Differential Linear Logic and Bounded Time Complexity

Riley Moriss

July 12, 2023

The goal of these notes is to be exposition and as a reference. I may include details or exclude them as I see fit, based largely on their relevance to the exposition or likelyhood that I will need to refer to them many times.

Contents

Linear Logic

History and Philosophy

Why study logic? What is the study of logic? Logic is the "science" of reasoning. It seems there are many ways to take this seriously. Maybe the most successful was "classical logic", there have been successful critiques against it but it seems largely untouched.

Regardless it seems that the most immediate way to study logic is to write one down and then examine its properties. One makes observations about what they think is correct argumentation or the nature that we can "a priori" combine truths or some such thing, and then codefies them as formal systems. By nature we want to abstract forms from arguments or truths to see what they have in common and deduce logic. Now once someone has written such a thing down it seems that if you take it seriously as capturing the essence of truth and its manipulations then studying it and its properties is quite important. What can and cant be proved in the system, how can such and so be proved. What form does a correct argument take, are there equivilent arguments for the same thing. etc. From this point of view. One then becomes interested in proofs as the bearers of truths. One wants to know how complicated a proof of something will be etc. You get the domains of logic and proof theory. This process somewhat lead to two main logics "classical" and "intuitionistic". LL in a sense contains them both, moreover it allows more delicate control over the structural rules of them. In this way LL can be seen as a tool for proving things about the logics that exist inside of it and that people for some reason care about (they capture the one true logical method or something).

Another point of veiw may be that Linear Logic is itself the one true logic (or a better attempt at it), that it better mirrors the essence of human reason, or the reason of god (i.e. the logic that is latent in the universe whatever that means). So LL is itself the thing we ought to study, not merely a tool.

One can consider evolutionarily how logic may have came to be, why we beleive in implication and conjunction and negation etc. The thing that LL initially was emphasissed for was its ability to represent the finite vs infinite. To represent causal implication and logical implication. This may be good justification for beleiving LL is closer to the strictly correct notion of reasoning because there are many rules of thought that are finitistic (causal), while there are the "logical" (mathematical) laws that are infinite that are also captured by LL.

If logic is supposed to capture how we think, or how truth in an abstract sense "works" then we can ask the meta-quesiton. How do we compare different logics. What makes such and such logic better or more true than another. This is obviously a subtle and meaningless question. Regardless it seems important. One answer is utility. A logic is correct iff I get the things I want to be true being true from it. There are empiricist criterions, a rule is logically valid in the limit of some inductive argument. One suggested by Dan that perhaps one should take seriously the idea of symmetry as a logical asset, the more symetry a system seems to have the better. What one means by symmmetry (delimiting) and why an inconsistent system is not the ultimate in symetry is not clear but its a concept that can be applied just as validly as empiricism or the more honest blind preference.

Well anyway one might claim that LL has more symetry than classical and for this reason is to be prefered because it also has the same utility etc.

It occurs to me that maybe I could think of conjuction, implication and negation as trancendental logical rules in the Kantian sense (necessary for the possibility of [LOGICAL] thought).

Intuitionistic Turn

UNDER CONSTRUCTION

What is meant by intuitionistic or constructive logics. Brouwer and Heyting were the spearheads.

The synthesis of the movement seems to be the proofs as programs paradigm. If there is a Curry-Howard type isomorphism, a deterministic cut elimination process or you restrict to single formulas on the right of the turnstile.

Why is this related to double negation contradiction etc. There is also the point that Choice is non-constructive but that is somewhat seperate.

Why is the last thing constructive, precissely because it corresponds to the other two.

REFERENCE AND PROOF FOR THIS. THE UNDERSTANDING IS ALWAYS IN THE PROOF

Well the point is that Girard "discovered" LL in a semantics for intuitionistic logic and it was somewhat of a selling point in the original work that his logic was "constructive with an involutive negation". I kind of want to understand the idea of constructive more deeply here.

Quick Summary of Linear Logic

[**?**], [**?**], [**?**] as well as the original paper and the appendix of "Proofs and Types" by Lafont.

Linear logic appears to capture the logical behaviour of things like causality, question and answer, action and reaction, once, infinitely many times, both and either. These interpretations all focus on the role of the "bang" and "why not" modalities, which formally replace the structural rules.

Classical logic has the property that all proofs of a sequent must be equivilent under cut elimination (See Lafonts introduction for an explination why); this means that there can be no non-trivial denotational semantics and no Curry-Howard like correspondence to computation (all programs that compute the same thing are identified). Linear Logic refines the structural rules of classical logic to remidie this, i.e. it gives a classical setting in which the Curry-Howard type correspondence will hold. This is the sense in which it is constructive. It is nicer that intuitionistic logic because it maintains the symmetries of classical logic (involutive negation) while claiming the correspondence to programs still. The deduction rules, using a one sided sequent caqlculus presentation for its compacness, are below:

Identity group:

$$\frac{}{\vdash A, A^{\perp}}\ \text{axiom} \qquad\qquad \frac{\vdash \Gamma, A \qquad \vdash A^{\perp}, \Delta}{\vdash \Gamma, \Delta}\ \text{cut}$$

Structural Rule:

$$\frac{\vdash \Gamma}{\vdash \Gamma'}\ \text{exchange } (\Gamma' \text{ is a permutation of } \Gamma)$$

Logical Rules:

$$\frac{}{\vdash 1}\text{ one}$$

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

$$\frac{}{\vdash \Gamma, \top}\text{ true}$$

$$\frac{\vdash \Gamma, A \qquad \vdash \Gamma, B}{\vdash \Gamma, A \& B}\text{ with}$$

$$\frac{\vdash ?\Gamma, A}{\vdash ?\Gamma, !A}\text{ of course}$$

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, ?A}\text{ deriliction}$$

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, \forall x.A}\text{ for all (x not free in } \Gamma)$$

$$\frac{\vdash \Gamma}{\vdash \Gamma, \bot}\text{ false}$$

$$\frac{\vdash \Gamma, A, B}{\vdash \Gamma, A ⅋ B}\text{ par}$$

(no rule for zero)

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, A \oplus B}\text{ left plus}$$

$$\frac{\vdash \Gamma, B}{\vdash \Gamma, A \oplus B}\text{ right plus}$$

$$\frac{\vdash \Gamma}{\vdash \Gamma, ?A}\text{ weak}$$

$$\frac{\vdash \Gamma, ?A, ?A}{\vdash \Gamma, ?A}\text{ contraction}$$

$$\frac{\vdash \Gamma, A[t/x]}{\vdash \Gamma, \exists x.A}\text{ exists}$$

The language is given over some set of atomic propositions and their negations $\{p, p^\perp, q, q^\perp, ...\}$. There are four constants $1, \bot, \top, 0$ for the connectives $\otimes, ⅋, \&, \oplus$ respectively as well as the modalities $!, ?$ and the quantifiers $\forall, \exists$. Apart from the atomi negations the negation operation is meta-notation (defined) for the following:

$$1^\perp \equiv \bot \qquad \bot^\perp \equiv 1$$

$$\top^\perp \equiv 0 \qquad 0^\perp \equiv \top$$

$$(p)^\perp \equiv p^\perp \qquad (p\bot)^\perp \equiv p$$

$$(A \otimes B)^b ot \equiv A^\perp ⅋ B^\perp \qquad (A ⅋ B)^\perp \equiv A^b ot \otimes B^\perp$$

$$(A \& B)^\perp \equiv A^\perp \oplus B^\perp \qquad (A \oplus B)^\perp \equiv A^\perp \& B^\perp$$

$$(!A)^\perp \equiv ?A^\perp \qquad (?A)^\perp \equiv !A\bot$$

$$(\forall x.A)^\perp \equiv \exists x.A^\perp \qquad (\exists x.A)^\perp \equiv \forall x.A^\perp$$

Likewise implication is meta-notation for the following

$$A \multimap B \equiv A^\perp ⅋ B$$

Its interesting to notes that the deriliction rule is equivilent to adding either of the axioms $B \multimap ?B$ or $!B \multimap B$.

Inclusions of Other Logics

There is an embedding intuitionistic $LJ \hookrightarrow LL$

Theorem. *A sequent in the propositional (no quantifiers) fragment of Gentzens LJ is provable iff its image is provable in LL under the translation*

$$A \wedge B \mapsto A \& B$$

$$A \vee B \mapsto !A \oplus !B$$

$$A \implies B \mapsto !A \multimap B$$

$$\neg A \mapsto !A \multimap 0$$

Proof.

Proof? Extend to full generality of quantifiers, I imagine its identity map.

Translation and similar theorom for classical (LK)

Cut Elimination

Want to connect this up to the nontrivial denotational models as well as how it replaces classical logic. Connect to the Curry-Howard correspondence for LL.

I will give an outline of the proof strategy given in [**?**].

Fix SN graphic

There are many technical definitions required to set up the apparatus about which we prove SN. The authors define approximately 3 distinct but similar systems of nets. There are the sliced pure structures (sps) that are perhaps the most general, and in which several theorems of normalisation are proven. There is the $s^\ell ps$ structures, effectively labelled sps that are used to show confluence and WN of a subcolleciton of sps. Then there are the LL nets, these are a slight variation of sps with a second additive box as well as labelled edges. These are obviously the main interest and their strong normalisation follows from them being slicable, translatable into the language of sps.

For each of these systems new cut elimination rules are defined. sps has 10, while $s^\ell ps$ takes the non-erasing subset of the sps cut rules and modifies them to edit the labels in appropriate ways (but mainly unchainged). LL nets have three new cut elimination rules; $(\forall/\exists)$, (ccad) and $(\&/\oplus_i)$ for dealing with the quantifiers and additive rules that are now in sps. The difficulty of translating the normalisation of sps into LL nets is effectively in dealing with the (ccad) rule.

The technical definitions required to formally set up these cut elimination relations are:

- Module: An sps which may have edges without sources at depth 0. These are hypotheses. The contractum of a reduction step will be a module.
- One hole context: A sps with (exactly one) special cell the hole. The hole has some arity and coarity and is supposed to be a missing link (or peice of the sps) so to say. You can plug modules into holes by matching the hypotheses and conclusions with the holes arity and coarrity.
- Context closure: The context closure of a binary relation R between modules is the smallest relation containing R such that for any one hole context $\omega[]$ if $\mu R\mu'$ then $\omega[\mu]R\omega[\mu']$.

