive units $\mathbf{0}$ and $\top$. We'll return to them in Section 5.

3 The Inverse Method with Focusing

We can exploit focusing to create fewer rules and take bigger steps. The only sequents that will explicitly appear in a focused inverse method proof are *stable sequents*, which are those where no invertible rule can be applied and proof must proceed by focusing either on the right or left.

Stable antecedents are either negative propositions A^- or suspended positive atoms $\langle P^+\rangle$. Stable succedents are either positive propositions A^+ or suspended negative atoms $\langle P^-\rangle$. We purposely omit $\mathbf{0}$ and $\top$ for now.

Negative Propositions	A^-	$::=$	$A \multimap B \mid A \mathbin{\&} B$
Stable Antecedents	Δ	$::=$	$\cdot \mid \Delta, \langle P^+\rangle \mid \Delta, A^-$
Positive Propositions	A^+	$::=$	$A \otimes B \mid \mathbf{1} \mid A \oplus B$
Stable Succedents	δ	$::=$	$\langle P^-\rangle \mid A^+$

We just use letters Δ and δ for the stable antecedents and succedents, since only those are of interest in this section.

This time, we do not introduce intermediate names ahead of time, but will do so during the rule generation process. Our goal is

$$A \multimap (B \mathbin{\&} C) \vdash (A \multimap B) \mathbin{\&} (A \multimap C)$$

This is not stable, so we have to apply inversion until we have reached one more more stable sequent. For this purpose we have to decide which atoms should be positive and which negative. For simplicity, we make them all positive. You may want to review the rules for focusing from Lecture 12. We reach two stable sequents:

$$A^+ \multimap (B^+ \mathbin{\&} C^+), \langle A^+\rangle \vdash C^+$$
$$A^+ \multimap (B^+ \mathbin{\&} C^+), \langle B^+\rangle \vdash C^+$$

We have to prove both of these to verify our goal sequent. Let's take the first one (the second one will be symmetric). We can only focus on $A^+ \multimap (B^+ \mathbin{\&} C^+)$ on the left or on C^+ on the right, since we cannot focus on suspended atom. Let's try the first one, defining $L_0 = A^+ \multimap (B^+ \mathbin{\&} C^+)$. We don't know under which circumstances we might focus on this proposition, but if we do the proof will start as follows (omitting other antecedents and succedents for now):

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash [A^+]\end{array} \quad \begin{array}{c}\vdots\\ [B^+ \mathbin{\&} C^+] \vdash\end{array}}{[A^+ \multimap (B^+ \mathbin{\&} C^+)] \vdash}}{L_0 \vdash}$$

There is only one way to proceed with the first open subgoal: right focus on A^+ only succeeds if $\langle A^+ \rangle$ is among the antecedents. That is, for focusing to succeed, such an antecedent must have been in the conclusion.

$$\dfrac{\dfrac{\dfrac{}{\langle A^+\rangle \vdash [A^+]}\,\mathsf{id}^+ \quad \begin{array}{c}\vdots\\ [B^+ \mathbin{\&} C^+] \vdash\end{array}}{\langle A^+\rangle, [A^+ \multimap (B^+ \mathbin{\&} C^+)] \vdash}}{\langle A^+\rangle, L_0 \vdash}$$

In the remaining open subproof we could proceed with focus on B^+ or focus on C^+. As a next step in either of these we lose focus and then have to apply inversion. This inversion will immediately suspect B^+ and C^+, respectively. We show the first version:

$$\dfrac{\dfrac{\dfrac{}{\langle A^+\rangle \vdash [A^+]}\,\mathsf{id}^+ \quad \dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}\vdots\\ \langle B^+\rangle \vdash\end{array}}{\langle B^+\rangle \;;\; \cdot \vdash}}{\cdot \;;\; B^+ \vdash}}{[B^+] \vdash}}{[B^+ \mathbin{\&} C^+] \vdash}\,\mathbin{\&}L_1}{\langle A^+\rangle, [A^+ \multimap (B^+ \mathbin{\&} C^+)] \vdash}}{\langle A^+\rangle, L_0 \vdash}$$

The sequent at the top of this rather bureaucratic chain of reasoning is stable. We can fill in some additional (stable) antecedents and succedents.

$$\dfrac{\dfrac{\dfrac{}{\langle A^+\rangle \vdash [A^+]}\;\mathsf{id}^+ \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\Delta, \langle B^+\rangle \vdash \delta}}{\Delta, \langle B^+\rangle \;;\; \cdot \vdash \delta}}{\Delta \;;\; B^+ \vdash \delta}}{\Delta[B^+] \vdash \delta}}{\Delta, [B^+ \mathbin{\&} C^+] \vdash \delta}\;\&L_1}{\Delta, \langle A^+\rangle, [A^+ \multimap (B^+ \mathbin{\&} C^+)] \vdash \delta}}{\Delta, \langle A^+\rangle, L_0 \vdash \delta}$$

