$\lambda\mu$: RELATING CONSTRUCTIVE & CLASSICAL LOGICS (*EXTENDED* HANDOUT)

Greg Restall ∗ *Arché, Philosophy Department, University of St Andrews*[*]

PROOFS, RULES AND MEANINGS ∗ ARCHÉ, ST ANDREWS ∗ 11 APRIL 2024

This talk is a reflection on the relationship between constructive and classical proof, and on the significance of Michel Parigot's $\lambda\mu$ calculus [14–16]. I attempt to understand the relationship between constructive and classical proof as a distinction orthogonal to the structural rules of *contraction* and *weakening*—since relevant, affine and linear logic also have constructive and classical variants—since looking at this difference from different perspectives may prove profitable. Along the way, the talk brings together three different interests of mine—*substructural logics* [18], *logical pluralism* [2], and *the philosophy of proof theory* [20].

1. TWO RULES / FOUR LOGICS

Gentzen/Prawitz-style natural deduction proofs for implication are very simple [7,17]. Atomic proofs are individual formulas, and there are two rules for constructing new proofs from old:

$$A \qquad \dfrac{\begin{matrix}[A]^i\\ \Pi \\ B\end{matrix}}{A \to B}{\scriptstyle \to I^i} \qquad \dfrac{\begin{matrix}\Pi \\ A \to B\end{matrix} \quad \begin{matrix}\Pi' \\ A\end{matrix}}{B}{\scriptstyle \to E}$$

Here is a proof, using these rules:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[p \to (q \to r)]^3 \quad [p]^1}{q \to r}{\scriptstyle \to E} \quad \dfrac{[p \to q]^2 \quad [p]^1}{q}{\scriptstyle \to E}}{r}{\scriptstyle \to E}}{p \to r}{\scriptstyle \to I^1}}{(p \to q) \to (p \to r)}{\scriptstyle \to I^2}}{(p \to (q \to r)) \to ((p \to q) \to (p \to r))}{\scriptstyle \to I^3}$$

Its conclusion stands at the root of the tree, and and there are no undischarged assumptions.

It is very natural to annotate proofs with λ *terms*, like so [1, 5, 9]:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[x : p \to (q \to r)] \quad [z : p]}{(xz) : q \to r}{\scriptstyle \to E} \quad \dfrac{[y : p \to q] \quad [z : p]}{(xy) : q}{\scriptstyle \to E}}{((xz)(xy)) : r}{\scriptstyle \to E}}{\lambda z((xz)(xy)) : p \to r}{\scriptstyle \to I^z}}{\lambda y\lambda z((xz)(xy)) : (p \to q) \to (p \to r)}{\scriptstyle \to I^y}}{\lambda x\lambda y\lambda z((xz)(xy)) : (p \to (q \to r)) \to ((p \to q) \to (p \to r))}{\scriptstyle \to I^x}$$

The rules for term annotation are straightforward:[1]

$$x : A \qquad \dfrac{\begin{matrix}[x : A]\\ \vdots \\ M : B\end{matrix}}{\lambda x M : A \to B}{\scriptstyle \to I^x} \qquad \dfrac{\begin{matrix}\vdots \\ M : A \to B\end{matrix} \quad \begin{matrix}\vdots \\ N : A\end{matrix}}{(M\,N) : B}{\scriptstyle \to E}$$

You can think of these *terms* as representing processes of *justification* or of *construction*. (Justify $A \to B$ by taking A as *given*, and using this to justify B. You can *use* a such a justification of $A \to B$ by *applying* it to a justification of A to produce a justification of B.)

Thinking of terms as representing processes motivates the following *reduction* rule:

$$\dfrac{\dfrac{\begin{matrix}[x : A]\\ \vdots \\ M : B\end{matrix}}{\lambda x M : A \to B}{\scriptstyle \to I^x} \quad \begin{matrix}\vdots \\ N : A\end{matrix}}{(\lambda x M\,N) : B}{\scriptstyle \to E} \quad \triangleright \quad \begin{matrix}\vdots \\ N : A \\ \vdots \\ M\{{}^N\!/_x\} : B\end{matrix}$$

Since the justification of $A \to B$ is a construction which converts a *hypothetical* justification of A into a justification for B, when we apply that to some *given* justification for A the result should be that original construction applied to the given justification.

This setting seems simple and natural, but the choice of how assumption discharge works (and equivalently, variable binding in λ terms), hides some design choices. For one thing, is *this* proof an instance of the rules?

$$\dfrac{\dfrac{[x : p]}{\lambda y x : q \to p}{\scriptstyle \to I^y}}{\lambda x\lambda y x : p \to (q \to p)}{\scriptstyle \to I^x}$$

We never *used* the supposition of q in the justification of p, and this is reflected in the term structure: the λy binds *vacuously*. There is no free y in x to bind. There is no sense in which the p has been derived *from* q. So, we have a choice: we can *allow* vacuous binding (the standard approach), or *forbid* it (in favour of some kind of *relevant* implication). To allow vacuous binding is to admit the standard structural rule of *weakening* (also called *thinning*).

DIGRESSION: You must restrict or modify the usual rules for conjunction if you want to forbid vacuous binding, since with $\mathsf{fst}\langle M, y\rangle$ you can mimic the use of an assumption y in the otherwise y-free M.

$$\dfrac{\dfrac{\dfrac{\dfrac{[x : p] \quad [y : q]}{\langle x, y\rangle : p \wedge q}{\scriptstyle \wedge I}}{\mathsf{fst}\langle x, y\rangle : p}{\scriptstyle \wedge E}}{\lambda y\,\mathsf{fst}\langle x, y\rangle : q \to p}{\scriptstyle \to I^y}}{\lambda x\lambda y\,\mathsf{fst}\langle x, y\rangle : p \to (q \to p)}{\scriptstyle \to I^x}$$

[*]Thanks to my Logic Lunch colleagues in Arché for listening to me talk about this material, and asking perceptive questions. ¶ This handout and the accompanying slides can be downloaded from `https://consequently.org/p/2024/lma`.

