Polarity in Proof Theory and Programming

Noam Zeilberger

2 September 2013*

1 Introduction

I'm going to be talking about the logical phenomenon of *polarity*, while trying to give a taste of what it has to do with programming. "Polarity" is a very overloaded word, so I want to begin by stating that we take it to mean something fairly specific in the proof theory of linear logic. To a first approximation, we say that the connectives

$$\otimes, \oplus, \exists$$

have *positive* polarity since they are *invertible* (or "reversible") on the left, while the connectives

$$\parr, \&, \forall$$

have *negative* polarity since they are invertible on the right. That is, all of the rules

$$\frac{A, B, \Gamma \vdash \Delta}{A \otimes B, \Gamma \vdash \Delta} \qquad \frac{\Gamma \vdash \Delta, A, B}{\Gamma \vdash \Delta, A \parr B}$$

$$\frac{A, \Gamma \vdash \Delta \quad B, \Gamma \vdash \Delta}{A \oplus B, \Gamma \vdash \Delta} \qquad \frac{\Gamma \vdash \Delta, A \quad \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \& B}$$

$$\frac{A, \Gamma \vdash \Delta}{\exists x.A, \Gamma \vdash \Delta} \qquad \frac{\Gamma \vdash \Delta, A}{\Gamma \vdash \Delta, \forall x.A}$$

are valid both when interpreted in the usual direction from premises to conclusion, and in the opposite direction. We'll get back to these proof-theoretic aspects later on, since they are important. However, something that I'm going to emphasize is that polarity is more than just a *formal* aspect of linear logic, but rather a general *perspective* on logical systems. One of the fundamental questions in logic is

where do the rules of logic *come* from?

*These notes are a revised version of a talk I gave at the *Summer School on Linear Logic and Geometry of Interaction* held at Torino, Italy from 27–31 August 2013.

Now, I'm not going to pretend to answer this question for you here. However, I do believe that the body of work on linear logic has demonstrated that *if* there were a good story for why logic is the way it is, then the concept of polarity would be an important component of this story. In fact, one of the points I want to emphasize is that polarity is *not* an entirely new phenomenon in linear logic, but rather a clarification of old ideas in logic and semantics of programming languages. In particular, I'm going to discuss the relationship between polarity and *continuation semantics* in the sense of Strachey and Wadsworth, as well as its relationship to the so-called "meaning-explanations" for logic and type theory popular in the 1970s.

With this by way of abstract motivation, I want to begin by attacking a slightly more modest question, namely

2 The meaning of existence

. . . By this I mean of course the meaning of the existential quantifier!

So, let's go back a hundred years in time, back to the debate between Hilbert's formalists and Brouwer's intuitionists. One of the central issues of this debate was the meaning of *existence proofs*, i.e., of proofs of existential statements

$$\exists x.A$$

To Brouwer, a proof of an existential statement had to be *constructive*, that is, it corresponded to a mental process for building some mathematical object t for which the property

$$A(t)$$

could be intuitively verified. In contrast, Hilbert saw no reason for such a restriction: to prove that

$$\exists x.A$$

it suffices to show that the non-existence of such an object t would lead to a contradiction.

Perhaps one of the reasons why this debate was never really settled is that it hinges on conventions of language. However, an important milestone in clarifying the debate was the so-called *double-negation* translation of Kolmogorov (1925), who showed how to translate propositions in the classical logic of Hilbert into the intuitionistic logic of Brouwer, while preserving provability. Consider

the following translation, defined by induction on formulas:

$$\begin{aligned}
p^K &= \neg\neg p \\
(A \wedge B)^K &= \neg\neg(A^K \wedge B^K) \\
(A \vee B)^K &= \neg\neg(A^K \vee B^K) \\
(A \to B)^K &= \neg\neg(A^K \to B^K) \\
(\forall x.A)^K &= \neg\neg\forall x.A^K \\
(\exists x.A)^K &= \neg\neg\exists x.A^K
\end{aligned}$$

As you can see, the idea is to simply insert a double-negation before every connective, and before logical atoms. A few years later, both Gentzen (1933) and Gödel (1933) independently discovered more frugal translations. Gentzen's translation is defined like so:

$$\begin{aligned}
p^G &= \neg\neg p \\
(A \wedge B)^G &= A^G \wedge B^G \\
(A \vee B)^G &= \neg(\neg A^G \wedge \neg B^G) \\
(A \to B)^G &= A^G \to B^G \\
(\forall x.A)^G &= \forall x.A^G \\
(\exists x.A)^G &= \neg\forall x.\neg A^G
\end{aligned}$$

Gödel's is almost identical, except for taking the clause

$$(A \to B)^{\ddot{G}} = \neg(A^{\ddot{G}} \wedge \neg B^{\ddot{G}})$$

The key properties to establish for all such translations $(-)^\dagger$ are the following pair of facts:

1. $\vdash^c A$ implies $\vdash^i A^\dagger$, and
2. $\vdash^c (A \equiv A^\dagger)$

Together, these facts imply that $\vdash^c A$ iff $\vdash^i A^\dagger$.

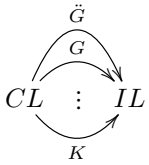
Now, one way of reading these results is that they serve as an interpreter for the classical mathematician when speaking to the intuitionistic mathematician. Thus, the intuitionist is told that when the classical logician says

$$\exists x.A$$

what they really "mean to say" is

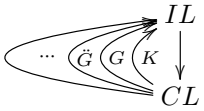
$$\neg\forall x.\neg A$$

and so on. But of course the problem with this reading of the double-negation translations is that there is more than one! How to decide between Kolmogorov's translation, Gentzen's, or Gödel's (or Kuroda's, or Krivine's, ...)?



And for that matter, very often a classical mathematician and an intuitionistic mathematician would *agree* on the truth of a proposition, and even on its proof. Why is a translation needed at all in such cases?

I would like you to consider that the right way of looking at the double-negation translations is rather like so:



The implicit inclusion from intuitionistic to classical propositions can be seen as a coercion which *forgets* some distinctions. For example, the intuitionistic propositions

$$A \to A \qquad \neg A \vee A \qquad \neg\neg A \to A$$

can all be seen as different *refinements* of the classical tautology

$$\neg A \vee A$$

since they become logically equivalent when interpreted in CL. Likewise, the classical logician's statement that

$$\exists x.A$$

should just be interpreted as a *rough* statement by the intuitionist. It can be refined into many different intuitionistic statements such as

$$\exists x.A \qquad \neg\neg\exists x.A \qquad \neg\forall x.\neg A \qquad \exists x.\neg\neg A$$

and so on, which may or may not be valid in different contexts.

Later on I'm going to argue that *polarization* is the same kind of phenomenon. That is, it involves making additional distinctions, which refine the meaning of logical systems.

3 Continuation semantics

First, though, I want to review some classical ideas from the theory of programming languages.