As well as the different types of cut links that rule out pathological (non-typable) nets,

- Clash: The premises of the cut are not dual edges
- Deadlock: The premises of the cut are the conclusions of the same axiom rule (loop)
- Reducible: Otherwise

The final technical definition needed is that of

- Swtiching of a flat: A subgraph obtained by forgetting directions and deleting one of the premises for every $\parr$ and $?c$ nodes.
- Switching acyclic flat: A net such that every switching is acyclic.

which will be a necissary condition for the strong normalisation of some nets. Recall that a flat is the basic building block of an sps, just a graph built of the differnt links, ensuring that the arity and coarity is matching (only condition, no boxes for the bangs are included yet).

1.5.1 Rewrite Theory Results

First some definitions. If $\xrightarrow{x}$ is a binary relation on arrows then

- $\xrightarrow{x=}$ is the reflexive closure
- $\xrightarrow{x+}$ is the transitive closure
- $\xrightarrow{x*}$ is the reflexive transitive closure

And the different types of confluence: (1) Local Confluence, (2) Confluence and (3) Strong Confluence

$$\begin{tikzcd} \pi \arrow[r, "x"] \arrow[d, "x"'] & \pi_2 \arrow[d, dashed, "\exists x*"] \\ \pi_1 \arrow[r, dashed, "\exists x*"'] & \exists \pi_3 \end{tikzcd}$$

(1)

$$\begin{tikzcd} \pi \arrow[r, "x*"] \arrow[d, "x*"'] & \pi_2 \arrow[d, dashed, "\exists x*"] \\ \pi_1 \arrow[r, dashed, "\exists x*"'] & \exists \pi_3 \end{tikzcd}$$

(2)

$$\begin{tikzcd} \pi \arrow[r, "x"] \arrow[d, "x"'] & \pi_2 \arrow[d, dashed, "\exists x"] \\ \pi_1 \arrow[r, dashed, "\exists x"'] & \exists \pi_3 \end{tikzcd}$$

(3)

There are three key lemmas in the proof that are referenced.

Theorem (Newman)**.** *A locally confluent and strongly normalizing relation is confluent*

Theorem (Hindley-Rosen)**.** *If two relations are confluent and commute then their union is confluent*

Theorem (Di Cosmo-Piperno-Geser)**.** *Given two relations $\xrightarrow{x}, \xrightarrow{y}$ such that $\xrightarrow{x}$ is SN and for every π_1, π_2, π there is a π_3 and arrows making the following commute*

$$\begin{tikzcd} \pi \arrow[r, "x"] \arrow[d, "y"'] & \pi_2 \arrow[d, dashed, "\exists y*"] \\ \pi_1 \arrow[r, dashed, "\exists x+"'] & \exists \pi_3 \end{tikzcd}$$

then $\xrightarrow{x}$ and $\xrightarrow{y}$ commute

1.5.2 The Proofs

If one can read the figure **??** then each of the arrows is pretty clear with perhaps one or two subtleties. So the main meat of the proof should be in the leaves of this diagram. These then are what we will outline here. My goal is to give the idea of the proof, a supplement to going through the proof carefully in the paper.

The subtleties of the arrows are: Recalling that SN^{ℓ} is basically the strong normalisation of labelled sps under non-erasing reduction steps. The labels however really play no role in the reduction so this makes $SN^{\neg e}$ an immediate consequence. The other arrows are either immediate or clear short and clear proofs in the paper (typically inspecting cases to apply the hypothesis).

Now asside from the lemmas proven in other papers (referenced above) there are four (six) subproofs that are very delicate. The others are largely inspecting cases or simple inductions. Those are:

Theorem. *$\xrightarrow{log}$ is SN*

Proof. The idea of proving any SN is to find some strictly reducing measurment of complexity and then induct on it. Here we will use the idea of a single threaded slice to create a reducing measure on the sps.

A single threaded slice (sgth) is simply a sps where the multisets have a single element sps have multisets so is a sgth a single multiset element still with multiplicity, or a single multiset element with multiplicity one?

(or empty). Then for a sps β we have $sgth(\beta)$ which is the (multi?)set containing all the different sgth contained in β (forget the other multiset info).

The proof is then a kind of double induction. First assume there is only one slice in the sps. This is a sort of WLOG assumption because the slices dont interact with one another during cut elimination and so the inductive case immediately follows from the single case, by applying it to each of the slices individually.

The second thing inducted on is the depth of the cut. Again WLOG we treat the case of the cut at depth 0 and then observe that (second inductive step) a cut at level d never introduces a cut at level d-1 (inspection of the elimination rules). Hence we can eliminate cuts level by level.

So a cut at level 0 of an sps with a single slice is WLOG all we need to consider. There are 6 logical rules. Only one is non-trivial the $(!/?d)$ reduction rule. The others all reduce the number of links in the sps at each reduction step so we are done. With $(!/?d)$ we need a more subtle measure. The one given by the authors is

$$|\pi|_{log} = \sum_{\beta \in sgth(\pi)} \text{\# of links in } \beta$$

The rest of their proof is pretty intelligible

Theorem. $\xrightarrow{str}$ *is SN on AC sps*

Proof. The concepts of

- Exponential paths: A path in an sps is a path in one of its flats (so it cannot go through a bang into a box, nor can it go from one element of a multiset to another). A path is exponential if
  - It crosses only $!$, $?w$, $?c$ or cut links
  - It can only cross upwards edges that are either conclusions or premises of ? links (?w, ?c)
  - It can only cross downwards the main conclusions of of a ! link that is connected to a cut link

  FIX EXPONENTIAL PATH IMAGE

- Expoentntial dependence: Two edges exponentially depend if there is an exponential path between them. For an edge a then call $pred(a)$ the set of edges such that there is an exponential path FROM a crossing just one node.

- Wd of an edge, ld of an edge and bang links: Intuitively the width and length of the edge in the proof.

  $$ln^{\pi}(a) = \begin{cases} 1 + ln^{\pi}(\text{ main conclusion of ! link }) + \underset{\gamma \in \text{ slices of ! box}}{sup} ln(a^{\gamma}) & \text{a is an auxilery conclusion of some ! link and } a^{\gamma} \text{ is the conclusion corresponding to a in the slice } \gamma \\ 1 + sup_{b \in pred(a)} ln^{\pi}(b) & else \end{cases}$$

  Note that if the edge is not apart of an exponential path then it will have length one because its set of predicessors will be empty.

  $$wd^{\pi}(a) = \begin{cases} 1 & pred(a) = \emptyset \\ wd^{\pi}(\text{main conclusion of ! link})\left(1 + \sum_{\gamma \in \text{ slices of ! box}} wd^{\gamma}(a^{\gamma})\right) & \text{a is an auxilery conclusion of some ! link} \\ 1 + \sum_{b \in pred(a)} wd^{\pi}(b) & else \end{cases}$$

  Note that if a is the auxilery conclusion of a bang link then its unique predicessor is the main conclusion of the same bang link.

  These definitions are extended to !-links by begin applied to the main conclusion edge.

Are essential in defining the measure on which the induction takes place. Note that the assumption of AC is here used to make these quantities Wd and ld well definied. This is because the only way an exponential path could be infinite (in a finite graph) is if there were cycles.

The measure is then
$$|\pi| = \sum_{\ell \in !^0(\pi)} wd^{\pi}(\ell)(ln^{\pi}(\ell) + |\pi^{\ell}|)$$
Where $!^0(\pi)$ is the set of depth 0 bang links and π^{ℓ} is the sps associated to the bang link ℓ.

One sees from the definition that the length and width are invariants of module changes (plugging in different modules to one hole contexts, under some hypotheses will not change the ln and wd). In particular fact 3.9 states that if all the pending edges of the module have exponential dependence on edges of the one hole context then when plugging in a different module the corresponding edge will have the same wd and ln (because by hypothesis the exponential dependence of that pending edge is ONLY in the one hole context and the measures are built out of exponential dependenec). It also states that the wd and ln of an edge of the one hole context is determined only by the ln and wd of the pending edges of the module being plugged in (also obvious because the one hole context must interact through these pending edges and the wd and ln measures capture recursively the internal structure).

The proof makes a lot of the same simplifying moves as the previous SN proof. So we restrict to the case where the proof has exactly one slice and notice that the str reduction steps preserve this property (they dont increase the number of slices at the reduction level; by inspection of the rules). We also only deal with cuts at depth 0 with the induction being immediate.

The proof is in three cases one for each str rule, these are further divided into two parts each.

- Showing that for every conclusion d of the sps its corresponding conclusion $\vec{b}$ of the reduced sps it holds that
$$ln(d) \geq ln(\vec{d}) \qquad wd(d) \geq wd(\vec{d})$$
- It will follow from the proofs of these facts that the measure strictly reduces at each step.

Case (!/?w): Is left as an exercise. The width and length of each of the conclusions will strictly decrease because this reduction step erases the box (width) and erases the bang link (length). Hence the measure also strictly decrease.

Case (!/?c): The key observations are that there are edges in the redex that have the same length and width allowing one to derive inequalities. Moreover one can split the measure into two parts, measuring the link being reduced and measuring all the others.

In this case one notices that $!^0(\alpha) = \{o\} \sqcup S$ and that the reduction does not effect S, i.e. every edge and link in S has a unique residue, $\vec{S}$. So if $\alpha \rightsquigarrow \alpha'$ we have that $!^0(\alpha') = \{\vec{o_1}, \vec{o_2}\} \sqcup \vec{S}$. (figure 15 in the paper is essential to understand the naming convention here)

reproduce the figure

And this gives
$$|\alpha| = wd(o)(ln(o) + |\pi^o|) + |S|$$
$$|\alpha'| = wd(\vec{o_1})(ln(\vec{o_1} + |\pi^{\vec{o_1}}|) + wd(\vec{o_2})(ln(\vec{o_2} + |\pi^{\vec{o_2}}|) + |\pi^o|) + |\vec{S}|$$
In this case one gets equality. In the calculuation the following two equalities are shown
$$ln(d) = 1 + sup(ln(\vec{d_1}), ln(\vec{d_2})) \qquad wd(d) = wd(\vec{d_1}) + wd(\vec{d_2})$$
where d is an auxilery conclusion of the reduced ! link.