[1]The formal treatment of the identity of terms is subtle, and the details will not matter much here, except for one point in Section 3. To make the argument there simple, we identify terms by α-equivalence: For our purposes, $\lambda x\lambda y(yx)$ is the same term as $\lambda y\lambda z(zy)$. For a proof theorist this amounts to saying that the identity of the tags used to label discharge classes is irrelevant.

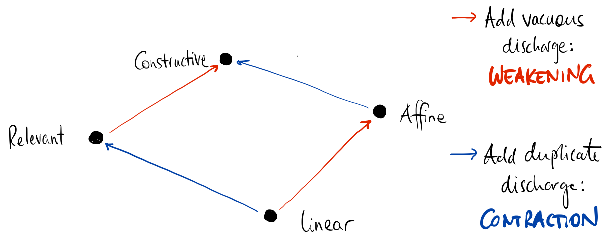
We will not spend any time on conjunction rules, but the fix is well understood in substructural logics. The standard natural deduction rule $\wedge E$ belongs to *additive* conjunction, while the $\wedge I$ rule is *multiplicative*, and these must be teased apart in a setting where we do without some of the usual structural rules [18, 21]. END DIGRESSION

Another design choice in the logic of implication involves whether multiple occurrences of an assumption can be discharged in one go. In the example proof above, with term

$$\lambda x\lambda y\lambda z((xz)(yz)) : (p \to (q \to r) \to ((p \to q) \to (p \to r))$$

two copies of the hypothesis p are discharged at once. (The λz binds two instances of z in $(xz)(yz)$.) If we think of the grounds for a judgement as resources which may expire after use, then we may have reason to restrict or to outright ban such *duplicate* discharge. To do so is to reject the structural rule of *contraction*.

Our two rules give rise to four different logics, once we make our choices concerning *weakening* and *contraction*.



We *keep* the $\to I$ and $\to E$ rules fixed and change the *context* in which those rules apply.[2]

The strongest of our four logics is *constructive* (minimal, implicational) logic. Can we extend this elegant analysis to *classical* logic? In the sequent calculus, we can. Here are the intuitionistic rules for the conditional, written sequent-style.

$$\frac{X \succ A \quad B, X' \succ C}{X, A \to B, X' \succ C}\,{\to}L \qquad \frac{X, A \succ B}{X \succ A \to B}\,{\to}R$$

The classical rules are found by allowing the sequent context to be more general, allowing for multiple formulas on the right:

$$\frac{X \succ A, Y \quad B, X' \succ Y'}{X, A \to B, X' \succ Y, Y'}\,{\to}L' \qquad \frac{X, A \succ B, Y}{X \succ A \to B, Y}\,{\to}R'$$

These can be seen as different incarnations of the one basic structure for $\to$:

$$\frac{\succ A \quad B \succ}{A \to B \succ}\,{\to}L_{core} \qquad \frac{A \succ B}{\succ A \to B}\,{\to}R_{core}$$

The core rules are applied in a given structural context by requiring the surrounding sequent to be as *general* as the given structural context allows. Notice that the choice of shape of sequent, between $X \succ A$ and $X \succ Y$ is orthogonal to the question of whether contraction or weakening are allowed.

My aim is to translate this result into a setting where sequents are not treated as a primitive notion, but arise out of the structural context of natural deduction proofs. Along the way, I hope to see how we can extend the simply-typed λ calculus and our understanding of processes of justification or construction to apply in this classical setting, independently of our choices concerning contraction and weakening.

2. ALTERNATIVES

The currently popular *philosophical* treatment of classical reasoning in natural setting is *bilateralist* [10–12, 19, 22, 23]. If we treat assertion and denial as equal partners, we can recover the full power of classical reasoning. One way to do this is to replace the familiar Gentzen/Prawitz–style natural deduction rules with new rules involving positively tagged formulas $(+A)$ and negatively tagged formulas $(-A)$. For example, here are Rumfitt's rules for conditionals, both positively and negatively tagged [22]:

802 Ian Rumfitt

+-→-I:

$$\frac{\begin{matrix}\overline{}\,(i)\\ +A \\ \vdots \\ +B\end{matrix}}{+(A \to B)}\,(i)$$

+-→-E:

$$\frac{+(A \to B) \quad +A}{+B}$$

–-→-I:

$$\frac{+A \quad -B}{-(A \to B)}$$

–-→-E:

$$\frac{-(A \to B)}{+A} \qquad \frac{-(A \to B)}{-B}$$

This sort of set-up has its virtues, but it is in no way a simple structural variation on the orginal rules. We have not kept the connective rules constant and applied them in a wider structural context. However, it is possible to use bilateralist motivations for a purely structural expansion of Gentzen/Prawitz natural deduction, and to do so in a way that is totally orthogonal to the structural rules of weakening and contraction.

The key bilateralist idea is that a classical sequent $A, B \succ C, D$ does not carry quite enough structure to represent the upshot of a *proof*. For that, we need to decide which formula is the *conclusion*. So mark a conclusion with a box like so: $A, B \succ \boxed{C}, D$. If we take C to be the *conclusion* of the proof, then the remaining formula D is part of the proof context, alongside A and B, but with *opposite polarity* to A and B. This can be read as

Given A *and* B, C *follows, unless* D.

Granting A *and* B, *and setting* D *aside, we have* C.

This reading motivates two simple structural rules, governing how formulas may be *set aside* (treated as *alternatives*, or, for short, *stored*) and later *retrieved*. These rules allow us to shift focus from one formula to another. A natural way to do so is to allow the focus to shift away from a formula entirely, as it transitions from one formula to another. The $\uparrow$ rule allows what *was* a conclusion to be set aside. The result is, in a natural deduction proof, a *dead end*, a proof with no conclusion. After all, if you *prove* A only to set it aside, there is no remaining alternative:

$$\frac{\begin{matrix}\Pi\\ A\end{matrix} \quad \cancel{A}}{\sharp}\,\uparrow \qquad \frac{X \succ \boxed{A}, Y}{X \succ \boxed{}\, A, Y}$$

[2] "Context" here is Nuel Belnap's *antecedently given context of deducibility* [3, p. 131]. Think of the natural deduction rules as added to a pre-existing notion of consequence or structure of justification: Does that pre-existing context satisfy the rule of weakening? (Whenever A follows from X then A follows from X, B too.) Does it satisfy contraction? (Whenever A follows from X, B, B it follows from X, B too.) *That* is what is at stake when you consider whether the $\to I/E$ rules should allow for contraction and for weakening.