The basic problem of *denotational semantics* is to assign a meaning

$$[\![e]\!]$$

to syntactic expressions e of some language L. As a warmup exercise, let us consider a very simple language (L_1) of arithmetic expressions:

$$e ::= \overline{n} \mid e_1 + e_2 \mid e_1 \times e_2$$

An expression can be interpreted as a natural number, defined inductively as follows:

$$\begin{aligned} [\![e]\!]_1 &: \mathbb{N} \\ [\![\overline{n}]\!]_1 &= n \\ [\![e_1 + e_2]\!]_1 &= [\![e_1]\!]_1 + [\![e_2]\!]_1 \\ [\![e_1 \times e_2]\!]_1 &= [\![e_1]\!]_1 \times [\![e_2]\!]_1 \end{aligned}$$

That is, numerals are interpreted as the natural numbers which they denote, and the addition and multiplication operations on expressions are interpreted as addition and multiplication of numbers. Of course, this all looks very circular and vacuous, like Tarski's definition,

"Snow is white" is true if and only if snow is white

or better, Girard's

"A brocolli B" is true if and only if "A" is true brocolli "B" is true.

To get a bit less trivial, let's now consider extending the language with *variable binding* (L_2):

$$e ::= \cdots \mid \textbf{let } x = e_1 \textbf{ in } e_2 \mid x$$

For example, we can now write the expression

$$\textbf{let } x = \overline{2} \textbf{ in let } y = \overline{3} \textbf{ in } x + y$$

which presumably we want to have the meaning as

$$\overline{5}$$

The real question of denotational semantics is, how do we define the meaning operator in a *compositional* way? That is, we want to build up the meaning of e out of the meanings of its subexpressions. Well, the idea is that rather than interpreting e directly as a natural number, we interpret it as a function from

environments to natural numbers.

$$\begin{aligned}
[\![e]\!]_2 &: (\mathit{Var} \xrightarrow{\text{fin}} \mathbb{N}) \to \mathbb{N} \\
[\![\overline{n}]\!]_2\gamma &= n \\
[\![e_1 + e_2]\!]_2\gamma &= [\![e_1]\!]_2\gamma + [\![e_2]\!]_2\gamma \\
[\![e_1 \times e_2]\!]_2\gamma &= [\![e_1]\!]_2\gamma \times [\![e_2]\!]_2\gamma \\
[\![x]\!]_2\gamma &= \gamma(x) \\
[\![\textbf{let } x = e_1 \textbf{ in } e_2]\!]_2\gamma &= [\![e_2]\!]_2\gamma[x \mapsto [\![e_1]\!]_2\gamma]
\end{aligned}$$

Let $[]$ stand for the empty environment. You can verify that

$$[\![\textbf{let } x = \overline{2} \textbf{ in let } y = \overline{3} \textbf{ in } x + y]\!]_2[] = 5$$

since

$$[\![x + y]\!]_2[x \mapsto 2, y \mapsto 3] = 5$$

Also observe that we recover the original meaning operator by applying to the empty environment, in the sense that

$$[\![e]\!]_2[] = [\![e]\!]_1$$

for all $e \in L_1$.

Next consider extending the language with *division* (L_3):

$$e ::= \cdots \mid e_1 \div e_2$$

The question now, of course, is how should we interpret division-by-zero? Well, one natural idea is to enlarge the type of the interpretation, so that rather than denoting functions from environments to natural numbers, expressions denote functions from environments to "natural numbers or division-by-zero errors",

$$[\![e]\!] : (\mathit{Var} \xrightarrow{\text{fin}} \mathbb{N}) \to (\mathbb{N} + 1)$$

Essentially this requires us to extend the arithmetic operations from $\mathbb{N}$ to the larger type $\mathbb{N} + 1$, for example extending the definition of addition like so:

$$\begin{array}{ccccc}
\mathrm{inl}(n_1) & + & \mathrm{inl}(n_2) & = & \mathrm{inl}(n_1 + n_2) \\
\mathrm{inl}(n_1) & + & \mathrm{inr}(*) & = & \mathrm{inr}(*) \\
\mathrm{inr}(*) & + & \mathrm{inl}(n_2) & = & \mathrm{inr}(*) \\
\mathrm{inr}(*) & + & \mathrm{inr}(*) & = & \mathrm{inr}(*)
\end{array}$$

This works, but is a bit annoying: why should we have to redefine the arithmetic operations on "pseudo numbers" they were never meant to support?

A different possibility is to define a so-called *continuation semantics*. Let $\perp$ stand for an arbitrary type, which we call the *answer type*, assumed to contain at least one predefined value $d : \perp$ (for "dummy" or "default"). The idea of

continuation semantics—also called *continuation-passing style*—is that we can interpret expressions into the answer type, given an additional function of type

$$\mathbb{N} \to \bot$$

as an argument, called the *continuation*. That is, the semantic operator has type

$$[\![e]\!]_3 : (\mathit{Var} \xrightarrow{\text{fin}} \mathbb{N}) \to (\mathbb{N} \to \bot) \to \bot$$

We define $[\![e]\!]_3$ as follows:

$$\begin{aligned}
[\![\overline{n}]\!]_3 \gamma\, k &= k\, n \\
[\![e_1 + e_2]\!]_3 \gamma\, k &= [\![e_1]\!]_3 \gamma\, \lambda n_1 . [\![e_2]\!]_3 \gamma\, \lambda n_2 . k(n_1 + n_2) \\
[\![e_1 \times e_2]\!]_3 \gamma\, k &= [\![e_1]\!]_3 \gamma\, \lambda n_1 . [\![e_2]\!]_3 \gamma\, \lambda n_2 . k(n_1 \times n_2) \\
[\![x]\!]_3 \gamma\, k &= k(\gamma(x)) \\
[\![\mathbf{let}\ x = e_1\ \mathbf{in}\ e_2]\!]_3 \gamma\, k &= [\![e_1]\!]_3 \gamma\, \lambda n_1 . [\![e_2]\!]_3 \gamma[x \mapsto n_1]\, k \\
[\![e_1 \div e_2]\!]_3 \gamma\, k &= [\![e_1]\!]_3 \gamma\, \lambda n_1 . [\![e_2]\!]_3 \gamma\, \lambda n_2 . \text{if } n_2 > 0 \text{ then } k(n_1 \div n_2) \text{ else } d
\end{aligned}$$

If you haven't seen this kind of thing before, it might take some getting used to. Here's an example:

$$\begin{aligned}
[\![\mathbf{let}\ x = \overline{0}\ \mathbf{in}\ \overline{3} \div x]\!]\, [\,]\, k &= [\![\overline{0}]\!]\, [\,]\, \lambda n_1 . [\![\overline{3} \div x]\!]\, [x \mapsto n_1]\, k \\
&= [\![\overline{3} \div x]\!]\, [x \mapsto 0]\, k \\
&= \ldots \\
&= [\![x]\!]\, [x \mapsto 0]\, \lambda n_2 . \text{if } n_2 > 0 \text{ then } k(3 \div n_2) \text{ else } d \\
&= d
\end{aligned}$$

Observe, though, that once again we can recover the previous semantics (up to isomorphism) by a clever instantiation: taking $\bot = \mathbb{N} + 1$ and $d = \text{inr}(*)$, we have

$$[\![e]\!]_3\, \gamma\, (\lambda x . \text{inl}\ x) = \text{inl}([\![e]\!]_2\, \gamma)$$

for all expressions $e \in L_2$ and all environments γ.