Now the result follows because it is clear that $|S| = |\vec{S}|$ and from the equalities the strict inequality of
$$wd(o)(ln(o) + |\pi^o|) > wd(\vec{o_1})(ln(\vec{o_1} + |\pi^{\vec{o_1}}|) + wd(\vec{o_2})(ln(\vec{o_2} + |\pi^{\vec{o_2}}|)$$
follows

Case (!/!): In this case one notices that $!^0(\alpha) = \{o, u\} \sqcup S$ and $!^0(\alpha') = \{\vec{u}\} \sqcup \vec{S}$. (figure 16) reproduce the figure

Hence

$$|\alpha| = wd(o)(ln(o) + |\pi^o|) + wd(u)(ln(u) + |\pi^u|) + |S|$$

$$|\alpha'| = wd(\vec{u})(ln(\vec{u} + |\pi^{\vec{u}}|) + |\vec{S}|$$

Now we have that $ln(u) = ln(\vec{u})$ and $wd(u) = wd(\vec{u})$ after some staring at the different edges (in particular the only one that would cause a problem is the edge involved in the cut however this doesnt contribute to the length or wd becuase you cant travel up such an edge in an exponential path).

Similar to the first case one then shows that for an auxilery conclusion of o one has $ln(d) = ln(\vec{d})$ and $wd(d) \leq wd(\vec{d})$ and then combines it with

$$wd(\vec{u})(ln(\vec{u}) + |\pi^{\vec{u}}|) < wd(u)(ln(u) + |\pi^u|) + wd(o)(ln(o) + |\pi^o|)$$

Theorem. $\xrightarrow{log\ell}$ *is locally confluent*

Proof. Recall that the ℓ reductions are non-erasing and act on labelled sps. Recall that local confluence is allowing the use of reduction steps in the reflexive transitive closure of the reduction relation. It is enough to show local confluence on the level of slices for the same reason we could assume there was only one earlier.

Now because we have to show local confluence we need to show that for any two $log\ell$ reductions there is a sequence of reductions that relates them. Unfortunately there are five reduction relations and therefore $5 + 4 + 3 + 2 + 1 = 15$ combinations. In the paper they treat of only one. It is by inspection effectively.

Theorem. $\xrightarrow{str\ell}$ *is locally confluent*

Proof. Again you have to inspect the cases, in this proof there are only two reduction relations so only 3 combinations. They show two in the paper.

Theorem. *For every sps the diagram commutes*

$$\begin{tikzcd}
\langle \pi, \ell \rangle \arrow[r, "log\ell"] \arrow[d, "str\ell"'] & \langle \pi_2, \ell_2 \rangle \arrow[d, "str\ell*"] \\
\langle \pi_1, \ell_1 \rangle \arrow[r, "log\ell+"'] & \langle \pi_3, \ell_3 \rangle
\end{tikzcd}$$

Proof. Restrict to one slice in the sps. Call r the cut link reduced in the $log\ell$ step and call t the cut link reduced in the $str\ell$ step.

Case 1: r has depth 0: The reduction of t does not affect r.

Case 2: t is depth 0 r contained in !-link o: Note that o is the !-link containing r at depth 0. If o is not involved in the t cut then we are fine. So there are some cases with o being different edges in the two cuts $(!/?c), (!/!)$. These cases are just by inspection too.

Case 3: t and r are in boxes: They are both contained in some bang link at depth zero. If its a different bang link we are done. If its the same link then the box reduces in the two directions of the diagram, then applying the induciton hypothesis we are done (either the cut r is in this box and we apply above or we apply this case again and go deeper). So the sps that fits in the diagram is the sps where the box containing the cuts being reduced is replaced by the box that was given by applying the induction.

Theorem. *Given a LL net β then $sl(\beta) \in WN^{\neg e}$*

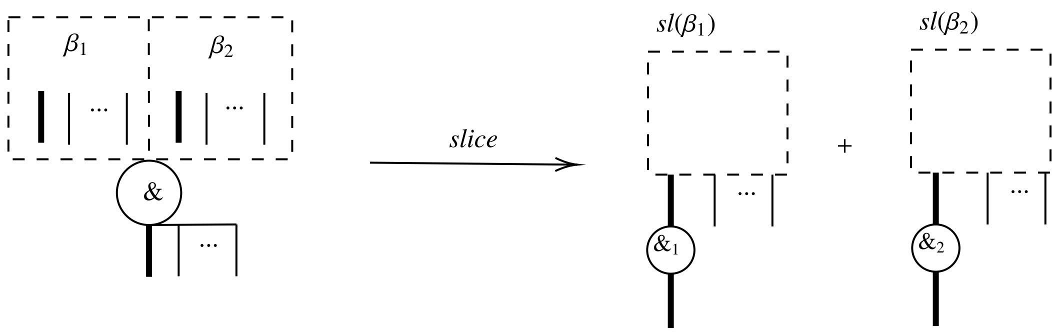
Proof. This proof has two parts. Understanding the translation of a LL net into sps and then applying Girards reducibility candidates to this object

Translating LL nets to sps The main difference is additive boxes. sps has additive links (unitary), in LL however they are handled via boxes. There is also the simplifying fact of the correctness criterion, a LL net is always switching acyclic moreover it has typed edges so all its cuts are reducible.

page 50 question

The slicing of an sps is definied by induction of a sequentialisation (translation of the net to some sequent calculus style proof) of the net. The authors state that slicing is independent of the sequentialisation however it seems this is inconsequential to the proof(because all LL nets are sequentialisable anyway). Now they give a formal definition of all cases, here is a summary

- If the net is an sps then leave it alone
- Look at the conclusions of the proof and "climb up" the wires, if you pass an sps link then leave it and slice one up
- If you reach a quantifier remove it and its conclusion connect the hypothesis up and slice one up
- Slicing a bang link just slices the box etc
- The only one where something non-trivial happens is slicing an & box. Let β be the & link and β_1 and β_2 be the left and right, respectively, boxes of the link. Then $sl(\beta)$ is given by adding a $\&_i$ link to every slice of $sl(\beta_i)$ on its main output and just plugging in the other wires. Then taking the union of these multisets.



Reducibility Candidates First a term is a LL net with a distinguished conclusion (the type). If you have two terms, one of type A and the other of type $A^\perp$ then we can form $CUT(\beta,\beta')$ by connecting the distinguished conclusions through a cut link.

Given a set of terms X of the same type, A, we form

$$X^\perp = \{\beta' \in \text{ terms of type } A^\perp : \forall \beta \in X \;\; sl(CUT(\beta,\beta')) \in WN^{\neg e}\}$$

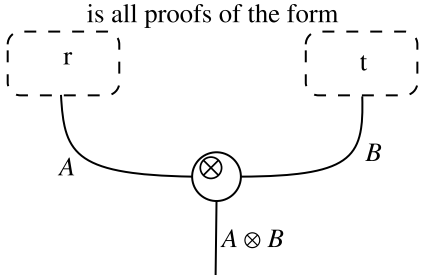
Note that

$$\text{X contains axiom link} \implies sl(X^\perp) \subseteq WN^{\neg e}$$
$$sl(X) \subseteq WN^{\neg e} \implies X^\perp \text{ contains axiom link}$$

Now we will define some operations on terms (with types); i.e. sets of proofs with designated conclusions. Designated conclusions are here denoted by solid dangling edges. Let R, T be term collections of proofs of respective types A and B. Let $r \in R$ and $t \in T$

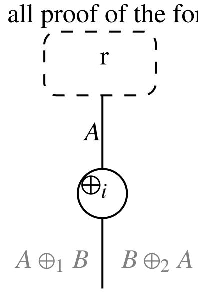
$R \otimes T$

is all proofs of the form



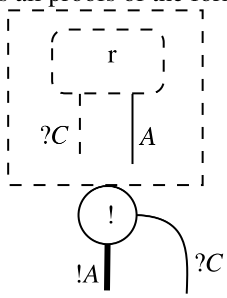
$\oplus_B^i R$

is all proof of the form



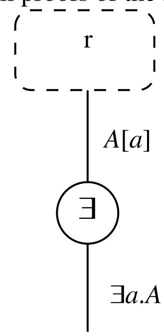
$\S R$

is all proofs of the form



$\exists R$

is all proofs of the form



Note the subtlety with $\S R$, it is the collection of terms formed on where $r \in R$ of such a form. So in particular even if R is nonempty, it may not contain any nets that contain a $?C$ conclusion, hence $\S R$ would be empty ([**?**]).

A reducibility candidate of type A is a colleciton of terms X of type A such that

- $X \neq \emptyset$
- $sl(X) \subseteq WN^{\neg e}$
- $X = X^{\perp\perp}$

Now given a proposition A (type) with $FV(A) \subseteq a = \{a_1, ..., a_n\}$, and a sequence of reducibility candidates $X = X_1, ..., X_n$ of respective type $B = B_1, ..., B_n$ then we define the reducibility candidate $\mathcal{R}(A[X/a])$ of type $A[B/a]$:

- $\mathcal{R}(1[X/a]) = \{1\}^\perp$
- $\mathcal{R}(\top[X/a]) = SN(\top)$ the set of strongly normalising terms of type $\top$
- $\mathcal{R}(a_i[X/a]) = X_i$
- $\mathcal{R}(A \parr B[X/a]) = (\mathcal{R}(A[X/a])^\perp \otimes \mathcal{R}(B[X/a])^\perp)^\perp$
- $\mathcal{R}(A \& B[X/a]) = (\oplus^1_B \mathcal{R}(A[X/a]^\perp) \cup \oplus^2_A \mathcal{R}(B[X/a]^\perp))^\perp$
- $\mathcal{R}(?A[X/a]) = (\S\mathcal{R}(A[X/a]^\perp))^\perp$
- $\mathcal{R}(\forall b.A[X/a]) = (\{t \in terms(\exists b.A^\perp[B/a]) :$ exists a Y a RC of type C with $t \in \exists\mathcal{R}(A^\perp[X, Y/a, b])\})^\perp$

This is extended to arbitrary types through orthogonality, i.e. the definition

$$\mathcal{R}(A^\perp[X/a]) = \mathcal{R}(A[X/a])^\perp$$

its not clear in the paper how the quantifier is definined possible typo in it...?