To the left of the natural deduction proof we have the corresponding sequent rule. $X \succ \boxed{A}, Y$ represents the context for the proof Π to conclusion A where X collects together everything we have assumed and Y consists of everything we have set aside. So, when we also set A aside (adding $\cancel{A}$ as an assumption) and leaving no formula as the conclusion, which we represent with $\sharp$, then the sequent representing this adds A to the collection of set-aside formulas, so the negative context is now A, Y, and no formula is in focus: the box is empty. So, we extend the natural deduction proof syntax with two new structural features: slashed formulas as new *leaves* in proof trees, and $\sharp$ to represent dead-end proofs.

Once we have reached a dead end, we need to be able to do something to back out of it, if we wish to prove anything at all. But this is obvious: if we reach a dead end, we can take one of the claims we previously set aside, and retrieve it as our conclusion:

$$\dfrac{\begin{matrix}[\cancel{A}]^i\\ \Pi\\ \sharp\end{matrix}}{A}\downarrow^i \qquad\qquad \dfrac{X \succ \Box\, A, Y}{X \succ \boxed{A}, Y}$$

These rules are purely structural, and they are independent of whether we allow or forbid contraction or weakening. If we add them to any of our four calculi, we get a *classical* proof system, corresponding to the original constructive system. Here is a proof of the constructively underivable Peirce's Law using the original natural deduction rules applied in this wider structural setting:

$$\dfrac{\dfrac{\dfrac{\dfrac{[(p\to q)\to p]^3 \quad \dfrac{\dfrac{\dfrac{[p]^1 \quad [\cancel{p}]^2}{\sharp}\uparrow}{q}\downarrow}{p\to q}{\to}I^1}{p}{\to}E \quad [\cancel{p}]^2}{\sharp}\uparrow}{p}\downarrow^2}{((p\to q)\to p)\to p}{\to}I^3$$

Notice that in this proof we use duplicate discharge of *alternatives* (at the $\downarrow^2$ step) and we used a *vacuous* retrieval at the $\downarrow$ step where we inferred q.[3] This proof of Peirce's Law uses both contraction and weakening of alternatives.

It is natural to extend the λ-term assignment rules to this classical natural deduction setting. To do so, we add a family of *labels* for each stored formula, and to annotate a dead end we need a *term* of type A and a label of corresponding type $\cancel{A}$. We'll call such a pair $\langle M|\alpha\rangle$ a *package*. Given package P, and a label α of type $\cancel{A}$, we mark retrieving A from P with the term $\mu\alpha P$. This binds the free occurrences of the label α in P. These rules were originally formulated by Michel Parigot [14–16]. The rules are:

$$\dfrac{\begin{matrix}\vdots\\ M:A\end{matrix} \quad \alpha:\cancel{A}}{\langle M|\alpha\rangle:\sharp}\uparrow \qquad\qquad \dfrac{\begin{matrix}[\alpha:\cancel{A}]\\ \vdots\\ P:\sharp\end{matrix}}{\mu\alpha P:A}\downarrow^\alpha$$

[3]The rather arbitrary 'falsity elimination' rule—according to which you can infer anything you like from a contradiction—has been traditionally understood as the ground of the irrelevant deduction from p and $\neg p$ to q, independent of the irrelevant deduction from p to $q \to p$, which uses vacuous discharge of assumptions. In the presence of alternatives, we can understand them as two sides of the one coin. In one case, it is an *assumption* that is vacuously discharged, in the other, it is an *alternative* that is vacuously retrieved [21].

Here the proof of Peirce's Law, annotated with $\lambda\mu$ terms.

$$\dfrac{\dfrac{\dfrac{\dfrac{[x:(p\to q)\to p] \quad \dfrac{\dfrac{\dfrac{[y:p]\quad[\alpha:\cancel{p}]}{\langle y|\alpha\rangle:\sharp}\uparrow}{\mu\beta\langle y|\alpha\rangle:q}\downarrow^\beta}{\lambda y\mu\beta\langle y|\alpha\rangle : p\to q}{\to}I^y}{(x\lambda y\mu\beta\langle y|\alpha\rangle):p}{\to}E \quad [\alpha:\cancel{p}]}{\langle(x\lambda y\mu\beta\langle y|\alpha\rangle)|\alpha\rangle:\sharp}\uparrow}{\mu\alpha\langle(x\lambda y\mu\beta\langle y|\alpha\rangle)|\alpha\rangle:p}\downarrow^\alpha}{\lambda x\mu\alpha\langle(x\lambda y\mu\beta\langle y|\alpha\rangle)|\alpha\rangle : ((p\to q)\to p)\to p}{\to}I^x$$

Given the dead-end conclusion $\sharp$, it makes sense to *use* it to define *negation*. First, define f as the formula representative of $\sharp$, and then use $\neg A$ as an abbreviation for $A \to f$.[4]

$$\dfrac{P:\sharp}{\mu P:f}fI \qquad\qquad \dfrac{M:f}{\langle M\rangle:\sharp}fE$$

Here, we treat f as the formula-representative of the dead-end. We do not need to retrieve any stored alternatives to derive f, and because f is available 'for free' at any dead-end, we do not need to label any f, once derived, in order to reach a dead-end. In fact, in our term-assignment system we do not have any labels of type f, as none are needed.