From this, and its symmetric variant with C^+ instad of B^+ we extract two big-step rules between stable sequents.

$$\dfrac{\Delta, \langle B^+\rangle \vdash \delta}{\Delta, \langle A^+\rangle, L_0 \vdash \delta}\;L_0^1 \qquad\qquad \dfrac{\Delta, \langle C^+\rangle \vdash \delta}{\Delta, \langle A^+\rangle, L_0 \vdash \delta}\;L_0^2$$

Interestingly, this does not expose any new subformulas we have to focus on since suspended atoms cannot be the subject of focusing. We still have the succedent C^+ in our original goal sequent.

$$\dfrac{\dfrac{\vdots}{\vdash [C^+]}}{\vdash C^+}$$

This can only succeed if C^+ is a suspended antecedent, so we obtain the rule

$$\dfrac{}{\langle C^+\rangle \vdash C^+}\;\mathsf{id}_C$$

Due to focusing we only obtain 3 rules compared to 8 before. We can only perform one step:

$$\begin{array}{lll} (1) & \langle C^+\rangle \vdash C^+ & (\mathsf{id}_C) \\ \hline (2) & \langle A^+\rangle, L_0 \vdash C^+ & (L_0^2\ 1) \end{array}$$

The sequent (2) is already our goal sequent, so we are done in 2 steps (and two more for symmetric conjunct that arose from the initial inversion). Compare this to the 10 sequents we derived before hitting our goal without the benefit of focusing!

As an example of something that cannot be proven we consider the type of the S combinator. This is true only if the logic admits contraction.

$$\vdash (A \multimap (B \multimap C)) \multimap ((A \multimap B) \multimap (A \multimap C))$$

This is not a stable sequent, so deciding once again that all atoms should be positive we get

$$A^+ \multimap (B^+ \multimap C^+), A^+ \multimap B^+, \langle A^+\rangle \vdash C^+$$

There are three propositions we could focus on, two on the left and one on the right.

$$\dfrac{\dfrac{\begin{matrix}\vdots\\ \vdash [A^+]\end{matrix} \quad \dfrac{\begin{matrix}\vdots\\ \vdash [B^+]\end{matrix} \quad \dfrac{\dfrac{\begin{matrix}\vdots\\ \langle C^+\rangle \vdash\end{matrix}}{C^+ \vdash}}{[C^+] \vdash}}{[B^+ \multimap C^+] \vdash}}{[A^+ \multimap (B^+ \multimap C^+)] \vdash}}{L_0 \vdash}\ L_0$$

The first two open subgoal can only be closed with identities, as in the last example. After filling in missing antecedents and succedents, we obtain:

$$\dfrac{\Delta, \langle C^+\rangle \vdash \delta}{\Delta, \langle A^+\rangle, \langle B^+\rangle, L_0 \vdash \delta}\ L_0$$

Focusing on the left on $A^+ \multimap B^+$ and on the right on C^+ similarly give us the following two rules:

$$\dfrac{\Delta, \langle B^+\rangle \vdash \delta}{\Delta, \langle A^+\rangle, L_1 \vdash \delta}\ L_1 \qquad\qquad \dfrac{}{\langle C^+\rangle \vdash C^+}\ \mathsf{id}_C^+$$

Our goal sequent is

$$L_0, L_1, \langle A^+\rangle \vdash C^+$$

Initially, we have only one sequent, and only L_0 applies.

(1)	$\langle C^+\rangle \vdash C^+$	(id_C)
(2)	$\langle A^+\rangle, \langle B^+\rangle, L_0 \vdash C^+$	$(L_0\ 1)$

Now we can apply L_1, where $\Delta = \langle A^+\rangle, L_0$ and $\delta = C^+$. We get:

(1)	$\langle C^+\rangle \vdash C^+$	(id_C)
(2)	$\langle A^+\rangle, \langle B^+\rangle, L_0 \vdash C^+$	$(L_0\ 1)$
(3)	$\langle A^+\rangle, L_0, \langle A^+\rangle, L_1 \vdash C^+$	$(L_1\ 2)$

At this point we have reached saturation and *almost* proved our goal sequent. The only problem is that we have two copies of $\langle A^+\rangle$. If we go back and look at the original goal we see that this makes sense: if we had two copies of $\langle A^+\rangle$ it would indeed be provable linearly.

This points out another important observation: if we think about the linear sequent (without the exponential or shifts), as we go up we never duplicate any propositions. The goal sequent has only one left occurrence of A^+, so a sequent with two left occurrences of A^+ as the one labeled (3) could not occur. So we should reject it and (in this example), we reach saturation essentially one step earlier.