Now for a philosophical question: what is the meaning of "$\mathbf{let}\ x = \overline{3} \div \overline{0}\ \mathbf{in}\ \overline{42}$"? A bit of calculation will show you that our semantics decides that this expression always signals a division-by-zero error. On the other hand, one could imagine a language where "$\mathbf{let}\ x = \overline{3} \div \overline{0}\ \mathbf{in}\ \overline{42}$" has the same meaning as "$\overline{42}$". In programming languages theory, this is the difference between binding of variables *by-value* and *by-name*.

It is actually quite easy to modify our denotational semantics to implement by-name variable binding. The type of the alternative meaning operator becomes:

$$[\![e]\!]'_3 : (\mathit{Var} \xrightarrow{\text{fin}} (\mathbb{N} \to \bot) \to \bot) \to (\mathbb{N} \to \bot) \to \bot$$

Then the only clauses that have to change are the ones dealing with variables:

$$[\![x]\!]'_3\gamma\, k = \gamma\, x\, k$$
$$[\![\textbf{let } x = e_1 \textbf{ in } e_2]\!]'_3\gamma\, k = [\![e_2]\!]'_3\gamma[x \mapsto [\![e_1]\!]'_3\gamma]\, k$$

You can verify that with this definition, the meaning of "**let** $x = \overline{3} \div \overline{0}$ **in** $\overline{42}$" is the same as the meaning of "$\overline{42}$".

Let me point out something now which may already be obvious. I'm using the symbol "$\bot$" in order to highlight an analogy with the classical double-negation translations, of course. When we introduced our continuation semantics, we went from interpreting expressions as [functions from environments to] natural numbers, to interpreting them as [functions from environments to] double-negated natural numbers. Again, $\bot$ is called an "answer type" in the theory of continuations. What is absolutely crucial, though, is that this answer type is *not* meant to stand for a contradiction in the classical sense of an *empty* type. Indeed, in this particular semantics we postulated that $\bot$ was inhabited with at least one element. It is better to think of $\bot$ as a sort of a universe which you can enter but never escape.

To conclude this section, I want to mention one more idea from the theory of programming languages, due to the late John Reynolds. Although continuation semantics is a very powerful and general technique, it can be a bit unsatisfying from a foundational viewpoint, since it seems to rely on a higher-order metalanguage. What exactly is this space of functions

$$\mathbb{N} \to \bot$$

and what manipulations are allowed on it? Certainly we could answer this question *precisely* in ZFC set theory for example, but the answer would be *different* if we took the metalanguage to be, say, OCaml. One would hope that these kinds of choices would be irrelevant to the meaning of expressions in such a simple language as L_3. And in any case, after all it should be possible (let us think like computer scientists) to *compile* expressions of L_3 into instructions of some low-level model of computation, such as Turing machines.

Reynolds' *defunctionalization* is an abstract technique which precisely addresses this objection. I won't have time to explain defunctionalization in detail, but the basic idea is that the specific continuations

$$k : A \to \bot$$

introduced in a higher-order definition can be enumerated into some small type B, while at the same time one defines a function

$$\langle - \mid - \rangle : A \times B \to \bot$$

that explains how to apply these specific continuations to values. (Note we are taking $A = \mathbb{N}$ in the semantics of L_3.) Then, one simply replaces all references

to the type $A \to \bot$ with the type B, and all instances of function application $k(v)$ by calls to the "apply function" $\langle v \mid k \rangle$.

The technique is essentially mechanical. Let me assert now that by defunctionalizing the interpretation $[\![-]\!]_3$, we arrive at the following "abstract machine":

Defunctionalized continuations:

$$\begin{aligned} k \;::=\; & K \mid P_1(\gamma, k, e) \mid P_2(\gamma, n, k) \\ & \mid T_1(\gamma, k, e) \mid T_2(\gamma, n, k) \\ & \mid D_1(\gamma, k, e) \mid D_2(\gamma, n, k) \\ & \mid L(\gamma, x, k, e) \end{aligned}$$

Evaluation function:

$$\begin{aligned} \langle\!\langle \overline{n} \mid \gamma \mid k \rangle\!\rangle &\rightsquigarrow \langle n \mid k \rangle \\ \langle\!\langle x \mid \gamma \mid k \rangle\!\rangle &\rightsquigarrow \langle \gamma(x) \mid k \rangle \\ \langle\!\langle e_1 + e_2 \mid \gamma \mid k \rangle\!\rangle &\rightsquigarrow \langle\!\langle e_1 \mid \gamma \mid P_1(\gamma, k, e_2) \rangle\!\rangle \\ \langle\!\langle e_1 \times e_2 \mid \gamma \mid k \rangle\!\rangle &\rightsquigarrow \langle\!\langle e_1 \mid \gamma \mid T_1(\gamma, k, e_2) \rangle\!\rangle \\ \langle\!\langle e_1 \div e_2 \mid \gamma \mid k \rangle\!\rangle &\rightsquigarrow \langle\!\langle e_1 \mid \gamma \mid D_1(\gamma, k, e_2) \rangle\!\rangle \\ \langle\!\langle \textbf{let } x = e_1 \textbf{ in } e_2 \mid \gamma \mid k \rangle\!\rangle &\rightsquigarrow \langle\!\langle e_1 \mid \gamma \mid L(\gamma, x, k, e_2) \rangle\!\rangle \end{aligned}$$

Apply function:

$$\begin{aligned} \langle n_1 \mid P_1(\gamma, k, e_2) \rangle &\rightsquigarrow \langle\!\langle e_2 \mid \gamma \mid P_2(\gamma, n_1, k) \rangle\!\rangle \\ \langle n_1 \mid T_1(\gamma, k, e_2) \rangle &\rightsquigarrow \langle\!\langle e_2 \mid \gamma \mid T_2(\gamma, n_1, k) \rangle\!\rangle \\ \langle n_1 \mid D_1(\gamma, k, e_2) \rangle &\rightsquigarrow \langle\!\langle e_2 \mid \gamma \mid D_2(\gamma, n_1, k) \rangle\!\rangle \\ \langle n_1 \mid L(\gamma, x, k, e_2) \rangle &\rightsquigarrow \langle\!\langle e_2 \mid \gamma[x \mapsto n_1] \mid k \rangle\!\rangle \\ \langle n_2 \mid P_2(\gamma, n_1, k) \rangle &\rightsquigarrow \langle n_1 + n_2 \mid k \rangle \\ \langle n_2 \mid T_2(\gamma, n_1, k) \rangle &\rightsquigarrow \langle n_1 \times n_2 \mid k \rangle \\ \langle n_2 \mid D_2(\gamma, n_1, k) \rangle &\rightsquigarrow \text{if } n_2 > 0 \text{ then } \langle n_1 \div n_2 \mid k \rangle \text{ else } D \end{aligned}$$

Theorem: $\langle\!\langle e \mid [\,] \mid K \rangle\!\rangle$ eventually reduces to D or to $\langle n \mid K \rangle$ for some n.