Here is a proof of A from $(A \to f) \to f$, using these rules:

$$\dfrac{\dfrac{\dfrac{y:(A\to f)\to f \quad \dfrac{\dfrac{\dfrac{[x:A]\quad[\alpha:\cancel{A}]}{\langle x|\alpha\rangle:\sharp}\uparrow}{\mu\langle x|\alpha\rangle : f}fI}{\lambda x\mu\langle x|\alpha\rangle : A\to f}{\to}I^x}{(y\lambda x\mu\langle x|\alpha\rangle):f}{\to}E}{\langle(y\lambda x\mu\langle x|\alpha\rangle)\rangle:\sharp}fE}{\mu\alpha\langle(y\lambda x\mu\langle x|\alpha\rangle)\rangle : A}\downarrow^\alpha$$

Taking $\neg A$ as shorthand for $A \to f$, these are the *I/E* rules for negation:

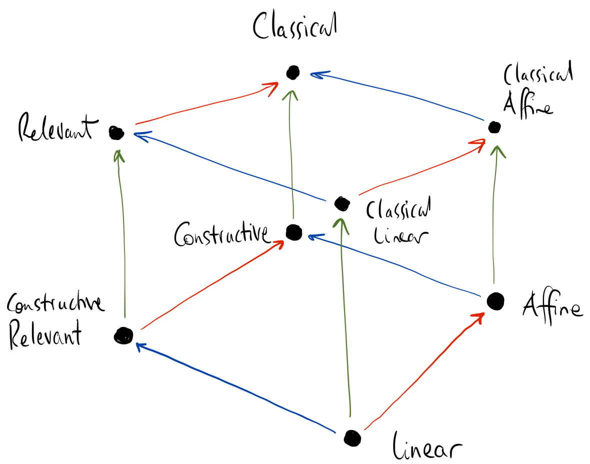
$$\dfrac{M:\neg A \quad N:A}{\langle(MN)\rangle:\sharp}\neg E \qquad\qquad \dfrac{\begin{matrix}[x:A]\\ \vdots\\ P:\sharp\end{matrix}}{\lambda x\mu P:\neg A}\neg I^x$$

The proof above simplifies to this (linear) proof, from $\neg\neg A$ to A.

$$\dfrac{\dfrac{y:\neg\neg A \quad \dfrac{\dfrac{[x:A]\quad[\alpha:\cancel{A}]}{\langle x|\alpha\rangle:\sharp}\uparrow}{\lambda x\mu\langle x|\alpha\rangle:\neg A}\neg I^x}{\langle(y\lambda x\mu\langle x|\alpha\rangle)\rangle:\sharp}\neg E}{\mu\alpha\langle(y\lambda x\mu\langle x|\alpha\rangle)\rangle:A}\downarrow^\alpha$$

[4]Thinking of negation in this way allows for the order of priority to be explicit. In Gentzen/Prawitz natural deduction (whether constructive or classical), negation and the contradictory formula $\perp$ are defined in terms of each other [13]. Here, the storage and retrieval structural rules fix the behaviour of reaching a dead end in a proof. $\sharp$ is given its interpretation by the purely structural rules, and the behaviour of assertion and denial. Then, f is defined in terms of $\sharp$: to assert f is to reach a dead end, and one allowable response when reaching a dead end is to assert f. Then $\neg A$ is a shorthand for $A \to f$. The order of priority is fixed: $\sharp$, then f then negation (using implication). What makes negation *negation* is ultimately explained in terms of $\sharp$, what is left when you derive something but also set it aside.

So, with the structural rules for alternatives, and the corresponding term assignment system, we now have eight different logical systems:



Our new μ terms have their own reduction rules, like the familiar β reduction rule for λ terms. If we take a package retrieve a label α, but then repackage that term with another label β, this retrieve/store pair can be done away with in a natural way by creating the original package using β in place of α:

$$\dfrac{\dfrac{\begin{matrix}[\alpha : \cancel{A}]\\ \vdots\\ P : \sharp\end{matrix}}{\mu\alpha P : A}\,{\downarrow^{\alpha}} \qquad \beta : \cancel{A}}{\langle \mu\alpha P|\beta\rangle : \sharp}\,{\uparrow} \qquad \triangleright \qquad \begin{matrix}[\beta : \cancel{A}]\\ \vdots\\ P\{^{\beta}/_{\alpha}\} : \sharp\end{matrix}$$

The same goes for the label-free f-packages, with no relabelling:

$$\dfrac{\dfrac{\begin{matrix}\vdots\\ P : \sharp\end{matrix}}{\mu P : f}\,fI}{\langle\mu P\rangle : \sharp}\,fE \qquad \triangleright \qquad \begin{matrix}\vdots\\ P : \sharp\end{matrix}$$

Another reduction connects retrieval with *application*. We would like to know how to apply a retrieved term $\mu\alpha P$ (of type $A \to B$) to another term N (of type A). This is more complex. Consider a proof with this structure:

$$\dfrac{\dfrac{\begin{matrix}[\alpha : \cancel{A \to B}]\\ \vdots\\ P : \sharp\end{matrix}}{\mu\alpha P : A \to B}\,{\downarrow^{\alpha}} \qquad \begin{matrix}\vdots\\ N : A\end{matrix}}{(\mu\alpha P\, N) : B}\,{\to}E$$

In such a proof, the label α is applied in P some number of times, each labelling site marked with $*$:

$$\dfrac{\dfrac{\begin{matrix}\dfrac{\begin{matrix}\vdots\\ * : A \to B\end{matrix} \qquad [\alpha : \cancel{A \to B}]}{\langle *|\alpha\rangle : \sharp}\,{\uparrow}\\ \vdots\\ P : \sharp\end{matrix}}{\mu\alpha P : A \to B}\,{\downarrow^{\alpha}} \qquad \begin{matrix}\vdots\\ N : A\end{matrix}}{(\mu\alpha P\, N) : B}\,{\to}E$$

So, if we hoist the derivation of A upwards to each site at which the label α is applied in P, we have the following derivation:

$$\dfrac{\begin{matrix}\dfrac{\dfrac{\begin{matrix}\vdots\\ * : A \to B\end{matrix} \qquad \begin{matrix}\vdots\\ N : A\end{matrix}}{(*\, N) : B}\,{\to}E \qquad [\beta : \cancel{B}]}{\langle (*\, N)|\beta\rangle : \sharp}\,{\uparrow}\\ \vdots\\ P\{^{\langle(*\, N)|\beta\rangle}/_{\langle *|\alpha\rangle}\} : \sharp\end{matrix}}{\mu\beta P\{^{\langle(*\, N)|\beta\rangle}/_{\langle *|\alpha\rangle}\} : B}\,{\downarrow^{\beta}}$$

Here, the notation for substitution $P\{^{\langle(*\, N)|\beta\rangle}/_{\langle *|\alpha\rangle}\}$ is to be understood as substituting, for each package $\langle M|\alpha\rangle$ (for any term M packaged up with our given label α) inside P with the package $\langle (M\, N)|\beta\rangle$, where β is free for α in P.