Lemma: The $\perp$ of a set of proof is always a RC; $A^{\perp\perp\perp} = A^\perp$.

Corollary: $\mathcal{R}(A[X/a])$ is a RC for any type A and any appropriately sized sequence of RC X.

This is fantastic, now there are reducibility candidates of any type that we can use to cut against the conclusion of any proof. Moreover we know how they work with substitution:

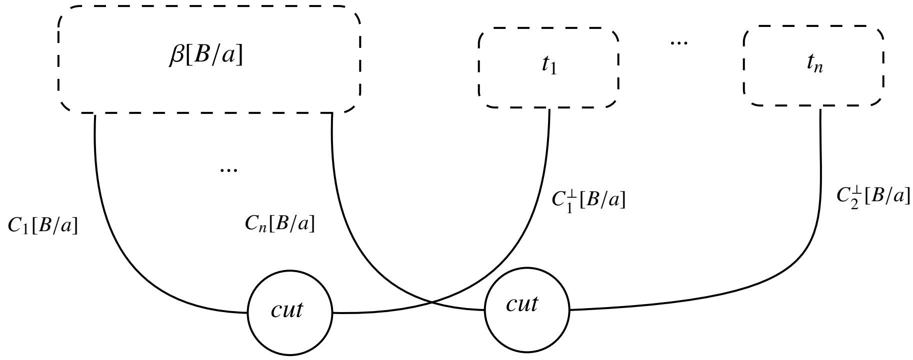
Lemma:

$$\mathcal{R}(A[C/b][\mathbf{X/a}]) = \mathcal{R}(A[X, \mathcal{R}(C[X/a])/a, b])$$

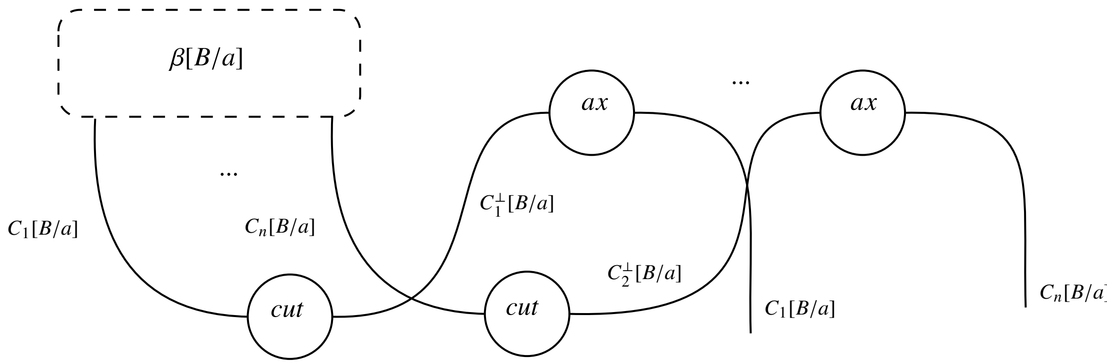
Where the bold substitution is that of RC and the other one is normal substituiton.

Completely unclear on what the right hand side even means...

So now we will show that every term is **reducible**: Given a proof β with conclusions $C = (C_1, ..., C_n)$ and $FV(C) \subseteq a = (a_1, ..., a_n)$ then we say β is reducible if for any appropriately sized sequence of RC X of type B cutting $\beta[B/a]$ against an arbitary $t \in \mathcal{R}(C^\perp[X/a])$ slices to a $WN^{\neg e}$ sps. Note that C is a sequence and therefore t is a sequence where $t_i \in \mathcal{R}(C_i^\perp[X/a])$, moreover by definition we have that t_i is of type $C_i^\perp[B/a]$ so the cut makes sense. This can be visualised as



This is sufficient because We can then let $B = a$ and hence t_i has type $C_i^\perp[B/a] = C_i^\perp[a/a] = C_i^\perp$ and $\beta[B/a] = \beta$. Moreover one oserves that by definition $\mathcal{R}(C^\perp) = \mathcal{R}(C)^\perp$ as well as $sl(\mathcal{R}(C)) \subseteq WN^{\neg e}$ hence by our properties of $\perp$ we know that $\mathcal{R}(C)^\perp = \mathcal{R}(C^\perp)$ contains an ax link. So if a net is reducible then in particular



which is $WN^{\neg e}$ iff β is (clearly because the cut elimination removing the ax link detour is non-erasing).

So we need only check that they are all reducible. This is done by 16 cases one for each rule of LL; or more precisely induction on a sequentialisation of the net. So there is a sequent calculus proof coresponding to the net and the last rule of this proof must be one of the 16 LL rules, hence 16 cases. The induction is then assuming the rest of the net is $WN^{\neg e}$ and you apply this rule then is that $WN^{\neg e}$.

It becomes about inspecting the shape of the two nets cut together, because we have $t \in \mathcal{R}(A)$ for some type A we know (by the inductive definition of $\mathcal{R}$) some of the structure of t and hence we can recurse into that structure. Note that the substitution Lemma is needed in the case of quantifiers.

Did the substitution lemma get used elsewhere?

Proof Nets Formalism

UNDER CONSTRUCTION

This is the notation for equivilence classes of proofs. It can be slick or quite complicated. Its nice for the multiplicative fragment, however things get out of hand with the exponentials and "boxes". I dont yet understand this.

You can build up the proof nets inductively or you can define them as those proof structures that satisfy the long trip condition. This is effectively saying something about the fact that you can embed a circle into the graph in a way that goes through each node exactly twice..

Theorem. *A proof structure is a proof net iff it has a long trip.*

Differential Linear Logic

The Differential Lambda Calculus

2.1.1 Linearity

In λ-calculus linearity means that an argument is used exactly once. This is naturally connected to the concept of head reduction which is a reduction strategy evaluating the subterms in linear position.

There is a lot more on head reduction I think to look into

In algebra linearity means commutation with sums and scalars. The goal of DiLC is to connect these senses of linearity.

2.1.2 Differentials

In differential geometry we think of the differential of a map $f : M \to N$ between smooth manifolds as a smooth map linear map between vector spaces $Df_p : T_pM \to T_{f(p)}N$ or in more generality

$$Df : M \to Hom(T_pM, T_{f(p)}N)$$

with $Df(p)(v)$ sometimes denoted as $Df_p \cdot v$.

In ordinary vector calculus the directional derivative of a map $f : \mathbb{R}^n \to R$ along a vector v is given by

$$D_v f(x) = \lim_{h\to 0} \frac{f(x-hv) - f(x)}{h}$$

Alernative notation is that you dot product the derivative vector with the vetor along which you are taking the derivative.

write this out, I think this will make their notation more transparent?

2.1.3 The Calculus

The terms of the calculus over some commutative unital semi-ring $R = \{a, b, ...\}$ are given by a countable set of term variables $\{x, y, ...\}$ and

$$s, t ::== x|\lambda x.s|(s)t|D_i s.(t_1, ..., t_n)|0|as + bt$$

The following identities are then made on terms:

- α-equivilence
- $(as + bt)u = a(s)u + b(t)u$
- $\lambda x.(as + bt) = a\lambda x.s + b\lambda x.t$
- $D_{i_1,...,i_n} s = D_{i_{\sigma(1)},...,i_{\sigma(n)}} s$ where σ is a permutation
- $D_i(D_{i_1,...,i_n} x \cdot (u_1, ..., u_n)) \cdot u = D_{i,i_1,...,i_n} x \cdot (u, u_1, ..., u_n)$
- $$D_i(D_1^n \lambda xt \cdot (u_1, ..., u_n)) \cdot u = \begin{cases} D_1^{n+1} \lambda xt \cdot (u, u_1, ..., u_n), i = 1 \\ D_1^n \lambda x(D_{i-1} t \cdot u) \cdot (u_1, ..., u_n), i > 1 \end{cases}$$
- $D_i(t)v \cdot (u_1, ..., u_n) = (D_{i+1} t \cdot (u_1, ..., u_n))v$
- $D_i(\sum a_s s) \cdot (\sum b_u u) = \sum a_s b_u D_i s \cdot u$

A technical remark from [?] is that a single differential Ds suffices where originally ER had defined one piecewise $D_i s$. ER set this up a little differently, likely for the purposes of certain proofs, as well as making the definition as just the free R module generated on some set of terms. We want the differential to mimick the behaviour of the normal derivative, intuitivley reading $D_i s \cdot t$ as the derivative of s along t with respect to the i^{th} variable.

Lemma. $D_i(D_j s \cdot u) \cdot v = D_j(D_i s \cdot v) \cdot u$

Analogous to the interchange of partial derivatives on smooth functions. If we let $u, v \in \mathbb{R}^n$ and denote $v_i = (0, ..., 0, v_i, 0, ..., 0)$ we get that

$$D_{v_i} D_{u_j} f(x) = D_{u_j} D_{v_i} f(x)$$

Substitution is then defined inductively in the natural way

- $D_{i_1,...,i_n} y \cdot (u_1, ..., u_n)[t/x] = D_{i_1,...,i_n} y[t/x] \cdot (u_1[t/x], ..., u_n[t/x])$
- $D_1^n \lambda y v \cdot (u_1, ..., u_n)[t/x] = D_1^n \lambda y(v[t/x]) \cdot (u_1[t/x], ..., u_n[t/x])$
- $(v)w[t/x] = (v[t/x])w[t/x]$
- $(\sum a_v v)[t/x] = \sum a_v v[t/x]$

with some restrictions on free variables etc.

Linear substitution is then defined as follows: We denote the linear substitution as the partial derivative of s with respect to x along u as $\frac{\partial s}{\partial x} \cdot u$ and is given inductively

why is this not just the directional derivative?