The insight here is to obtain an inverse method for a given logic we follow these steps (first without focusing):

1. Obtain the usual backwards sequent calculus without cut, and identity limited to atoms (and prove the admissibility of cut and general identity).
2. Label the left- and right-subformulas of the goal sequent.
3. Derive specialized inference rules for each label, and then discard the general rules.
4. Consider any logic-specific additions or modifications of the specialized rules.
5. Saturate the space of sequents derivable with the specialized rules. Even if the logic is undecidable, we can explore the search space by forward reasoning although it may not saturate.
6. If we find a proof of the goal sequent, we succeed.
7. If we saturate without generating the goal sequent, we fail

This is modified slightly for focusing, because we need to generate (and prove!) a focused version of our logic first. Then we generate "big-step" rules that go from stable sequent to stable sequent. The (non-atomic) propositions in the new stable sequent are then named and according to their sidedness focused on to derive more rules.

4 Strict, Affine, and Structural Logic

Assume we have a logic with contraction, such as strict logic. Then we just add the rule of contraction

$$\frac{\Delta, A, A \vdash C}{\Delta, A \vdash C}\ \mathsf{contract}$$

In this system, because we apply the rules from the premises to the conclusion this actually *is* contraction—usually we use it to achieve duplication of a proposition. In the absence of quantifiers (as in this lecture), we could also just treat antecedents as set and write $\Delta_1 \cup \Delta_2$ instead of Δ_1, Δ_2 whenever they are combined. We would then never have more than one copy of a contractible proposition in a stable sequent.

If we have weakening, matters are a bit more complicated. For example, we should not have rules such as

$$\frac{}{\Delta, \langle A^+ \rangle \vdash A^+}\ \mathsf{id}_A^+$$

That's because even if we have a finite set of labels, there are still many possibilities for Δ. We don't want to enumerate them. We can think of it this way: in backward reasoning, we postpone weakening all the way to the leaves of the proof tree (id or $\mathbf{1}R$). In forward reasoning, we also postpone weakening, but downwards, towards the root of the proof tree.

So where exactly do we finally need to apply weakening? One situation is where we have derived something that can be weakened to our goal sequent. We capture this with a subsumption relation: $(\Delta \vdash A) \leq (\Delta' \vdash A')$ if $\Delta \subseteq \Delta'$ and $A = A'$. Whenever we apply an inference, we can check three properties:

Forward Subsumption: If inference yields $\Delta' \vdash A'$ and there is a sequent $\Delta \vdash A$ in our database such that $\Delta \vdash A \leq \Delta' \vdash A'$ then we do not add the new sequent. We already know something stronger.

Backward Subsumption: If the inference yields $\Delta \vdash A$ and there is a sequent $\Delta' \vdash A'$ in our database such that $\Delta \vdash A \leq \Delta' \vdash A'$ then we replace the old sequent by the newer (stronger) one.

If inference yields $\Delta \vdash A$ and $(\Delta \vdash A) \leq G$ where G is the goal sequent, we succeed.

This is an example of the general principle of *subsumption* in forward inference. Towards saturation, we don't check facts in the database for *equality*, but a more general *subsumption* criterion for redundancy. What that might be may change from inference system to inference system.

Let's try this with the quintessential property that is true in *affine logic* but not in linear logic (writing $A \multimap B$ for affine implication):

$$\vdash A \multimap (B \multimap A)$$

In the small-step system, we introduce two names

$$\begin{array}{rcl} R_0 & = & A^L \multimap R_1 \\ R_1 & = & B^L \multimap A^R \end{array}$$

We generate the rules below. There is no identity for B because it occurs only as B^L and not B^R.

$$\frac{}{A \vdash A}\ \mathsf{id}_A \qquad \frac{\Delta, A \vdash R_1}{\Delta \vdash R_0}\ {\multimap}R_0 \qquad \frac{\Delta, B \vdash A}{\Delta \vdash R_1}\ {\multimap}R_1$$

From $A \vdash A$ we cannot apply a single rule! That's problematics because our proposition actually holds in affine logic. In this example it happens that $A \vdash A$ can be weakened to $\Delta, B \vdash A$ with $\Delta = A$.

We can rectify this by allowing weakening when matching against the premises of rules: B doesn't have to be in the sequent. Allowing this we would derive $A \vdash R_1$ and then $\cdot \vdash R_0$, which is our goal sequent. We could also generate another rule that accounts for B being absent.

$$\frac{\Delta \vdash A}{\Delta \vdash R_1} \; -R_1'$$

This would get awkward however when we move to focusing since there might be too many variants of the rules.

Returning to our example, if we apply inversion we obtain the stable goal sequent

$$\langle A^+ \rangle, \langle B^+ \rangle \vdash A^+$$

We can only focus on A^+ on the right, which gives us

$$\frac{}{\langle A^+ \rangle \vdash A^+} \; \mathsf{id}_A^+$$

and $(\langle A^+ \rangle \vdash A^+) \leq (\langle A^+ \rangle, \langle B^+ \rangle \vdash A^+)$.