If the result of the application of our labelled term to N is a term of type f, on the other hand, it does not get a fresh label, so the result of the application is simpler. The original proof simplifies as follows:

$$\dfrac{\dfrac{\begin{matrix}\dfrac{\begin{matrix}\vdots\\ * : A \to f\end{matrix} \quad [\alpha : \cancel{A \to f}]}{\langle *|\alpha\rangle : \sharp}\,{\uparrow}\\ \vdots\\ P : \sharp\end{matrix}}{\mu\alpha P : A \to f}\,{\downarrow^{\alpha}} \qquad \begin{matrix}\vdots\\ N : A\end{matrix}}{(\mu\alpha P\, N) : f}\,{\to}E \qquad \triangleright \qquad \dfrac{\begin{matrix}\dfrac{\dfrac{\begin{matrix}\vdots\\ * : A \to f\end{matrix} \quad \begin{matrix}\vdots\\ N : A\end{matrix}}{(*\, N) : f}\,{\to}E}{\langle (*\, N)\rangle : \sharp}\,fE\\ \vdots\\ P\{^{\langle(*\, N)\rangle}/_{\langle *|\alpha\rangle}\} : \sharp\end{matrix}}{\mu P\{^{\langle(*\, N)\rangle}/_{\langle *|\alpha\rangle}\} : B}\,fI$$

The reduction rules in the $\lambda\mu$ calculus are then:

$$\begin{array}{rcll}
(\lambda x M\, N) & \triangleright & M\{^{N}/_{x}\} & \\
\langle \mu\alpha P|\beta\rangle & \triangleright & P\{^{\beta}/_{\alpha}\} & \\
\langle \mu P\rangle & \triangleright & P & \\
(\mu\alpha P\, N) & \triangleright & \mu\beta P\{^{\langle(*\, N)|\beta\rangle}/_{\langle *|\alpha\rangle}\} & \text{(if not type } f\text{)}\\
 & \triangleright & \mu P\{^{\langle(*\, N)\rangle}/_{\langle *|\alpha\rangle}\} & \text{(if type } f\text{)}
\end{array}$$

Figure 1 shows a natural deduction proof from $p \to (p \to q)$ using a duplicate *retrieval* (at the $\downarrow^{\beta}$ step, marked with !!). Two copies of label β are bound in the one $\mu\beta$ term. When we evaluate this proof term using the reduction rules the duplicate μ binding is reduced to a duplicate λ binding. One reduction process goes as follows, where at each step I have $\boxed{\text{framed}}$ the subterm reduced in the next step:

$$\begin{array}{cl}
 & \lambda w\mu\gamma\langle\boxed{(\mu\beta\langle\lambda z\mu\langle(\mu\alpha\langle\lambda y\mu\langle(xy)|\alpha\rangle|\beta\rangle z)|\gamma\rangle|\beta\rangle w)}\rangle\\
\triangleright & \lambda w\mu\gamma\langle(\lambda z\mu\langle(\mu\alpha\langle\boxed{(\lambda y\mu\langle(xy)|\alpha\rangle w)}\rangle z)|\gamma\rangle w)\rangle\\
\triangleright & \lambda w\mu\gamma\langle(\lambda z\mu\langle(\mu\alpha\boxed{\langle\mu\langle(xw)|\alpha\rangle\rangle}z)|\gamma\rangle w)\rangle\\
\triangleright & \lambda w\mu\gamma\langle(\lambda z\mu\langle\boxed{(\mu\alpha\langle(xw)|\alpha\rangle z)}|\gamma\rangle w)\rangle\\
\triangleright & \lambda w\mu\gamma\langle(\lambda z\mu\boxed{\langle\mu\delta\langle((xw)z)|\delta\rangle|\gamma\rangle}w)\rangle\\
\triangleright & \lambda w\mu\gamma\langle\boxed{(\lambda z\mu\langle((xw)z)|\gamma\rangle w)}\rangle\\
\triangleright & \lambda w\mu\gamma\boxed{\langle\mu\langle((xw)w)|\gamma\rangle\rangle}\\
\triangleright & \lambda w\mu\gamma\langle((xw)w)|\gamma\rangle
\end{array}$$

The resulting term describes a much more direct proof from

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : p \to (p \to q) \qquad [y : p]}{(xy) : p \to q}{\to}E \qquad [\alpha : \cancel{p \to q}]}{\langle (xy)|\alpha\rangle : \sharp}\uparrow}{\lambda y\mu\langle (xy)|\alpha\rangle : \neg p}\neg I^{y} \qquad [\beta : \cancel{\neg p}]^{!}}{\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle : \sharp}\uparrow}{\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle : p \to q}\downarrow^{\alpha} \qquad [z : p]}{(\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle z) : q}{\to}E \qquad [\gamma : \cancel{q}]}{\langle (\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle z)|\gamma\rangle : \sharp}\uparrow}{\lambda z\mu\langle (\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle z)|\gamma\rangle : \neg p}\neg I^{z} \qquad [\beta : \cancel{\neg p}]^{!}}{\langle \lambda z\mu\langle (\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle z)|\gamma\rangle|\beta\rangle : \sharp}\uparrow}{\mu\beta\langle \lambda z\mu\langle (\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle z)|\gamma\rangle|\beta\rangle : \neg p}\downarrow^{\beta !!} \qquad [w : p]}{\langle (\mu\beta\langle \lambda z\mu\langle (\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle z)|\gamma\rangle|\beta\rangle w)\rangle : \sharp}\neg E}{\mu\gamma\langle (\mu\beta\langle \lambda z\mu\langle (\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle z)|\gamma\rangle|\beta\rangle w)\rangle : q}\downarrow^{\gamma}}{\lambda w\mu\gamma\langle (\mu\beta\langle \lambda z\mu\langle (\mu\alpha\langle \lambda y\mu\langle (xy)|\alpha\rangle|\beta\rangle z)|\gamma\rangle|\beta\rangle w)\rangle : p \to q}{\to}I^{w}$$