- $$\frac{\partial D_{i_1,...,i_n} y \cdot (u_1, ..., u_n)}{\partial x} \cdot v = \delta_{x=y} D_{i_1,...,i_n} v \cdot (u_1, ..., u_n) + \sum_{i=1}^{n} D_{i_1,...,i_n} y \cdot (u_1, ..., \frac{\partial}{\partial x} u_i, ..., u_n)$$
- $$\frac{\partial D_1^n \lambda y v \cdot (u_1, ..., u_n)}{\partial x} \cdot t = D_1^n \lambda y(\frac{\partial v}{\partial x} \cdot t) \cdot (u_1, ..., u_n) + \sum_{i=1}^{n} D_1^n \lambda y v \cdot (u_1, ..., \frac{\partial}{\partial x} u_i \cdot t, ..., u_n)$$
- $$\frac{\partial (v)w}{\partial x} \cdot u = (\frac{\partial v}{\partial x} \cdot u)w + (D_1 v \cdot (\frac{\partial w}{\partial x} \cdot u))w$$
- $$\frac{\partial}{\partial x}(\sum a_v v) \cdot u = \sum a_v \frac{\partial v}{\partial x} \cdot u$$

There is a passing resemblance to the chain rule here in the application case.

The operation wants to be linear, so in the case of the application the v is in a linear position so we can take its partial derivative without problem. The w is not in linear position however so applying the operation to it must take two steps, replacing the $(v)w$ application with $(D_1 v \cdot w)w$ to get a "linear copy of w" then applying the partial derivative.

why is this a linear copy?

Lemma.

$$\frac{\partial D_i t \cdot u}{\partial x} \cdot v = D_i(\frac{\partial t}{\partial x} \cdot v) \cdot u + D_i t \cdot (\frac{\partial u}{\partial x} \cdot v)$$

Lemma. *If x is not free in t then*

$$\frac{\partial t}{\partial x} \cdot u = 0$$

Obvious parallel to the partial derivative of a constant (with respect to some variable) being zero

Lemma. *If y is not free in u*

$$\frac{\partial}{\partial x}(\frac{\partial t}{\partial y} \cdot v) \cdot u = \frac{\partial}{\partial y}(\frac{\partial t}{\partial x} \cdot u) \cdot v + \frac{\partial t}{\partial y} \cdot (\frac{\partial v}{\partial x} \cdot u)$$

Combining the last two we get

Lemma. *When y is not free in u and x is not free in v*

$$\frac{\partial}{\partial x}(\frac{\partial t}{\partial y} \cdot v) \cdot u = \frac{\partial}{\partial y}(\frac{\partial t}{\partial x} \cdot u) \cdot v$$

Interchange of second order partial derivatives.

The substitutions work well together

Lemma. *If x and y are distinct and y is not free in u or v then*

$$\frac{\partial t[v/y]}{\partial x} \cdot u = (\frac{\partial t}{\partial x} \cdot u)[v/y] + (\frac{\partial t}{\partial y} \cdot (\frac{\partial v}{\partial x} \cdot u))[v/y]$$

Lemma. *If x is not free in v and y is distinct from x we have*

$$(\frac{\partial t}{\partial x} \cdot u)[v/y] = \frac{\partial t[v/y]}{\partial x} \cdot (u[v/y])$$

Finally we can decompose the derivatives into finite sums of simpler terms (towards a Taylor expansion):

Lemma.

Lemma.

2.1.4 Confluence

The last thing to add to the calculus is a reduction rule. We extend beta reduction to include the following reductions:

- $(\lambda x.s)t \rightsquigarrow s[t/x]$
- $D_1 \lambda x.s \cdot u \rightsquigarrow \lambda(\frac{\partial s}{\partial x} \cdot u)$

It takes quite some work to then show that this is well defined etc.

Theorem. *This relation is confluent.*

Theorem. *If two ordinary lambda terms are beta equivalent in the differential lambda calculus then they are are beta equivalent in ordinary lambda calculus.*

2.1.5 Strong Normalization

We introduce types to the system with a collection of atomic types and then given two types A, B then $A \to B$ is a type. The normal lambda typing rules from lambda calculus carry over with some new typing rules:

$$\frac{\Gamma \vdash s : A_1, ..., A_i \to B \qquad \Gamma \vdash u : A_i}{\Gamma \vdash D_i s \cdot u : A_1, ..., A_i \to B} \text{(Differential Application)}$$

$$\frac{\Gamma \vdash s : A \qquad \Gamma \vdash t : A}{\Gamma \vdash as + bt : A} \text{(Linear Combination)}$$

$$\frac{}{\Gamma \vdash 0 : A} \text{(Linear Combination)}$$

Then if the semi-ring overwhich we have defined the calculus has the following properties

- $ab = 0 \implies a = 0 \vee b = 0$
- $a + b = 0 \implies a = b = 0$ (positivity)
- $\forall a \in R$ there are only finitely many $b, c \in R$ with $a = b + c$

then we can prove that this is a strongly normalising calculus. With only positivity then we still have weak normalisation.

2.1.6 Taylor Expansion

Theorem (Leibniz Rule). *For terms t and u, and distinct variables x and y, such that y is not free in u we have that*

$$\frac{\partial t[x/y]}{\partial x} \cdot u = (\frac{\partial t}{\partial x} \cdot u)[x/y] + (\frac{\partial t}{\partial y} \cdot u)[x/y]$$

contrast

$$\frac{\partial (uv)}{\partial x} = u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial x}$$

im not really seeing it ...

"clear logical meaning, expressing how derivation behaves when interacting with a contraction in cut elimination"?

Theorem (Leibniz Formula). *Under the same restrictions*

$$\frac{\partial^n t[x/y]}{\partial x^n} \cdot u^n = \sum_{p=0}^{n} \binom{n}{p} \left(\frac{\partial^n t}{\partial x^p \partial y^{n-p}} \cdot u^n \right) [x/y]$$

Lemma (Deriving Applications). *Let $t = t_1, ..., t_k$ be a sequence of terms, x a variable and u a simple term. y a distinct variable to x not free in t or u. x is not free in u. Then if $n \geq 1$ we have*

$$\frac{\partial^n (x)t}{\partial x^n} \cdot u^n = n\frac{\partial^{n-1} (u)t}{\partial x^{n-1}} \cdot u^{n-1} + (\frac{\partial^n (y)t}{\partial x^n} \cdot u^n)[x/y]$$

Which has the special case

$$(\frac{\partial^n (x)t}{\partial x^n} \cdot u^n)[0/x] = n(\frac{\partial^{n-1} (u)t}{\partial x^{n-1}} \cdot u^{n-1})[0/x]$$

Theorem (Taylors Theorem). *If s and u are terms of the ordinary lambda calculus and ξ is a distinguished variable such that $(s)u \simeq_\beta \xi$ then there is a unique $n \in \mathbb{Z}$ with*

$$(D_1^n s \cdot u^n)0 \not\simeq_\beta 0$$

Moreover for this n we have that

$$(D_1^n s \cdot u^n)0 \simeq_\beta n!\xi$$

Hence the Taylors formula holds

$$s(u) = \sum_{n \geq 0} \frac{1}{n!} (D_1^n s \cdot u^n)0$$

s(u) instead of (s)u now wtf, its not even in the syntax

> ***Proof.*** Sketch.
>
> Argue from the fact that (s)u reduces to ξ that it must be of a certain form, assuming that it is in head normal form without loss of generality (all terms are beta equivilent to a unique head normal form term).
>
> Notice that form has a recursive structure.
>
> Define a number that is related to the number of beta reductions to induct on. It counts the number of substitutions of successive head variables of s in the linear head reduction of (s)u.
>
> Using the previous Leibniz formula and deriving application lemma to compute $(D_1^n s \cdot u^n)0 \simeq_\beta n(D_1^{n-1} s' \cdot u^{n-1})0$ where s' is part of the recursive structure.
>
> The result then follows.

The example of the self application of $\lambda x.xx$ is calculuated to have taylor expansion 0.

That makes no sense? So this term is identically 0? They use equality in the thm between the application and the Taylor series.. What do they mean?

Differential LL

2.2.1 Motivation

The notion of linearity can be seen in linear logic as well. A proof can be called linear in a hypothesis if that hypothesis is used exactly once during cut elimination i.e. not duplicated or eliminated.

In everyday mathematics derivations take a nonlinear map (between manifolds) and give a linear map (between vector spaces), the new structural rule of coderiliction mimic this behaviour of taking a nonlinear proof and making it linear (in the above sense). This is dual to deriliction which take a linear proof and makes it nonlinear. Erhardt stresses that this coderiliction gives a $!A$ without making it duplicable, this makes this fragment have some nice properties (strong normalisation, all proofs are linear combinations of simple proofs) .

but its not linear like MLL is so what are they talking about.

> “ *one really needs to take the point of view of computational trinitarianism in order to understand the transition from linear logic to differential linear logic. It is more difficult to understand naively the proofs of differential linear logic than the ones of linear logic.* ”
>
> **- nlab**

The idea of computational trinitarianism is that computation, logic and category theory are three sides of the same coin. They are all talking about the same object in different ways. Taking this really seriously means that if you observe something in a computation it must have a logical meaning, and if you see some categorical structure it must have a computational and logical meaning.

2.2.2 Syntax

In outline it is full LL with some new deduction rules

$$\frac{\Gamma, !A \vdash B}{\Gamma, A \vdash B}\ \text{(Coderiliction)}$$

$$\frac{\Gamma, !A \vdash B}{\Gamma, !A, !A \vdash B}\ \text{(Cocontraction)}$$

$$\frac{\Gamma, !A \vdash B}{\Gamma \vdash B}\ \text{(Coweakening)}$$

$$\frac{\vdash \Gamma \quad \dots \quad \vdash \Gamma}{\vdash \Gamma}\ \text{(sum)}$$

What is given in [?] is a bigger term calculus and full cut elimination transformations. The term calculus has more information in typing and contexts which make its semantics in categories maybe a little clearer, maybe not. The sum rule is written

$$\frac{\forall i \in [n] \quad \Phi \vdash p_i : \Gamma \text{ and } \mu_i \in R}{\Phi \vdash \sum \mu_i p_i : \Gamma}\ \text{(sum)}$$

so the linear combination information is in the context not the type.