Also, if we generate a right rule for external choice $A \mathbin{\&} B$ we can no longer require the two branches to have the same antecedents. We define $\Delta_1 \; \mathsf{max} \; \Delta_2$ for multisets to take the maximum of the multiplicity of each element in the two multisets. Then we have the forward rule

$$\frac{\Delta_1 \vdash A \quad \Delta_2 \vdash B}{\Delta_1 \; \mathsf{max} \; \Delta_2 \vdash A \mathbin{\&} B} \; \&R$$

When mixing logics, for example, in the adjoint framework, we have to combine the various considerations.

5 $\top$ and $\mathbf{0}$ Revisited

We didn't cover this in lecture, but how would we handle

$$\frac{}{\Delta, \mathbf{0} \vdash \delta} \; \mathbf{0}L$$

in the forward direction? We do not want to enumerate possible antecedents or succedents, we want to leave them open. Then we would have something like

$$\frac{}{\mathbf{0} \vdash^W \cdot} \; \mathbf{0}L \qquad \frac{}{\cdot \vdash^W \top} \; \top R$$

where the W on the sequent indicates that the sequent can be weakenend (in antecedent or succedent). Then we precise sequents and weakenable sequents and we have to carefully define rule application in the mixed case and investigate how this attribute of sequents propagates.

Alternatively, we might be able to introduce *metavariables* D and d to stand for an arbitrary Δ and δ respectively and write these as

$$\frac{}{D, \mathbf{0} \vdash d}\ \mathbf{0}L \qquad \frac{}{D \vdash \top}\ \top R$$

and instantiate them as part of the rule application process.

References

Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3):197–347, 1992.

Jean-Marc Andreoli. Focussing and proof construction. *Annals of Pure and Applied Logic*, 107(1–3):131–163, 2001.

Stephen Brookes. A semantics for concurrent separation logic. *Theoretical Computer Science*, 365(1–3):227–270, 2007.

Kaustuv Chaudhuri. *The Focused Inverse Method for Linear Logic*. PhD thesis, Carnegie Mellon University, December 2006. Available as technical report CMU-CS-06-162.

Kaustuv Chaudhuri and Frank Pfenning. A focusing inverse method prover for first-order linear logic. In R.Nieuwenhuis, editor, *Proceedings of the 20th International Conference on Automated Deduction (CADE-20)*, pages 69–83, Tallinn, Estonia, July 2005a. Springer Verlag LNCS 3632.

Kaustuv Chaudhuri and Frank Pfenning. Focusing the inverse method for linear logic. In L.Ong, editor, *Proceedings of the 14th Annual Conference on Computer Science Logic (CSL'05)*, pages 200–215, Oxford, England, August 2005b. Springer Verlag LNCS 3634.

Anatoli Degtyarev and Andrei Voronkov. The inverse method. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 4, pages 181–272. Elsevier Science and MIT Press, 2001.

Joshua Hodas and Dale Miller. Logic programming in a fragment of intuitionistic linear logic. *Information and Computation*, 110(2):327–365, 1994. A preliminary version appeared in the Proceedings of the Sixth Annual IEEE Symposium on Logic in Computer Science, pages 32–42, Amsterdam, The Netherlands, July 1991.

Jack Hughes and Dominic Orchard. Resourceful program synthesis from graded linear types. In Maribel Fernández, editor, *30th International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR 2020)*, pages 151–170, Bologna, Italy, September 2020. LNCS 12561.

Pablo López, Frank Pfenning, Jeff Polakow, and Kevin Watkins. Monadic concurrent linear logic programming. In A.Felty, editor, *Proceedings of the 7th International Symposium on Principles and Practice of Declarative Programming (PPDP'05)*, pages 35–46, Lisbon, Portugal, July 2005. ACM Press.

Sergei Maslov. The inverse method of establishing deducibility in the classical predicate calculus. *Soviet Mathematical Doklady*, 5:1420–1424, 1964.

Maria Inês Melo e Sousa. *Synthesis of Programs from Linear Types*. M.Sc. thesis, University of Porto, 2021.

Peter O'Hearn. Resources, concurrency and local reasoning. *Theoretical Computer Science*, 375(1–3):271–307, 2007.

John C. Reynolds. Separation logic: A logic for shared mutable data structures. In *Proceedings of the 17th Symposium on Logic in Computer Science*, pages 55–74, Copenhagen, Denmark, July 2002. IEEE Computer Society.

Andrei Voronkov. Theorem proving in non-standard logics based on the inverse method. In Deepak Kapur, editor, *11th International Conference on Automated Deduction (CADE 1992)*, pages 648–662, Saratoga Springs, New York, 1992. Springer LNAI 607.