Figure 1: A derivation using contraction on *alternatives*

$p \to (p \to q)$ to $p \to q$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : p \to (p \to q) \qquad [w : p]}{(xw) : p \to q}{\to}E \qquad [w : p]}{((xw)w) : q}{\to}E \qquad [\gamma : \cancel{q}]}{\langle ((xw)w)|\gamma\rangle : \sharp}\uparrow}{\mu\gamma\langle ((xw)w)|\gamma\rangle : q}\downarrow^{\gamma}}{\lambda w\mu\gamma\langle ((xw)w)|\gamma\rangle : p \to q}{\to}I^{w}$$

This proof is not quite as direct as it could be. There is no *need* to take the detour through storing q only to immediately retreive it. The term is fully reduced by way of the reduction rules, in just the same way that the term $\lambda x(Mx)$ is β-reduced, even though the corresponding natural deduction proof

$$\dfrac{\dfrac{M : A \to B \qquad x : A}{(Mx) : B}{\to}E}{\lambda x(Mx) : A \to B}{\to}I^{x}$$

goes through the detour of eliminating the conditional only to reintroduce it. This motivates $\triangleright_\eta$, setting $\lambda x(Mx) \triangleright_\eta M$, $\mu\alpha\langle M|\alpha\rangle \triangleright_\eta M$ and $\mu\langle M\rangle \triangleright_\eta M$ and then this proof η-reduces to

$$\dfrac{\dfrac{\dfrac{x : p \to (p \to q) \qquad [w : p]}{(xw) : p \to q}{\to}E \qquad [w : p]}{((xw)w) : q}{\to}E}{\lambda w((xw)w) : p \to q}{\to}I^{w}$$

3. TRANSLATION / NORMALISATION

It's well known that classical logic can be found inside intuitionistic logic using any number of different double negation translations [6]. In fact, classical logic is found inside intuitionistic logic not only at the level of *provability*, but also at the level of *proofs*, and even at the level of proof dynamics—*normalisation*. These results are *robust*. They extend to all four structural settings, linear, relevant, affine, and full.

Here is one translation that embeds the classical formulas, ~~stored~~ formulas and the punctuation mark ♯ signifying a dead-end, inside a constructive (minimal, i.e., negation-free) language, and simultaneously embeds classical λμ terms (including labels and packages) inside the simply typed λ calculus. We select a fresh propositional atom q (unused in the classical language), and we define our translation mapping from the classical language to the minimal language as follows:

$$\begin{aligned}
\overline{\sharp} &= q \\
\overline{f} &= q \\
\overline{\cancel{p}} &= p \to q \\
\overline{p} &= (p \to q) \to q \\
\overline{\cancel{A \to B}} &= (\overline{A} \to \overline{B}) \to q \\
\overline{A \to B} &= ((\overline{A} \to \overline{B}) \to q) \to q
\end{aligned}$$

We abbreviate $C \to q$ as $\neg_q C$. Note that in this translation, for every classical formula A (other than f), $\overline{A} = \neg_q \overline{\cancel{A}}$.

For terms, packages and labels, given any variable x of type A we find a corresponding variable $\overline{x}$ of type $\overline{A}$, and for every label α of type $\cancel{A}$, we find a unique variable $\overline{\alpha}$ of type $\overline{\cancel{A}}$. Using this correspondence between the classical term variables and labels, we define a translation from classical terms, labels and packages to simply typed lambda terms as follows:[5]

$$\begin{aligned}
\overline{\lambda x N} &= \lambda y(y\lambda\overline{x}\,\overline{N}) \\
\overline{(M\,N)} &= \lambda z(\overline{M}\,\lambda y((y\,\overline{N})z)) \\
\overline{\langle M|\alpha\rangle} &= (\overline{M}\,\overline{\alpha}) \\
\overline{\mu\alpha P} &= \lambda\overline{\alpha}\,\overline{P} \\
\overline{\mu P} &= \overline{P} \\
\overline{\langle N\rangle} &= \overline{N}
\end{aligned}$$

This translation preserves types in the sense that if M has type A, then $\overline{M}$ has type $\overline{A}$ (and similarly for labels and packages) and furthermore, if the source terms are linear (relevant or affine), so is the translation of that term. For type preservation, we argue by induction on the construction of the term, using justifications in Figure 2 for each different term constructor.

[5] Parigot discusses a number of other translations in his paper discussing strong normalisaiton for the λμ calculus [16]. I choose this translation because it is simple, it is orthogonal to the other structural rules, and it allows for all four λμ reductions to be preserved.

$$\dfrac{\dfrac{[y:\neg_q(\overline{A}\to\overline{B})] \quad \dfrac{\begin{matrix}[\overline{x}:\overline{A}]\\ \vdots \\ \overline{N}:\overline{B}\end{matrix}}{\lambda\overline{x}\overline{N}:\overline{A}\to\overline{B}}}{(y\lambda\overline{x}\overline{N}):q}}{\lambda y(y\lambda\overline{x}\overline{N}):\neg_q\neg_q(\overline{A}\to\overline{B})} \qquad \dfrac{\dfrac{\begin{matrix}\vdots\\ \overline{M}:\neg_q\neg_q(\overline{A}\to\overline{B})\end{matrix} \quad \dfrac{\dfrac{\dfrac{[y:\overline{A}\to\overline{B}] \quad \begin{matrix}\vdots\\ \overline{N}:\overline{A}\end{matrix}}{(y\overline{N}):\overline{B}} \quad [z:\overline{\cancel{B}}]}{((y\overline{N})z):q}}{\lambda y((y\overline{N})z):\neg_q(\overline{A}\to\overline{B})}}{(\overline{M}\,\lambda y((y\overline{N})z)):q}}{\lambda z(\overline{M}\,\lambda y((y\overline{N})z)):\overline{B}}$$

$$\dfrac{\begin{matrix}\vdots\\ \overline{M}:\overline{A} \quad \overline{\alpha}:\overline{\cancel{A}}\end{matrix}}{(\overline{M}\,\overline{\alpha}):q} \qquad \dfrac{\begin{matrix}[\overline{\alpha}:\overline{\cancel{A}}]\\ \vdots\\ \overline{P}:q\end{matrix}}{\lambda\overline{\alpha}\overline{P}:\neg_q\overline{\cancel{A}}} \qquad \dfrac{\begin{matrix}\vdots\\ \overline{P}:q\end{matrix}}{\overline{\mu P}:q} \qquad \dfrac{\begin{matrix}\vdots\\ \overline{N}:q\end{matrix}}{\overline{\langle N\rangle}:q}$$

Figure 2: Translating classical inferences into (minimal) constructive inferences

Furthermore, each of the classical $\lambda\mu$ reduction steps can be translated into λ term $\beta\eta$-reductions.