This is interesting to me because the type to me is the logical part and the term is the program. No but I think the context is the actual proof, this is a term calculus for the proof, so the left of the : can be seen as the lambda term or the actual proof itself and the right is the type or the thing that is proved. This explains the use of the structural rules and derecliction rules in the context.

2.2.3 Basic Results

Yes the system is confluent and has normalization.

2.2.4 What does this have to do with differentiation

`https://www.pls-lab.org/en/Differential_Linear_Logic` has a nice little paragraph. Basically in some categorical models we have a derivation map that takes a function $f : !X \to Y$ (nonlinear because of the bang)

$$D(f) : !X \to (X \multimap Y)$$

i.e. a map that takes a point in the domain of f and gives a linear (no bang) map at that point, intuitively a linear approximation to the function there.

Stratified Linear Logic

Complexity Theory

3.1.1 Recall Acceptance

We say that a TM, M, accepts w iff there is a sequence of configurations starting at w and ending at the accept state. The language of a TM, $L(M)$, is the set of all accepted strings. A language is Turing recognisable iff there is a TM that enumerates it.

3.1.2 Complexity Classes

Notes on the "Advanced Theoretical Computer Science" subject slides.

> ***Definition: Big O Notation*** *We say that $g \in O(f)$ or sometimes $g = O(f)$ iff there is some $n_0, c \in \mathbb{N}$ such that for every $n > n_0$ we have that*
> $$g(n) \leq cf(n)$$
> *Asymptotically g is bounded by f.*

A function $t : \mathbb{N} \to \mathbb{N}$ is polynomial iff $\exists r \in \mathbb{N}$ such that $t \in O(n^r)$. The function hierarchy looks like the following

$$1 < log(n) < n < n^c < exp((log(n))^c(log(log(n)))^{1-c}) < c^n < n! < c^{b^n} < \text{ Ackerman functions etc}$$

Formally an algorthm is a Turing Machine. Running the algorithm on some data is having that data encoded on the tape of the TM when it is run. The runtime of the algorithm is then the number of steps that it takes for the TM to halt (enter its accept state). The idea in complexity is to bound the runtime as a function of the length of the input.

> ***Definition: Time Complexity of TM*** *The time complexity of given TM, M, is*
> $$t_M(n) = max_{w \in \Sigma^n}\{m : M(w) \text{ halts after m steps }\}$$
> *So the longest time for the machine to halt on any given length n input.*

The complexity class of a function is then the collection of all languages that are decided in less than the time of the function i.e.

$$TIME(t) = \{L : L \text{ is a language that is decided by some TM M with } t_M \in O(t)\}$$

Theorem. *Every linear time language is regular.*

Different types of TM can have slightly different complexity properties (there are algorithms which can be given faster implimentations on two tape vs one tape machines). They are however closed under certain classes.

> ***Definition: Polynomial Time*** *P is the class of languages decidable by a deterministic TM in polynomial time*
> $$P = \bigcup_k TIME(n^k)$$

Any (reasonable) deterministic model of a TM will have the same class P.

A verifier for a language A is a deterministic TM (algorithm) M such that

$$A = \{w : \exists c, \text{ M accepts } \langle w, c\rangle\}$$

Then a language is polynomial time verifiable if this algorithm is polynomial time in the length of w.

Definition: NP

$$NP = \{\textit{ languages with a polynomial verifier}\}$$
$$= \bigcup_k NTIME(n^k)$$

so the collection of all languages decidable by a non-deterministic TM in polynomial time.

The runtime of a non-deterministic TM (NTM) is the maximum over all branches of the computation. Again the class NP is robust against changes to model of non-deterministic computation.

3.1.3 Hardness

Definition: *A language $\mathcal{A}$ reduces in polynomial time to a language B, denoted $\mathcal{A} \leq_p B$ iff there is an algorithm f such that*

$$w \in \mathcal{A} \iff f(w) \in B$$

B is NP hard iff for any $A \in NP$ we have that $A \leq_p B$. B is NP complete (NPC) if in addition $B \in NP$. There are several interesting theorems about this

- $A \leq_p B, B \in P \implies A \in P$
- $B \in NPC, B \in P \implies NP = P$
- $B \in NP - hard, B \leq_p C \implies C \in NP - hard$
- $P \neq NP \implies NPI = NP \setminus (P \cup NPC) \neq \emptyset$ i.e. if P is not NP then there are non-polynomial non-NP complete problems.

P is closed under compliment however it is not known whether the same is true of NP. The hypothesised class is called co-NP i.e.

$$co - NP = \{\mathscr{L} : \mathscr{L}^C \in NP\}$$

Stratified LL: $LL_\S$

3.2.1 Motivation

There have been two successful strategies for creating subsystems of LL with bounded complexity properties. The idiosyncratic soft logic of Lafont [**?**] and a collection of systems from Girard, Terui, Baillot, Mazza and others that utilise a system of stratification or levels. Similar to type theory the idea is to label sequents with a "level" and then disallow the communication of levels (through the deduction rules only allowing cuts and other operations to happen on the same level). After sever years of refinement [**?**] introduced a general framework in which to understand these logics using a notion of stratification. The logic is called Stratified Linear Logic and abbreviated to $LL_\S$

The general insight is that when stratification is tied to the exponentials complexity properties emerge because those are the operations in LL that are responsible for the computational power.

3.2.2 The Calculus

The calculus has all the formulas of LL with one new modality $A ::== A|\S A$, that is self dual (negation defined as) $(\S A)^\perp \equiv \S A^\perp$. The sequent calculus is now a 2-sequent system, which just means that each formula has a label of level attached.

Identity group:

$$\frac{}{\vdash A^i, A^{\perp^i}}\text{ axiom} \qquad\qquad \frac{\vdash \Gamma, A^i \qquad \vdash A^{\perp^i}, \Delta}{\vdash \Gamma, \Delta}\text{ cut}$$

Structural Rule:

$$\frac{\vdash \Gamma}{\vdash \Gamma'}\text{ exchange } (\Gamma' \text{ is a permutation of } \Gamma)$$

Logical Rules:

$$\frac{}{\vdash 1^i}\ \text{one} \qquad \frac{\vdash \Gamma}{\vdash \Gamma, \bot^i}\ \text{false}$$

$$\frac{\vdash \Gamma, A^i \qquad \vdash B^i, \Delta}{\vdash \Gamma, (A \otimes B)^i, \Delta}\ \text{times} \qquad \frac{\vdash \Gamma, A^i, B^i}{\vdash \Gamma, (A \parr B)^i}\ \text{par}$$

(no rule for zero)

$$\frac{}{\vdash \Gamma, \top^i}\ \text{true} \qquad \frac{\vdash \Gamma, A^i}{\vdash \Gamma, (A \oplus B)^i}\ \text{left plus}$$

$$\frac{\vdash \Gamma, A^i \qquad \vdash \Gamma, B^i}{\vdash \Gamma, (A \& B)^i}\ \text{with} \qquad \frac{\vdash \Gamma, B^i}{\vdash \Gamma, (A \oplus B)^i}\ \text{right plus}$$

$$\frac{\vdash ?\Gamma, A^i}{\vdash ?\Gamma, !A^i}\ \text{promotion} \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?A^i}\ \text{weak}$$

$$\frac{\vdash \Gamma, A^i}{\vdash \Gamma, ?A^i}\ \text{deriliction} \qquad \frac{\vdash \Gamma, ?A^i, ?A^i}{\vdash \Gamma, ?A^i}\ \text{contraction}$$

$$\frac{\vdash \Gamma, A^i}{\vdash \Gamma, \forall x.A^i}\ \text{for all (x not free in } \Gamma) \qquad \frac{\vdash \Gamma, A^i[t/x]}{\vdash \Gamma, \exists x.A^i}\ \text{exists}$$

$$\frac{\Gamma, A^{i+1}}{\Gamma, \S A^i}\ \text{paragraph}$$

Some things to notice: The cut, as well as other logical rules must take place between two formulas of the same level. The paragraph modality can decrease the depth, and is the only rule that changes the index of a formula.

There is a new cut elimination transformation to deal with the new modality

$$\frac{\dfrac{\Gamma, A^{i+1}}{\Gamma, \S A^i}\ \text{paragraph} \qquad \dfrac{\Delta, A^{\perp^{i+1}}}{\Delta, \S A^{\perp^i}}\ \text{paragraph}}{\Gamma, \Delta}\ \text{cut} \quad \rightsquigarrow \quad \frac{\Gamma, A^{i+1} \qquad \Delta, A^{\perp^{i+1}}}{\Gamma, \Delta}\ \text{cut}$$

One last modification is to restrict the collection of valid proofs to only those that have all formulas of the same level in the conclusion, this ensures that $\vdash A, B$ is provable iff $\vdash A \parr B$ is provable.

3.2.3 Basic Results

The system enjoys cut elimination.

There is a $(\mathbb{Z}, +)$ action on formulas $k \bullet A^i \mapsto A^{i+k}$ and proofs $k \bullet \pi \mapsto$ the same proof with all formulas acted on by k.

Lemma. $\vdash \Gamma$ *is derivable iff* $\vdash k \bullet \Gamma$ *is derivable*

Hence it is not the level that matters but the relative level within a proof.

There is a proof net formalism with all the usual properties and there is a categorical construction that can be given to any model of LL to make it a model of $LL_\S$

Inclusion of Other Logics

There are several other LL variants that can be easily shown to sit inside $LL_\S$. A list of the ones mentioned in [**?**] are LLL, ELL, L^3, L^4 and obviously full second order LL.

3.3.1 LL

Simply remove the paragraph deduction rule and you have second order LL.

3.3.2 LLL

Light Linear Logic, Girards logic of polytime, is given when we restric to the fragment where stratification corresponds to exponential depth (where depth of exponentials is explained in [**?**]).