4 Inversion principles and the justification of the logical laws

Much has been made of an offhand remark of Gentzen in his article on natural deduction (1935):

> The introduction rules give, so to say, a definition of the constant in question, and the elimination rules are in the end only consequences thereof. *[Die Einführungen stellen sozusagen die "Definitionen" der betreffenden Zeichen dar, und die Beseitigungen sind letzen Endes nur Konsequenzen hiervon...]*

Dag Prawitz was one of the first to try to justify Gentzen's remark, by applying a so-called *inversion principle*. Prawitz defined a *canonical proof* of a proposition as one ending in an introduction rule. Then the inversion principle states that an elimination rule is justified, just in case a proof of the conclusion can be derived directly from *canonical* proofs of the premises (i.e., without resort to the elimination rule itself). Rather than trying to explain exactly what this means, let me illustrate with an example.

Consider the standard introduction and elimination rules for conjunction:

$$\frac{\vdash A \quad \vdash B}{\vdash A \wedge B} \qquad \frac{\vdash A \wedge B}{\vdash A} \quad \frac{\vdash A \wedge B}{\vdash B}$$

According to Prawitz, the introduction rule

$$\frac{\vdash A \quad \vdash B}{\vdash A \wedge B}$$

justifies (for example) the first elimination rule

$$\frac{\vdash A \wedge B}{\vdash A}$$

since given a canonical proof of $\vdash A \wedge B$ one can extract a proof of $\vdash A$:

$$\begin{array}{c} \vdots\alpha \quad \vdots\beta \\ \dfrac{\vdash A \quad \vdash B}{\vdash A \wedge B} \\ \vdash A \end{array} \quad \mapsto \quad \begin{array}{c} \vdots\alpha \\ \vdash A \end{array}$$

Similarly, the introduction rule for implication

$$\frac{\begin{array}{c}[\vdash A] \\ \vdash B\end{array}}{\vdash A \to B}$$

justifies the elimination rule of modus ponens

$$\frac{\vdash A \to B \quad \vdash A}{\vdash B}$$

as illustrated by the following argument:

$$\begin{array}{c} [\vdash A] \\ \vdots\beta \\ \dfrac{\vdash B}{\vdash A \to B} \quad \vdots\alpha \\ \dfrac{\qquad\qquad}{\vdash B} \end{array} \quad \vdash A \quad \mapsto \quad \begin{array}{c} \vdots\alpha \\ \vdash A \\ \vdots\beta \\ \vdash B \end{array}$$

As a last example, the pair of introduction rules for disjunction

$$\frac{\vdash A}{\vdash A \vee B} \quad \frac{\vdash B}{\vdash A \vee B}$$

justify elimination by case-analysis,

$$\dfrac{\vdash A \vee B \qquad \begin{array}{c}[\vdash A]\\ \vdash C\end{array} \qquad \begin{array}{c}[\vdash B]\\ \vdash C\end{array}}{\vdash C}$$

since

$$\dfrac{\begin{array}{c}\vdots\alpha\\ \vdash A\end{array} \qquad \begin{array}{c}[\vdash A]\\ \vdots\beta\\ \vdash C\end{array} \qquad \begin{array}{c}[\vdash B]\\ \vdots\gamma\\ \vdash C\end{array}}{\begin{array}{c}\vdash A \vee B\\ \vdash C\end{array}} \quad \mapsto \quad \begin{array}{c}\vdots\alpha\\ \vdash A\\ \vdots\beta\\ \vdash C\end{array}$$

and

$$\dfrac{\begin{array}{c}\vdots\alpha'\\ \vdash B\end{array} \qquad \begin{array}{c}[\vdash A]\\ \vdots\beta\\ \vdash C\end{array} \qquad \begin{array}{c}[\vdash B]\\ \vdots\gamma\\ \vdash C\end{array}}{\begin{array}{c}\vdash A \vee B\\ \vdash C\end{array}} \quad \mapsto \quad \begin{array}{c}\vdots\alpha'\\ \vdash B\\ \vdots\gamma\\ \vdash C\end{array}$$

As you can see, these "justification" steps basically correspond to β-reductions in lambda calculus. I'm not going to argue that this really provides a philosophical "justification" of the elimination rules, but instead move on to discussing a later analysis by Michael Dummett, which was inspired by Prawitz's but goes considerably deeper.

Rather than confining attention to the elimination rules, Dummett was interested in the justification of *arbitrary* logical inferences. The essence of his idea was that Prawitz's inversion principle could be applied more broadly if one starts with a more restricted notion of canonical proof. As an example, consider the inference

$$\dfrac{\vdash A \wedge (B \vee C)}{\vdash (A \wedge B) \vee (A \wedge C)}$$

and suppose that the premise has a proof ending in an introduction rule:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\alpha\\ \vdash A\end{array} \qquad \begin{array}{c}\vdots\beta\\ \vdash B \vee C\end{array}}{\vdash A \wedge (B \vee C)}}{\vdash (A \wedge B) \vee (A \wedge C)}$$

Well now suppose further that the proof of the right premise continues with another introduction rule:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\alpha\\ \vdash A\end{array} \qquad \dfrac{\begin{array}{c}\vdots\beta'\\ \vdash B\end{array}}{\vdash B \vee C}}{\vdash A \wedge (B \vee C)}}{\vdash (A \wedge B) \vee (A \wedge C)}$$

Then the premises can be reorganized into a direct proof of the conclusion:

$$\dfrac{\dfrac{\vdots\alpha}{\vdash A}\quad\dfrac{\dfrac{\vdots\beta'}{\vdash B}}{\vdash B\vee C}}{\vdash A\wedge(B\vee C)}}{\vdash (A\wedge B)\vee(A\wedge C)}\quad\mapsto\quad\dfrac{\dfrac{\vdots\alpha}{\vdash A}\quad\dfrac{\vdots\beta'}{\vdash B}}{\vdash A\wedge B}}{\vdash (A\wedge B)\vee(A\wedge C)}$$

In the case where the right premise continues with the other introduction rule for disjunction, we can make a similar reduction:

$$\dfrac{\dfrac{\vdots\alpha}{\vdash A}\quad\dfrac{\dfrac{\vdots\gamma}{\vdash C}}{\vdash B\vee C}}{\vdash A\wedge(B\vee C)}}{\vdash (A\wedge B)\vee(A\wedge C)}\quad\mapsto\quad\dfrac{\dfrac{\vdots\alpha}{\vdash A}\quad\dfrac{\vdots\gamma}{\vdash C}}{\vdash A\wedge C}}{\vdash (A\wedge B)\vee(A\wedge C)}$$

We have thus justified the inference

$$\dfrac{\vdash A\wedge(B\vee C)}{\vdash (A\wedge B)\vee(A\wedge C)}$$

by an inversion principle.