For $(\lambda x M\, N) \triangleright M\{^N/_x\}$:

$$\begin{aligned}\overline{(\lambda x M\, N)} &= \lambda z(\overline{\lambda x M}\,\lambda w((w\,\overline{N})z)) \\ &= \lambda z(\lambda v(v\,\lambda\overline{x}\overline{M})\,\lambda w((w\,\overline{N})z)) \\ &\triangleright \lambda z(\lambda w((w\,\overline{N})z)\,\lambda\overline{x}\overline{M}) \\ &\triangleright \lambda z((\lambda\overline{x}\overline{M}\,\overline{N})z) \\ &\triangleright_\eta (\lambda\overline{x}\overline{M}\,\overline{N}) \\ &\triangleright \overline{M}\{^{\overline{N}}/_{\overline{x}}\} \\ &= \overline{M\{^N/_x\}}\end{aligned}$$

For $\langle\mu\alpha P|\beta\rangle \triangleright P\{^\beta/_\alpha\}$:

$$\begin{aligned}\overline{\langle\mu\alpha P|\beta\rangle} &= (\overline{\mu\alpha P}\,\overline{\beta}) = (\lambda\overline{\alpha}\overline{P}\,\overline{\beta}) \\ &\triangleright \overline{P}\{^{\overline{\beta}}/_{\overline{\alpha}}\} = \overline{P\{^\beta/_\alpha\}}\end{aligned}$$

For $\langle\mu P\rangle \triangleright P$ it suffices to note that $\overline{\langle\mu P\rangle} = \overline{P}$.

For $(\mu\alpha P\, N) \triangleright \mu\beta P\{^{\langle(*N)|\beta\rangle}/_{\langle *|\alpha\rangle}\}$:[6]

$$\begin{aligned}\overline{(\mu\alpha P\, N)} &= \lambda y(\overline{\mu\alpha P}\,\lambda x((x\,\overline{N})y)) \\ &= \lambda y(\lambda\overline{\alpha}\overline{P}\,\lambda x((x\,\overline{N})y)) \\ &\triangleright \lambda y\overline{P}\{^{\lambda x((x\,\overline{N})y)}/_{\overline{\alpha}}\} \\ &\triangleright \lambda y\overline{P}\{^{((*\,N)y)}/_{(*\,\overline{\alpha})}\} \\ &= \lambda\overline{\beta}\,\overline{P}\{^{\overline{\langle(*N)|\beta\rangle}}/_{\overline{\langle *|\alpha\rangle}}\} \\ &= \overline{\mu\beta P\{^{\langle(*N)|\beta\rangle}/_{\langle *|\alpha\rangle}\}}\end{aligned}$$

It follows that a great deal of the behaviour of classical proof is found inside constructive proof of formulas of the form $\overline{A}$.

4. MEANINGS

What does this mean for the relationship between classical and constructive reasoning? Consider this extract from Errett Bishop and Douglas Bridges' 1985 *Constructive Analysis* [4, p. 7]:

> To see how some of the most basic results of classical analysis lack computational meaning, take the assertion that every bounded non-void set A of real numbers has a least upper bound. (The real number b is the *least upper bound* of A if $a \leqq b$ for all a in A, and if there exist elements of A that are arbitrarily close to b.) To avoid unnecessary complications, we actually consider the somewhat less general assertion that every bounded sequence (x_k) of rational numbers has a least upper bound b (in the set of real numbers). If this assertion were constructively valid, we could compute b, in the sense of computing a rational number approximating b to within any desired accuracy; in fact, we could program a digital computer to compute the approximations for us. For instance, the computer could be programmed to produce, one by one, a sequence $((b_k, m_k))$ of ordered pairs, where each b_k is a rational number and each m_k is a positive integer, such that $x_j \leqq b_k + k^{-1}$ for all positive integers j and k, and $x_{m_k} \geqq b_k - k^{-1}$ for all positive integers k. Unless there exists a general method M that produces such a computer program corresponding to each bounded, constructively given sequence (x_k) of rational numbers, we are not justified, by constructive standards, in asserting that each of the se-

[6] In this derivation we rely on the identification of α-equivalent terms.

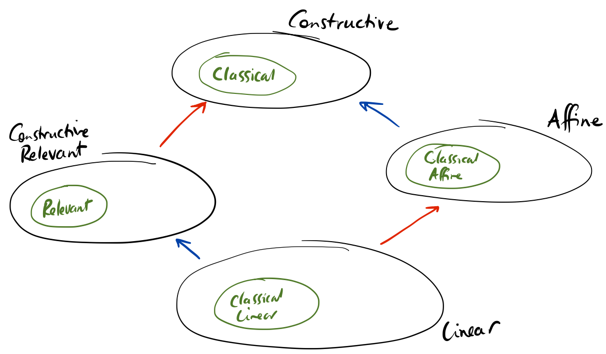
Let's call this PERSPECTIVE #1: Classical reasoning *extends* constructive reasoning. There are statements which can be *proved* classically that *cannot* be proved constructively.

Contrast that quote with this extract from Robert Harper's 2016 *Practical Foundations for Programming Languages* [8, p. 104]:

> The law of the excluded middle provides a prime example. Constructively, this principle is not universally valid, as we have seen in Exercise **12.1**. Classically, however, it is valid, because every proposition is either false or not false, and being not false is the same as being true. Nevertheless, classical logic is consistent with constructive logic in that constructive logic does not refute classical logic. As we have seen, constructive logic proves that the law of the excluded middle is positively not refuted (its double negation is constructively true). Consequently, constructive logic is stronger (more expressive) than classical logic, because it can express more distinctions (namely, between affirmation and irrefutability), and because it is consistent with classical logic.
>
> Proofs in constructive logic have computational content: they can be executed as programs, and their behavior is described by their type. Proofs in classical logic also have computational content, but in a weaker sense than in constructive logic. Rather than positively affirm a proposition, a proof in classical logic is a computation that cannot be refuted. Computationally, a refutation consists of a continuation, or control stack, that takes a proof of a proposition and derives a contradiction from it. So a proof of a proposition in classical logic is a computation that, when given a refutation of that proposition derives a contradiction, witnessing the impossibility of refuting it. In this sense, the law of the excluded middle has a proof, precisely because it is irrefutable.