Make this explicit and precise. Connect to Asperti and Teruis LALL and Mazzas L^4 which are all polytime too

3.3.3 ELL

Girards system for Elementary (towers of exponentials) time computations. This is given by restricting to sequents of the form $\vdash \S\Delta^i, \Gamma^{i+1}$, where there are no paragraphs in Γ.

3.3.4 L^3

The subsystem of Baillot and Mazzas Linear Logic by Levels that captures elementary functions. We simply require that every exponential is preceeded by a §. The paper then goes on to give several different classifications of L^3 [**?**]. Can we extend them to L4?

Interactive

Geometric

Semantic

Denotational Semantics

References are mainly [**?**], [**?**], [**?**], [**?**], [**?**].

The Idea of Semantics

The idea of semantics is that of creating invariants. It allows us to distinguish the potentially complicated objects of the logic in a framework where there are more possible moves. Another possible motivation is that the things that a natural semantics does and does not see may be reasonable candidates for notions of equality between proofs / programs

Heaven LOGIC PROOF CONCEPT

The rules of correct reason

A use of the rules to reason

The essential structure of the proof

Earth

Syntax

LL

DiLL

$\lambda - Calculus$

Proofs

$\pi, \ \pi'$

Programs

Equivilence Classes of Proofs

α

β

cut elimination

Complexity class

$[\![-]\!]^{-1}$

Models

contexts where proofs and programs are "essentially" the same

4.1.1 Model Theory

> There is nothing at the bottom of mathematics but philosophy
>
> *Pepijn Schmitz*
> *Stack Exchange (second link)*

This is a part of logic and set theory that has never made sense to me and as such I want to go through some materials with a philosophical and sceptical eye. I will attempt to note down any epiphanies on the logical validity of this area of inquiry as I see it.

Ok so i read some stack exchange posts lol, but there was some wisdom there:

```
https://math.stackexchange.com/questions/4091198/represent-the-definition-of-elementary-substructure-in-fol
https://math.stackexchange.com/questions/1334678/does-mathematics-become-circular-at-the-bottom-what-is-at-the-bottom-of-mathema
https://math.stackexchange.com/questions/173735/how-to-avoid-perceived-circularity-when-defining-a-formal-language
https://mathoverflow.net/questions/47399/dont-the-axioms-of-set-theory-implicitly-assume-numbers
https://math.stackexchange.com/questions/201703/an-apparently-vicious-circle-in-logic
https://math.stackexchange.com/questions/121128/when-does-the-set-enter-set-theory
```

I want to summarise what I take to be the main take away from these discussions. There seem to be a few different copes when it comes to foundations. Clearly the concepts that we are attempting to make rigerous are employed in the definitions, we look at sets when defining logic, number and logic when defining set, set and logic when defining number etc. These concepts appear all to readily to our mathematical imagination and it is easy to gloss over them without engaging seriously in the apparent circularity. The main "escapes" from these circles seem to be

- The Meta-Theory: You bring something to the game and only understand or talk about a system within another. These meta systems will be of varying formality. For instance we may have an "intuitive" set theory (and number theory) that we use to build the concept of FOL and then we use this combined system to formalise ZFC. The meta theory is always (necissarily) more powerful than the object language, because it must contain the object language.

  Even more starkly to begin with any kind of discussion of language we need notions of repetition, sameness, character, shape etc

  Again we must bring to the table more than we are able to formalise. Essentially all discussions take place within a metalanguage and this is what allows us to move forward, taking the metalanguage as given we formalise within it. What lies at the bottom must be the total language, a language that contains all others within it. If I dare to apply my regular concepts to such an (imaginary) object then I would see that it is necissarily inconsistent and therefore we are reasoning about our systems inside a meta theory that is ultimately inconsistent. Does this make the endeavour meaningless? It seems I must assume everything to get something. Well I would say that I dont need to actually take for granted the entire (inconsistent) metatheory but only a fragment.

This is often done by developing ZFC inside a casual ZFC

- Everything is Logic: ZFC can be formalised as a first order language purely as strings, axioms and inference rules (as opposed to thinking about a collection of axioms in a language generated as a first order theory (using logic to give inference rules etc)). It is purely its own entity. It seems that we still need the concepts inherent in writing symbols, constructing strings and changing strings. This seems to be something and so we have recourse to the above meta-theory idea, however it does seem a reduction in the conceptual bagage. We have reduced it to a physical idea.
- No Sets Foundations: When the term set is used when developing foundations and logic "there is no ontological commitment" to the sets of ZFC. There is in fact a computational / algorithmic interpretation to what is going.

> “ *But it can also be read in a different way. The definition can be used to generate a completely effective procedure that a human can carry out to tell whether an arbitrary string is a formula. In this way, we can understand inductive definitions in a completely effective way without any recourse to set theory. When someone says "Let A be a formula" they mean to consider the situation in which I have in front of me a string written on a piece of paper, which my parsing algorithm says is a correct formula. I can perform that algorithm without any knowledge of "sets" or ZFC.Another important example is "formal proofs". Again, I can treat these simply as strings to be manipulated, and I have a parsing algorithm that can tell whether a given string is a formal proof. The various syntactic metatheorems of first-order logic are also effective. For example the deduction theorem gives a direct algorithm to convert one sort of proof into another sort of proof. The algorithmic nature of these metatheorems is not always emphasized in lower-level texts - but for example it is very important in contexts like automated theorem proving.* ”

> “ *We know that 2+2=4, not because of number theory or the Peano axioms, but because that reflects our daily experience.* ”

Of course all the copes must live inside a meta theory and I have expressed somewhere in a diary somewhere a metaphysical beleif that there is inteligebility only within a meta theory, so each method is itself stated in a meta theory and comes down to a belif in a given metatheory, for example a belief in finite computation.

4.1.2 Philosophy and History

Sources:

- part D of "Philosophy and Model Theory" by Button and Walsh. [**?**]
- Internet Encyclopedia of Philosophy; Semantic Theory of Truth; Jan Woleński [**?**]
- Tarski's Truth Definitions (Stanford Encyclopedia of Philosophy) [**?**]
- What is Tarski's theory of truth?; Permalink `https://escholarship.org/uc/item/2bt294j8` [**?**]

History The notions of structure and sentence were not well defined for a good portion of the 20th century and they made free use of set theoretic concepts. (Godel 1932) made use of languages of uncountable cardinality

Model theory started largely informally in works around 1900. These ideas were first formalised in a non model theoretic way and later formalised into model theory.

Much of this formalisation was into the language of set theory. I took issue with statements of model theory saying so and so was true, but isnt this what we do when we assert a mathematical claim. No perhaps not, perhaps it is claiming that we could evince a syntactical manipulation from the axioms to the required theorem through deduction rules. The idea of truth only sits in the background making such statements meaningful..

Perhaps this endeavor is underpinned by Tarskis "Concept of Truth" (Tarski 1983, VIII), the paper and the philosophy.

Philosophy The project of model thoery seems to be underpined by Tarskis theory of truth, outlined in 1933 (1956 in English). Truth was to be defined semantically and inductively through the concept of satisfaction. This "Semantic theory of truth" was in the vein of Aristotles correspondence theory of truth.

A key criterion of a good theory of truth to Tarski was that it should include all sentences that were of the form

"x" is true iff x.

this was known as the T-scheme.

Tarski made the move to relativise the notion of truth of a sentence to a given meta-language. This ensures that sentences of a language can only be true relative to another language (reminisent of the theory of types). i.e. in the T scheme "is true" belongs to a different language than "'x'".

"Schnee ist weiss" is true iff the snow is white

is a classic example of assigning truth conditions to an object language (German) in terms of a meta-language (English). The english sentence above does not use the object sentence "Schnee ist weiss", it only mentions it. Note that ordinary languages, such as English, contain their own metalanguages, this is the property of being semantically closed.

Two criterion motivating Tarskis definition of truth were material and formal correctness. These manifested as the requirment that the definition does not imply paradoxes in the object or meta-languages (this was part of the motivation for reletivising truth) and material correctness was the motivation for including the T-scheme.

Hence in model theory (or the standard formalisation there of) we get a presentation of truth for an object language where the meta-theory is explicitly ZFC.

Important to note is that if we attempt to create a "universal" meta-language, one capable of transcending the heirarchy implicit in the object, meta-language distinction then it would be neccessarily semantically closed and hence inconsistent. For this reason we limit ourselves to not talking about truth of sentences in the meta-language. One simply assumes the intuitive concept of truth in the meta-theory.

> “*Clearly, SDT is a-criterial. This means that the definition in question does not generate any truth-criterion, although it says what truth is. If mathematics is taken into account, proof can be regarded as a measure of truth. However, there arises a problem. Let the symbol Pr denote the provability operator. By the Löb theorem, we have PrA implies A, a theorem very similar to TrA implies A. But, due to the first incompleteness theorem, the formula A implies PrA cannot be consistently added to the provability logic. Hence, there is no counterpart of the T-scheme with Pr instead Tr, that is, the scheme PrA iff A. So, we must conclude that proof is not a complete truth-criterion even in mathematics*”
>
> **Jan Woleński [?]**

This is clearly schitzophrenic. I dont care about T scheme, I care about proof. I dont care about attaining truth I care about KNOWING I have attained truth. Replace truth with proof.

There is something like the following happening. The idea is to define a predicate that is true iff the thing it is applied to is true. This can be thought of inside ZFC as attempting to FIND or show the existence of a predicate that can be used in comprehension to create a set containing only true sentences. When we write a sentence of a FOL we are implicitly asserting it, i.e. asserting its truth, so that concept is always there, now we ask is there a logical predicate that can make that distinction itself. This is the question that Tarski answered to me. Obviously "truth" is not a feature of the language (object language) it is a human concept, so there need to be criteria for when the predicate is sufficiently "like" truth. This is the role of Tarskis formal and material correctness conditions.

4.1.3 The Fear

I dont know if its in here but the big problem with semantics is that there is no a priori way, or no intrinsic way, to translate between two different syntaxes. No way to translate between subject languages without first putting them inside a meta language.