As Dummett observed, though, care must be taken in formulating the right definition of "canonical proof". For example, one might argue that the inference

$$\dfrac{\vdash A\to(B\vee C)}{\vdash (A\to B)\vee(A\to C)}$$

is justified, since

$$\dfrac{\dfrac{[\vdash A]}{\vdots\beta}}{\dfrac{\vdash B}{\vdash B\vee C}}}{\vdash A\to(B\vee C)}\quad\mapsto\quad\dfrac{\dfrac{[\vdash A]}{\vdots\beta}}{\dfrac{\vdash B}{\vdash A\to B}}}{\vdash (A\to B)\vee(A\to C)}$$

and

$$\dfrac{\dfrac{[\vdash A]}{\vdots\gamma}}{\dfrac{\vdash C}{\vdash B\vee C}}}{\vdash A\to(B\vee C)}\quad\mapsto\quad\dfrac{\dfrac{[\vdash A]}{\vdots\gamma}}{\dfrac{\vdash C}{\vdash A\to C}}}{\vdash (A\to B)\vee(A\to C)}$$

But obviously this inference is invalid even in classical logic (take $A = B\vee C$).

Dummett's solution was to relax the notion of canonical proof under hypothetical reasoning. Concretely in the case of first-order intuitionistic logic, if we scan a canonical proof from bottom-to-top, we will read a series of introductions of

disjunctions, conjunctions, and existential quantifiers, followed by a $\to I$ or $\forall I$ rule, after which the proof has arbitrary format.

On the other hand, what is so special about introduction rules? Dummett also considered the possibility of fixing the meaning of the connectives by their *elimination* rules. This induces an alternative inversion principle, which again can be used to justify arbitrary inferences (including the introduction rules): an inference is justified, just in case any canonical *consequence* of the conclusion can be directly extracted as a consequence of the premises. Dummett called this eliminations-oriented approach a "pragmatist" meaning-theory, dual to the introductions-oriented "verificationist" meaning-theory.

Again let us illustrate with an example. Suppose we take the elimination rules for conjunction

$$\frac{\vdash A \wedge B}{\vdash A} \qquad \frac{\vdash A \wedge B}{\vdash B}$$

as defining the canonical consequences of $\vdash A \wedge B$. Then the conjunction-introduction rule

$$\frac{\vdash A \quad \vdash B}{\vdash A \wedge B}$$

is justified in the following sense: any canonical consequence of the conclusion must begin by projecting either $\vdash A$ or $\vdash B$, and in either case we can derive the consequence directly from the premises:

$$\frac{\dfrac{\vdash A \quad \vdash B}{\vdash A \wedge B}}{\begin{array}{c}\vdash A \\ \vdots\alpha\end{array}} \quad \mapsto \quad \begin{array}{c}\vdash A \\ \vdots\alpha\end{array}$$

$$\frac{\dfrac{\vdash A \quad \vdash B}{\vdash A \wedge B}}{\begin{array}{c}\vdash B \\ \vdots\beta\end{array}} \quad \mapsto \quad \begin{array}{c}\vdash B \\ \vdots\beta\end{array}$$

As an example of using the "pragmatist" inversion principle to justify a more general inference, the rule

$$\frac{\vdash (A \to B) \wedge (A \to C)}{\vdash A \to B \wedge C}$$

may be justified by the following pair of reductions:

$$\frac{\dfrac{\dfrac{\vdash (A \to B) \wedge (A \to C)}{\vdash A \to B \wedge C} \quad \vdots\alpha \qquad \vdash A}{\vdash B \wedge C}}{\begin{array}{c}\vdash B \\ \vdots\beta\end{array}} \quad \mapsto \quad \frac{\dfrac{\vdash (A \to B) \wedge (A \to C)}{\vdash A \to B} \quad \vdots\alpha \qquad \vdash A}{\begin{array}{c}\vdash B \\ \vdots\beta\end{array}}$$

$$\frac{\dfrac{\dfrac{\vdash (A \to B) \wedge (A \to C)}{\vdash A \to B \wedge C} \quad \vdots\alpha \qquad \vdash A}{\vdash B \wedge C}}{\begin{array}{c}\vdash C \\ \vdots\gamma\end{array}} \quad \mapsto \quad \frac{\dfrac{\vdash (A \to B) \wedge (A \to C)}{\vdash A \to C} \quad \vdots\alpha \qquad \vdash A}{\begin{array}{c}\vdash C \\ \vdots\gamma\end{array}}$$

Finally, Dummett considered how to relate the two "verificationist" and "pragmatist" perspectives on logical inference, and settled on a requirement of "harmony" between them: essentially that the connectives admit both interpretations.

On the other hand, in our post-modern world, we know that rather than always insisting on harmony, sometimes it is better to simply accept *diversity*. Typically, the perspective of linear logic tells us that the "verificationist" definition of conjunction (via its introduction rule) and the "pragmatist" definition of conjunction (via its elimination rules) are both different *refinements* of the intuitionistic concept of conjunction: one corresponds to $\otimes$, the other to $\&$. So now let us finally turn to the polarity phenomenon in linear logic.

5 Polarity and focalization in linear logic

I want to begin by discussing the connection between inversion principles in the sense of Prawitz and Dummett, and the proof-theoretic concept of invertibility. As I mentioned before, in proof theory we say that an inference rule is *invertible* if its conclusion implies its premises. For example, the $\&$-right rule in linear logic

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \& B, \Delta}\ \&R$$

is invertible, since from any proof of $\Gamma \vdash A \& B, \Delta$ may be extracted proofs of $\Gamma \vdash A, \Delta$ and $\Gamma \vdash B, \Delta$. One way to see this formally is by invoking cut:

$$\frac{\Gamma \vdash A \& B, \Delta \quad \dfrac{A \vdash A}{A \& B \vdash A}\ \&L_1}{\Gamma \vdash A, \Delta}\ cut \qquad \frac{\Gamma \vdash A \& B, \Delta \quad \dfrac{B \vdash B}{A \& B \vdash B}\ \&L_2}{\Gamma \vdash B, \Delta}\ cut$$

A standard convention is to indicate invertible rules with a double horizontal line, for example the following are invertible rules:

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \& B, \Delta}\ \&R \qquad \frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \otimes B \vdash \Delta}\ \otimes L \qquad \frac{\Gamma \vdash A \multimap B, \Delta}{\Gamma, A \vdash B, \Delta}\ \multimap R$$

Now, although historically we speak in terms of provability, in modern terms really invertibility is about an *isomorphism of judgments*. For example, the invertibility of $\multimap R$ says that we have equalities

$$\frac{\dfrac{\overset{\beta}{\Gamma, A \vdash B, \Delta}}{\Gamma \vdash A \multimap B, \Delta}\ \multimap R \quad \dfrac{\overline{A \vdash A} \quad \overline{B \vdash B}}{A \multimap B, A \vdash B}\ \multimap L}{\Gamma, A \vdash B, \Delta}\ cut \quad = \quad \overset{\beta}{\Gamma, A \vdash B, \Delta}$$

and

$$\Gamma \vdash A \overset{\eta}{\multimap} B, \Delta \quad = \quad \dfrac{\dfrac{\Gamma \vdash A \overset{\eta}{\multimap} B, \Delta \qquad \dfrac{\overline{A \vdash A} \quad \overline{B \vdash B}}{A \multimap B, A \vdash B}\ \multimap L}{\Gamma, A \vdash B, \Delta}\ cut}{\Gamma \vdash A \multimap B, \Delta}\ \multimap R$$

for all proofs β and η of $\Gamma, A \vdash B, \Delta$ and $\Gamma \vdash A \multimap B, \Delta$ respectively.

Again, a basic observation about linear logic is that the connectives

$$\otimes, \oplus, \exists$$

are invertible on the left (i.e., their left rules are invertible), while the connectives

$$\parr, \&, \forall$$

are invertible on the right. In turn, $\otimes, \oplus, \exists$ are *not* invertible on the right, while $\parr, \&, \forall$ are not invertible on the left. For example, the $\otimes$-right rule

$$\frac{\Gamma_1 \vdash A, \Delta_1 \quad \Gamma_2 \vdash B, \Delta_2}{\Gamma_1, \Gamma_2 \vdash A \otimes B, \Delta_1, \Delta_2}\ \otimes R$$

is not invertible, as we can see by considering the sequent $A \otimes B \vdash A \otimes B$: it is derivable, yet in general there is no proof ending with the $\otimes R$ rule.

This natural division of the connectives was first discovered in the context of linear logic proof search by Jean-Marc Andreoli, who called these two groups of connectives *synchronous* and *asynchronous*. His motivation was roughly the following.

Suppose that you start with an arbitrary formula A, and want to try to find a proof of $\vdash A$. A very naive strategy is to simply proceed bottom-up, viewing the rules of the sequent calculus as goal transformers. By applying right rules and left rules nondeterministically, eventually either there are no more rules to try (in which case the search fails), or there are no more goals to prove (in which case the search succeeds with a proof).

The problem with this naive approach, though, is that it is far too nondeterministic to be practical. Thus Andreoli devised a more refined strategy which exploits the natural division of the connectives in two phases he termed *inversion* and *focusing*:

1. During the inversion phase, whenever an asynchronous connective appears on the right, or a synchronous connective on the left, it is decomposed eagerly, by applying the corresponding invertible rule.

2. Eventually, the inversion phase ends when we arrive at goal sequents containing no more invertible connectives. The next step is to enter the focusing phase by picking *one* formula in the sequent, and decomposing that formula and its subformulas by applying (non-invertible) rules until we get back to an invertible subformula.

Before making this more precise, let me illustrate with an example:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{p \vdash [p]}\quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{r \vdash [r]}\quad \dfrac{\dfrac{\dfrac{}{s \vdash [s]}}{s \vdash [s \oplus p]}}{s \vdash s \oplus p}}{[s] \vdash s \oplus p}}{r[r \multimap s] \vdash s \oplus p}}{r, r \multimap s \vdash s \oplus p}}{r \otimes (r \multimap s) \vdash s \oplus p}}{p \vdash [p \oplus q]}\quad \dfrac{}{[r \otimes (r \multimap s)] \vdash s \oplus p}}{[(p \oplus q) \multimap (r \otimes (r \multimap s))]p \vdash s \oplus p}}{(p \oplus q) \multimap (r \otimes (r \multimap s)), p \vdash s \oplus p}}{p \vdash ((p \oplus q) \multimap (r \otimes (r \multimap s))) \multimap (s \oplus p)}}{\vdash p \multimap ((q \oplus r) \multimap (r \otimes (r \multimap s))) \multimap (s \oplus p)}$$

The important and at first seemingly miraculous fact is that this strategy is *complete*: if a sequent is derivable, then it has a proof alternating inversion and focusing phases.

In order to formalize exactly what this means, I'm going to use a slight variation on Andreoli's original one-sided formulation of focusing proofs. Besides our treatment of atomic formulas which is a slight technical simplification, the more important conceptual point is that rather than speaking of synchronous and asynchronous *connectives* as Andreoli originally did, we're going to follow Girard and make a syntactic distinction between positive and negative *formulas*:

$$P, Q ::= P \otimes Q \mid P \oplus Q \mid \downarrow N \mid p$$
$$N, M ::= N \parr M \mid N \mathbin{\&} M \mid \uparrow P \mid p^\perp$$

The so-called *shift connectives* $\downarrow$ and $\uparrow$ act as coercions between positive and negative formulas, which otherwise look exactly like ordinary formulas of linear logic. Note that the shifts are dual in the standard sense:

$$(\uparrow P)^\perp = \downarrow (P^\perp)$$
$$(\downarrow N)^\perp = \uparrow (N^\perp)$$

Now, let Γ stand for a list of negative formulas N, and Δ stand for a list of positive formulas P. We consider two kinds of sequents:

$$\vdash \Gamma; \Delta \qquad \textit{inverting}$$
$$\vdash \Delta[P] \qquad \textit{focused}$$

The system is defined as follows:[1]

[1] The minor technical simplification of Andreoli's system is the atomic inversion rule, which inverts a negative atom $p^\perp$ by postulating a focused derivation of p parameteric in a positive context δ. Note that a similar treatment is given by [Reed and Pfenning (2010)]. Thanks are due to Alexis Saurin for noticing a bug in my original treatment. (The motivation for atomic inversion should become clear below, after we discuss the general inversion principle.)