Call this PERSPECTIVE #2: The constructive language *extends* the classical language. There are things we can *state* constructively that *cannot* be stated classically.

This second perspective induces a different way to relate the classical logics to their constructive counterparts. We might depict it like this:



Which of these pictures is *correct*?

I think this depends on what you *mean*, in the sense that it depends on how you individuate the very claims we make in our reasoning, those items that have meaning. We usually take this as *given*. There is one field of statements, and classical and constructive mathematicians argue about which statements in this field are correct.

> Take ***the assertion*** *that every bounded non-void set A of real numbers has a least upper bound . . .*

This fits PERSPECTIVE #1 taking classical logic to be an extension of constructive logic, allowing for more proofs.

However, if you take it that propositional content is determined by what *norms* govern it, then the usual picture is not the *only* one. Constructive justification is *stricter* than classical justification. Since there are fewer ways to give constructive justification, you can do more with such a justification when you have one.

CLASSICALLY: to state something is to rule something out, in that if you and I *rule out* the same things, we have *said* the same thing.

CONSTRUCTIVELY: p and $\neg\neg p$ *rule out* the same things, but they might (construtively) entail *different* things.

PERSPECTIVE #2A: The constructive distinction between p and $\neg\neg p$ is a meaningful difference in what is *said*. The classical logician erases or ignores differences that are present in propositional content.

PERSPECTIVE #2B: The constructive distinction between p and $\neg\neg p$ is not a difference in propositional content. If we allow only constructive justification, we are in a wider field of *pre*-propositions, only some of which are governed by all the norms that determine propositional content.

Our *formal* results are consistent with PERSPECTIVES #1, #2A and #2B. Each is an admissible perspective, consistent with our understanding of the relationship between classical and constructive proof.

THE MORAL OF THIS STORY: Take time to *recognise* these different perspectives, and *learn* what is involved in taking up each stance.

REFERENCES

[1] H. P. BARENDREGT. "Lambda Calculi with Types". In SAMSON ABRAMSKY, DOV GABBAY, AND T. S. E. MAIBAUM, editors, *Handbook of Logic in Computer Science*, volume 2, chapter 2, pages 117–309. Oxford University Press, 1992.

[2] JC BEALL AND GREG RESTALL. *Logical Pluralism*. Oxford University Press, Oxford, 2006.

[3] NUEL D. BELNAP. "Tonk, Plonk and Plink". *Analysis*, 22:130–134, 1962.

[4] ERRETT BISHOP AND DOUGLAS BRIDGES. *Constructive Analysis*. Springer-Verlag, 1985.

[5] ALONZO CHURCH. *The Calculi of Lambda-Conversion*. Number 6 in Annals of Mathematical Studies. Princeton University Press, 1941.

[6] GILDA FERREIRA AND PAULO OLIVA. "On Various Negative Translations". *Electronic Proceedings in Theoretical Computer Science*, 47:21–33, January 2011.

[7] GERHARD GENTZEN. "Untersuchungen über das logische Schliessen". *Math. Zeitschrift*, 39, 1934.

[8] ROBERT HARPER. *Practical Foundations for Programming Languages*. Cambridge University Press, Second edition, 2016.

[9] J. ROGER HINDLEY. *Basic Simple Type Theory*. Cambridge University Press, Cambridge, U.K., 1997.

[10] LUCA INCURVATI AND JULIAN J. SCHLÖDER. "Weak Rejection". *Australasian Journal of Philosophy*, 95(4):741–760, 2017.

[11] LUCA INCURVATI AND JULIAN J. SCHLÖDER. *Reasoning With Attitude: Foundations and Applications of Inferential Expressivism*. Oxford University Press, New York, 2023.

[12] LUCA INCURVATI AND PETER SMITH. "Rejection and valuations". *Analysis*, 70(1):3–10, 2010.

[13] NILS KÜRBIS. *Proof and Falsity*. Cambridge University Press, Apr 2019.

[14] MICHEL PARIGOT. "λμ-Calculus: An Algorithmic Interpretation of Classical Natural Deduction". In ANDREI VORONKOV, editor, *International Conference on Logic for Programming Artificial Intelligence and Reasoning*, volume 624 of *Lecture Notes in Artificial Intelligence*, pages 190–201. Springer, 1992.

[15] MICHEL PARIGOT. "Classical proofs as programs". In GEORGE GOTTLOB, ALEXANDER LEITSCH, AND DANIELE MUNDICI, editors, *Computational Logic and Proof Theory*, volume 713 of *Lecture Notes in Computer Science*, pages 263–276. Springer, 1993.

[16] MICHEL PARIGOT. "Proofs of Strong Normalisation for Second Order Classical Natural Deduction". *The Journal of Symbolic Logic*, 62(4):1461–1479, 1997.

[17] DAG PRAWITZ. *Natural Deduction: A Proof Theoretical Study*. Almqvist and Wiksell, Stockholm, 1965.

[18] GREG RESTALL. *An Introduction to Substructural Logics*. Routledge, 2000.

[19] GREG RESTALL. "Multiple Conclusions". In PETR HÁJEK, LUIS VALDÉS-VILLANUEVA, AND DAG WESTERSTÅHL, editors, *Logic, Methodology and Philosophy of Science: Proceedings of the Twelfth International Congress*, pages 189–205. KCL Publications, 2005.

[20] GREG RESTALL. *Proofs and Models in Philosophical Logic*. Cambridge University Press, 2022.

[21] GREG RESTALL. "Structural Rules in Natural Deduction with Alternatives". *Bulletin of the Section of Logic*, 52(2):109–143, 2023.

[22] IAN RUMFITT. ""Yes" and "No"". *Mind*, 109(436):781–823, 2000.

[23] TIMOTHY SMILEY. "Rejection". *Analysis*, 56:1–9, 1996.