I see logic as standing on its own, a pure syntax. To "translate it" is to talk about it in a metalangauage that isnt a priori connected, i.e. The subject langugae of this meta language isnt necissarily the old logic that we wanted to talk about. In fact it cannot be because one of its properties (subject / not subject) has changed. So when we for instance instantiate a logic in ZFC and then prove things about it how can we know that what we are proving is inherent to the logic and not the translation? It may seem like trifling but it is a sincere concern that this meta-mathematical process of translating a syntactic system into another may be perverting it.

What makes a semantics of LL/DiLL

To know what a semantics of LL we must be clear what a LL is in the first place. It is atomic formulas, connectives, deduction rules and **cut elimination transformations**. Changing these things will change what it means to create a semantics. From our notes on LL we know that the units are

$$0, 1, \perp, \top$$

Unary connectives

$$!, ?$$

And binary connectives

$$\oplus, \otimes, \&, \parr$$

And these things are what remain unchanged between "different" LL's. In general however the cut elimination rules are highly idiosyncratic and it is non-trivial to compare the systems with different rules. We shall take a general cut relation then and call it $\sim$, leaving the specifics unclear for now, noting only that it is strongly normalising on proofs and results in a cut free normal form.

Now the way we want to write down a semantics will depend on the structure we are using to represent (1) The logic in mathematics (2) And the model.

For us this will take the form of categories or ZFC typically. To summarise

Representing a logic as a category: Classifying Category / Syntactic Category: We take the syntactic cateogory of the logic:

not clear on any of this, but it seems like the slickest way to formulate these things potentially

Pieces of the Logic		Interpretation in ZFC	Interpretation in Categories
Formulas	A, B, C, ...	Sets	Objects
Function symbols	0-ary: 0, 1, $\top$, $\perp$, Unary: $!, ?, (-)^{\perp}$, Biary: $\oplus, \otimes, \&, \parr$	Functions of the same arity	Functions on objects of the same arity
Proofs	π, π'	...	...
Equivilence relations on proofs	cut, α	$\pi \sim \pi' \implies [\![\pi]\!] = [\![\pi']\!]$	
Deduction Rules	...	...	...

- Objects: α equivilence classes of contexts (lists of sorts)
- Morphisms: Equivilience classes of context morphisms.
- Identity:
- Compositions:

And the fact that composition is associative and works with the identities is immediate.

A model of a the logic is then a functor out of this category.

A Simpler Approach In less generality we can first consider $LL \hookrightarrow DiLL \hookrightarrow ZFC$ by consdiering the set of proofs(modulo cut equivilence?) $\mathfrak{P}$, the set of formulas $\mathfrak{F}$ and the set of connectives $\mathfrak{R}$, then a semantics is a map (set theoretic function) sending

$$\begin{aligned} [\![]\!] : \mathfrak{F} &\to F \\ \mathfrak{P} &\to P \\ \mathfrak{R} &\to R = \{ \text{relations on elements of } F \} \end{aligned}$$

We want to capture in the semantics is how the deduction rules behave. We want the denotations to be compositional. So for each connective, r, of the logic we want a same arity function on the set A such that

$$[\![XrY]\!] = [\![X]\!][\![r]\!][\![Y]\!]$$

This is exactly the idea of a classic model theory.

I think that on formulas there is the condition that if $\Gamma \iff \Delta$ then $[\![\Gamma]\!] = [\![\Delta]\!]$? This is alluded to in Seely *aut paper and Girard saying "cannonical isos are satisfied", maybe this should be the same for only some formulas? (Categorically isomorphism not equality necissarily)

We also want a way to talk about proofs however. To each proof we also simply associate a set. Girard gives the following conditions:

- Because proofs are considered up to α and cut equivilence we also require if $\pi \sim \pi'$ then $[\![\pi]\!] = [\![\pi']\!]$, where $\sim \in \{\sim_{cut}, \sim_{\alpha}\}$. Alternitively we can replace $\mathfrak{P}$ with the set of equivilience classes and then this idea is built in already.
- The denotation should be a congruence relation in the sense that if $[\![\lambda]\!] = [\![\lambda']\!]$ then applying the same logical rule to λ, λ' to get π, π' should result in $[\![\pi]\!] = [\![\pi']\!]$

  This seems really weird, why should the denotations of formulas be entirely dependent on which logical rules are applied, surely it matters to which of the formulas they are applied? Look at my handwritten notes for a simple example that doesnt make sense to have the same denotation idk? He has some other enigmatic comments on this too.

If we restrict our set F to being a collection of objects in a fixed category and out set P to be a suitable collection of morphisms in that category then we have a categorical semantics. In this case we want the set R to be replaced not with relations but with endofunctors (or bifunctors etc).

It is natural to consider them as morphisms from their inputs to outputs however there is a technicality here, we want to send a proof of the sequent $\Gamma \vdash \Delta$ a map whose domain "corresponds" to Γ and whose codomain "corresponds" to Δ. This is why we should consider not just the set of formulas but the set of sequents or at least the set of sequences of formulas and assign denotations to this.

Ok the solution is to define using the meta-equivilence $A, B \vdash C, D \iff A \otimes B \vdash C \oplus D$

Justify this using one of the above principles, its uncomfortable becasue this is outsied the logic, its an equivilence of provability.

a proof to be a morphism

$$[\![\pi : \Gamma \vdash \Delta]\!] \in Hom(\otimes\Gamma, \oplus\Delta)$$

This is a definition, a decision and a largely ad hoc one at that.
Seely also requires that

$$[\![cut(\pi|\pi')]\!] = [\![\pi]\!] \circ [\![\pi']\!]$$

Could this be derived from the congruence concept of Girard? Its clearly kind of motivated however also somehow not from a general principle just a vibe.

The final thing to make this properly categorical is to then extend the denotations of connectives to functors instead of functions on the denotations of objects.

So to summarise: Given the following data

- Formulas
- Connectives
- Deduction rules
- Proofs
- Equivilence relations on proofs

Definition: Semantics *A semantics is a map* $[\![]\!]$ *assigning to each formula a set and each connective a relation of the same arrity such that*

$$[\![ArB]\!] = [\![A]\!][\![r]\!][\![B]\!]$$

Definition: Operational Semantics *An operational Semantics is a semantics* $[\![]\!]$ *that also assigns to each proof a set such that if*

$$\pi \sim \pi' \implies [\![\pi]\!] = [\![\pi']\!]$$

and if $[\![\lambda]\!] = [\![\lambda']\!]$ *then applying the same logical rule to* λ, λ' *to get* π, π' *then* $[\![\pi]\!] = [\![\pi']\!]$.

Definition: Categorical Operational Semantics *A categorical operational semantics is an operational semantics such that there is a category* $\mathcal{C}$ *such that* $\forall A\ [\![A]\!] \in \mathcal{C}$ *and* $\forall \pi\ [\![\pi : \Gamma \vdash \Delta]\!] \in Hom([\![\otimes\Gamma]\!], [\![\oplus\Delta]\!])$ *and* $[\![cut(\pi|\pi')]\!] = [\![\pi]\!] \circ [\![\pi']\!]$.

So we are saying that the denotations form a sort of subcategory; Objects are formulas, morphisms are proofs and composition is cutting. Note that the associativity of composition is an axiom of the category and this forces certain cut relations to be present.

Finally $[\![r]\!]$ *is a endo-functor of the appropriate arity for each relation symbol r.*

Theorem. *If* $(\mathcal{C}, \otimes, I, \multimap, \perp, !, \times)$ *is a* ***nontrivial*** *symmetric closed monoidal category with a dualising object, all products and a cotripple then there is a nontrivial categorical operational semantics*

$$[\![]\!] : \mathscr{L} \to \mathcal{C}$$

By nontrivial we mean that $|\mathcal{C}| > 1$ and $\exists A, B, C \in \mathcal{C}\ \ A \otimes B \cong C \ncong A, B$.

I dont know what im talking about but I want what these people are saying not to be meaningless and this is what im seeing

We are using the convention that the negation is a definied operation. So $A^\perp$ is variable for A a variable or defined through DeMorgan laws.

Try to prove this...

If I ever understand the classifying topos thing then check if these things can be unified etc

2-Categories It seems natural to consider a 2-category as a model of a logic with the identifications

$$Formulas \to Objects$$
$$Sequents \to Morphisms$$
$$Proofs \to 2 - Moriphisms$$

Apparently no one has done this and looking at it further the way to make sequents morphisms seems to trivialise the two category structure.

Possible Directions

- Invarience of implicit computational complexity properties under different cut elimination rules
- Link LAST and LALL into the stratified LL framework
- Abstract approach to stratification in LL [**?**] has four characterisations of L^3 (elementary time), are there similar characterisaitons of L^4 (polynomial time)
- Understanding what DiLL tells us about computation.
  - Dan claims if the derivative of a proof is zero wrt a variable then the proof "doesnt use" the input in some sense
  - It detects if a program computing a sum is really two seperate programs (by differentiating the summands by the different variables)
  - Cutting a proof against itself has a derivative of a certain kind apparently
  - Can the rate at which cut elimination expands the size of the proof be detected by DiLL

    $$\pi \rightsquigarrow_{cut} \pi' \frac{d}{dx}(\pi - \pi') = ?$$

  - What about Taylor expansions, what do they tell us? Primitive proofs? Basis of proofs?
  - There is a whole body of literature saying that DiLL is just a dress for the reesource lambda calculus.
- Give a good exposition of these 4 criteria of L^4
- Join $LL_\S$ and DiLL.
  - Abstract how the derivative should interact with the paragraph
  - Dan's suggestion is to compute derivatives of minimal examples in minimal semantics
  - Could see what the general § functor characterisation does when mixed with D from the categorical semantics of DiLL. Look at what kind of categories are models for both $LL_\S$ and DiLL.
  - Can any of the theorems about DiLL or SLL be simplified by using the tools of the other?
- Actually do some math in LAST (Terui actually shows that all the peano axioms except induction are provable, there is a modified notion of induction, so this would be the main difference, just tracking this change through PA proofs)
- Formalise the cut elimination/strong normalisation of LL in Lean
- Make a study of the different cut elimination rules (generators) and the relation that they generate. Are they all the same and if so how does this relations "uniquesness" determine the categorical structure of the semantics.

Todo list

References