Multiplicatives:

$$\frac{\vdash N, M, \Gamma; \Delta}{\vdash N \parr M, \Gamma; \Delta} \qquad \frac{\vdash \Delta_1[P] \quad \vdash \Delta_2[Q]}{\vdash \Delta_1, \Delta_2[P \otimes Q]}$$

Additives:

$$\frac{\vdash N, \Gamma; \Delta \quad \vdash M, \Gamma; \Delta}{\vdash N \mathbin{\&} M, \Gamma; \Delta} \qquad \frac{\vdash \Delta[P]}{\vdash \Delta[P \oplus Q]} \quad \frac{\vdash \Delta[Q]}{\vdash \Delta[P \oplus Q]}$$

Shifts:

$$\frac{\vdash \Gamma; P, \Delta}{\vdash \uparrow P, \Gamma; \Delta} \qquad \frac{\vdash N; \Delta}{\vdash \Delta[\downarrow N]}$$

Focalization and atomic inversion:

$$\frac{\vdash \Delta[P]}{\vdash \Delta, P} \qquad \frac{\begin{array}{c}[\vdash \delta[p]] \\ \vdash \Gamma; \delta, \Delta\end{array}}{\vdash p^\perp, \Gamma; \Delta}$$

Let $|-|$ be the operation which converts a polarized formula to a standard formula of LL, by erasing shifts. For example, we have

$$|p \otimes \downarrow(q^\perp \parr \uparrow q)| = |\uparrow(\downarrow\uparrow p \otimes \downarrow(\uparrow\downarrow q^\perp \parr \uparrow q))| = p \otimes (q^\perp \parr q)$$

and so on. The correctness of Andreoli's strategy is implied by the following result, also called the *focalization theorem*:

$$\vdash |\Gamma|, |\Delta| \text{ iff } \vdash \Gamma; \Delta$$

Taking a step back from the proof-search interpretation, I want to consider this result in more abstract terms.

The first thing I want to explain is the connection between inversion phases and inversion principles. One way to prove the focalization theorem is to first prove the admissibility of cut and identity for the focalized sequent calculus,

$$\frac{\vdash \Gamma_1, P^\perp; \Delta_1 \quad \vdash \Gamma_2; P, \Delta_2}{\vdash \Gamma_1, \Gamma_2; \Delta_1, \Delta_2} \qquad \frac{}{\vdash P^\perp; P}$$

Now, in ordinary sequent calculus, one can derive identities for arbitrary formulas from atomic axioms, by a simple inductive argument. For example, the derivation

$$\frac{\dfrac{\vdash P^\perp, P}{\vdash P^\perp, P \oplus Q} \quad \dfrac{\vdash Q^\perp, Q}{\vdash Q^\perp, P \oplus Q}}{\vdash P^\perp \mathbin{\&} Q^\perp, P \oplus Q}$$

shows how to build an identity for $P \oplus Q$ from the identities for P and Q. The difficulty in repeating this argument for a focalized sequent calculus is that

the inversion and focusing phases can't be interleaved—for example, the above derivation violates Andreoli's protocol by applying a right rule on $P \oplus Q$ while there is still an invertible formula ($P^\perp$ or $Q^\perp$) remaining in the sequent.

In order to prove both the identity and cut theorems, it is helpful to prove the following intermediate result, which could be called the *inversion principle*:

$$\vdash P_n^\perp, \ldots, P_1^\perp; \Delta \text{ iff } \vdash \Delta_1[P_1], \ldots, \vdash \Delta_n[P_n] \text{ implies } \vdash \Delta_1, \ldots, \Delta_n, \Delta$$

I recommend that you try to prove the backwards direction of the inversion principle as an exercise, and then derive identity axioms as a corollary.

If we specialize to the case where $n = 1$, the inversion principle reads as

$$\vdash P^\perp; \Delta \text{ iff } \vdash \Delta'[P] \text{ implies } \vdash \Delta', \Delta$$

and if we rewrite this correspondence in two-sided notation, we have two equivalent formulations:

$$P \vdash \Delta \text{ iff } \Gamma \vdash [P] \text{ implies } \Gamma \vdash \Delta$$
$$\Gamma \vdash N \text{ iff } [N] \vdash \Delta \text{ implies } \Gamma \vdash \Delta$$

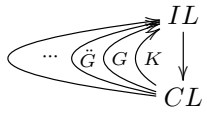
But these are just Dummett's inversion principle. Adopting his terminology, we could say that positive formulas have a "verificationist meaning-theory", as

> *Δ is a justified inference from P just in case Δ can already be derived from the premises of a canonical proof of P.*

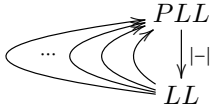
Likewise, negative formulas have a "pragmatist meaning-theory", as

> *N is a justified inference from Γ just in case any canonical consequence of N can already be derived from Γ.*

The second thing I want to point out is that formally, the focalization theorem has a very similar structure to the correctness theorems for the classical double-negation translations. Recall that I suggested you should think of the double-negation translations like this:



Well, I also want you to think of the focalization theorem like so:



where the arrows going from bottom to top represent arbitrary *polarizations* of the formulas by insertion of shifts. The point is that just as intuitionistic logic may be interpreted as a *refinement* of classical logic, polarized linear logic may be interpreted as a refinement of classical linear logic.

In fact this is really more than an analogy, in the sense that polarities can be seen as a way of *deconstructing* the classical double-negation translations.

6 Pattern-matching and continuation-passing

Putting philosophy aside, I want to try to hint at what all this has to do with programming. I'm going to give names to the different forms of proof in the focalized sequent calculus, adopted from the theory of programming languages. A proof of a focused sequent

$$\vdash \Delta[P]$$

is called a *value*, while a proof of an inverting sequent

$$\vdash \Gamma; \Delta$$

is called a *program*. I'll use the letters v and e respectively

$$\overset{v}{\vdash \Delta[P]} \qquad \overset{e}{\vdash \Gamma; \Delta}$$

to range over these different classes of proofs. I'll also overload the letter v to *label* the assumptions in Γ, which are called *value variables*, whereas I'll use the letter k to label the assumptions in Δ, which are called *continuation variables*. For example, writing

$$\overset{e}{\vdash \underset{v_1}{N_1}, \underset{v_2}{N_2}; \underset{k_1}{P_1}, \underset{k_2}{P_2}, \underset{k_3}{P_3}}$$

indicates that the program e may refer to the value variables v_1 and v_2 and the continuation variables k_1, k_2, and k_3.

The focalization and atomic inversion rules can be seen as *the* basic principles for building programs: either by plugging a value with a continuation variable, or by abstracting in a value variable:

$$\frac{\overset{v}{\vdash \Delta[P]}}{\underset{\langle v|k\rangle}{\vdash \Delta, \underset{k}{P}}} \qquad \frac{[\overset{v}{\vdash \delta[p]}] \quad \overset{e}{\vdash \Gamma; \delta, \Delta}}{\underset{v.e}{\vdash \underset{v}{p^\perp}, \Gamma; \Delta}}$$

Likewise, the rules of the focusing phase can be interpreted as different rules for building up values. For example, we can annotate the $\otimes$ and $\oplus$ rules like so:

$$\frac{\overset{v_1}{\vdash \Delta_1[P]} \quad \overset{v_2}{\vdash \Delta_2[Q]}}{\underset{(v_1,v_2)}{\vdash \Delta_1, \Delta_2[P \otimes Q]}} \qquad \frac{\overset{v}{\vdash \Delta[P]}}{\underset{\text{inl } v}{\vdash \Delta[P \oplus Q]}} \quad \frac{\overset{v}{\vdash \Delta[Q]}}{\underset{\text{inr } v}{\vdash \Delta[P \oplus Q]}}$$

The focusing phase ends at a polarity shift, which says that a continuation can be represented as a value:

$$\dfrac{\vdash \underset{v}{P^\perp}; \Delta}{\underset{\vdash \Delta[\downarrow P^\perp]}{\downarrow(v.e)}}$$

Finally, the rules of the inversion phase give different rules for building programs by decomposing values:

$$\dfrac{\vdash \underset{v_1}{N}, \underset{v_2}{M}, \Gamma; \Delta}{\underset{\vdash \underset{v}{N \parr M}, \Gamma; \Delta}{\langle v | (v_1, v_2).e \rangle}} \qquad \dfrac{\vdash \underset{v_1}{N}, \Gamma; \Delta \quad \vdash \underset{v_2}{M}, \Gamma; \Delta}{\underset{\vdash \underset{v}{N \mathbin{\&} M}, \Gamma; \Delta}{\langle v | \{\mathrm{inl}\ v_1.e_1, \mathrm{inr}\ v_2.e_2\} \rangle}} \qquad \dfrac{\vdash \Gamma; \underset{k}{P}, \Delta}{\underset{\vdash \underset{v}{\uparrow P}, \Gamma; \Delta}{\langle v | \downarrow k.e \rangle}}$$

Now, taking back a step, it's not difficult to see that any value is going to be constructed as a tree of various operations, with a given *fringe* of continuations. For example, the two derivations

$$\dfrac{\dfrac{\vdash \underset{v_1}{P^\perp}; \Delta_1}{\underset{\vdash \Delta_1[\downarrow P^\perp]}{\downarrow(v_1.e_1)}} \quad \dfrac{\vdash \underset{v_2}{R^\perp}; \Delta_2}{\underset{\vdash \Delta_2[\downarrow R^\perp]}{\downarrow(v_2.e_2)}}}{\underset{\vdash \Delta_1, \Delta_2[\downarrow P^\perp \otimes (\downarrow Q^\perp \oplus \downarrow R^\perp)]}{(\downarrow(v_1.e_1), \mathrm{inr}\downarrow(v_2.e_2))}}$$

and

$$\dfrac{\dfrac{\dfrac{\vdash \underset{v_1}{P^\perp}; \Delta_1}{\underset{\vdash \Delta_1[\downarrow P^\perp]}{\downarrow(v_1.e_1)}} \quad \dfrac{\vdash \underset{v_2}{R^\perp}; \Delta_2}{\underset{\vdash \Delta_2[\downarrow R^\perp]}{\downarrow(v_2.e_2)}}}{\underset{\vdash \Delta_1, \Delta_2[\downarrow P^\perp \otimes \downarrow R^\perp]}{(\downarrow(v_1.e_1), \downarrow(v_2.e_2))}}}{\underset{\vdash \Delta_1, \Delta_2[(\downarrow P^\perp \otimes \downarrow Q^\perp) \oplus (\downarrow P^\perp \otimes \downarrow R^\perp)]}{\mathrm{inr}(\downarrow(v_1.e_1), \downarrow(v_2.e_2))}}$$

both have the same fringe, differing only in the surrounding trees of operations. Let's write these two derivations more concisely:

$$v = (\downarrow(v_1.e_1), \mathrm{inr} \downarrow(v_2.e_2))$$
$$v' = \mathrm{inr}(\downarrow(v_1.e_1), \downarrow(v_2.k_2))$$

We call the surrounding trees with holes for continuations *patterns*,

$$p = (\downarrow -, \mathrm{inr} \downarrow -)$$
$$p' = \mathrm{inr}(\downarrow -, \downarrow -)$$

while we call the fringe of continuations a *substitution*:

$$\theta = \downarrow(v_1.e_1), \downarrow(v_2.e_2)$$

(Formally, note that patterns are just special kinds of values, with a fringe of identity continuations.) What we have just said is that any value can be *factored* as a pattern composed with a substitution, i.e.,

$$v = p\theta$$
$$v' = p'\theta$$

But by the inversion principle, this in turn means that in order to define a continuation, it suffices to define its action on patterns. In other words, we have a justification for *pattern-matching* notation.

As one last mental experiment, let's go back to our little exercise in arithmetic. Suppose we are given some suitable definition of the natural numbers as an atomic type *nat*. In particular, assume that we have the ability to add and multiply values of type *nat*, to divide values on the condition that the second value is non-zero, and to test if a value is non-zero. Can you use these assumptions to fill in the following programs?

$$\vdash \uparrow\downarrow nat^{\perp} \underset{v}{\overset{add}{\otimes}} \uparrow\downarrow nat^{\perp}; \underset{k}{nat} \qquad \vdash \uparrow\downarrow nat^{\perp} \underset{v}{\overset{mul}{\otimes}} \uparrow\downarrow nat^{\perp}; \underset{k}{nat} \qquad \vdash \uparrow\downarrow nat^{\perp} \underset{v}{\overset{div}{\otimes}} \uparrow\downarrow nat^{\perp}; \underset{k}{nat}$$

Hint: to write *div*, you might have to cheat by calling a "daimon".

Selected bibliography

[Andreoli (1991)] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3):297–347, 1992.

[Dummett (1991)] Michael Dummett. *The Logical Basis of Metaphysics.* Harvard University Press, 1991. (Note this is an expanded version of the William James Lectures, 1976.)

[Girard (1991)] Jean-Yves Girard. A new constructive logic: Classical logic. *Mathematical Structures in Computer Science*, 1:255–296, 1991.

[Girard (2001)] Jean-Yves Girard. Locus solum: From the rules of logic to the logic of rules. *Mathematical Structures in Computer Science*, 11(3):301–506, 2001.

[Kolmogorov (1925)] Andrei Nikolaevich Kolmogorov. On the principle of the excluded middle. *Matematicheskii Sbornik*, 32:646–667, 1925. English translation in *From Frege to Godel: A Source Book in Mathematical Logic*, 1879–1931, Jean van Heijenoort (ed.).

[Prawitz (1974)] Dag Prawitz. On the idea of a general proof theory. *Synthese*, 27:63–77, 1974.

[Reed and Pfenning (2010)] Jason Reed and Frank Pfenning. Focus-preserving embeddings of substructural logics in intuitionistic logic. Unpublished manuscript, 2010.

[Reynolds (1972)] John C. Reynolds. Definitional interpreters for higher order programming languages. In *ACM 72: Proceedings of the ACM annual conference*, 717–740, 1972.

[Reynolds (1993)] John C. Reynolds. The Discoveries of Continuations. *Lisp and Symbolic Computation*, 6:3/4, 233–247, 1993.