Cyclic Multiplicative Proof Nets of Linear Logic with an application to Language Parsing

Roberto Maieli
joint work with V. Michele Abrusci

Dipartimento di Matematica e Fisica
Università degli Studi "Roma Tre"
maieli@mat.uniroma3.it

WoLLIC 2015 - IU, Bloomington - July, 20-23, 2015

outline

1. CyMLL proof nets (PNs), with cut-elimination and sequentialization
2. embedding Lambek Calculus in to CyMLL PNs (Lambek PNs)
3. language parsing via Lambek CyMLL PNs
4. further works

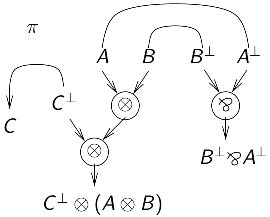
“proofs” vs “proof nets”

In his seminal article on *linear logic* (LL, 1987), Jean-Yves Girard develops two alternative notations for proofs:

- a **sequential syntax** where proofs are expressed as **derivation trees** in a sequent calculus

$$\Pi' : \dfrac{\vdash C, C^\perp \qquad \dfrac{\dfrac{\vdash A, A^\perp \qquad \vdash B, B^\perp}{\vdash A\otimes B, B^\perp, A^\perp}\otimes}{\vdash A\otimes B, B^\perp \parr A^\perp}\parr}{\vdash C, C^\perp\otimes(A\otimes B), B^\perp\parr A^\perp}\otimes \qquad \Pi'' : \dfrac{\dfrac{\vdash C, C^\perp \qquad \dfrac{\vdash A, A^\perp \qquad \vdash B, B^\perp}{\vdash A\otimes B, B^\perp, A^\perp}\otimes}{\vdash C, C^\perp\otimes(A\otimes B), B^\perp, A^\perp}\otimes}{\vdash C, C^\perp\otimes(A\otimes B), B^\perp\parr A^\perp}\parr$$

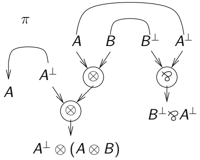
- a **parallel syntax** where proofs are expressed as **bipartite graphs** called **proof-nets**

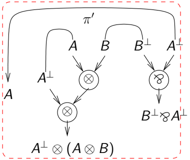


“proof nets” vs “proofs alike“

- PNs are one of the most innovative inventions of LL: they represent demonstrations in a ”geometric” (i.e., ”non inductive”) way, abstracting away from the technical bureaucracy of sequent proofs.
- PNs quotient classes of derivations that are equivalent up to some irrelevant permutations of inference rules instances.
  - while a derivation tree defines a unique proof-net, a PN may represent several derivation trees, each derivation tree witnessing a particular order of the PN sequentialization;
  - a PN requires to separate “real proofs” (*proof-nets*) from “proof alike” (*proof-structures*) using **correctness criteria**;
  - correctness criteria reveal the ”geometric” essence of the logic, beyond its ”grammatical” presentation as a sequent calculus.

$$\Pi' : \dfrac{\vdash A, A^{\perp} \qquad \dfrac{\dfrac{\vdash A, A^{\perp} \qquad \vdash B, B^{\perp}}{\vdash A \otimes B, B^{\perp}, A^{\perp}}\otimes}{\vdash A \otimes B, B^{\perp} \parr A^{\perp}}\parr}{\vdash A, A^{\perp} \otimes (A \otimes B), B^{\perp} \parr A^{\perp}}\otimes$$





the CyMLL fragment of linear logic

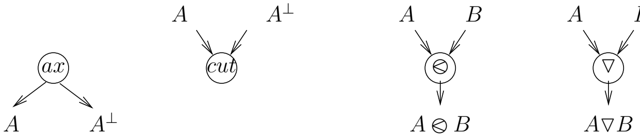
- Assume **literals** $a, a^\perp, b, b^\perp, \ldots$ with a polarity: *positive* for atoms, $a, b, \ldots$ and *negative* $a^\perp, b^\perp \ldots$ for their duals.
- A **formula** is built from literals by means of two groups of *connectives*: *negative* $\triangledown$ ("par") and *positive* $\otimes$ ("tensor").
- **De Morgan laws**: $(A \otimes B)^\perp = B^\perp \triangledown A^\perp$ and $(A \triangledown B)^\perp = B^\perp \otimes A^\perp$.
- A **CyMLL proof** is any derivation tree built by the following inference rules where sequents Γ, Δ are lists of formulas occurrences endowed with a **total cyclic order** (or **cyclic permutation**):

$$\frac{}{\vdash A, A^\perp}\,id \qquad \frac{\vdash \Gamma, A \qquad A^\perp \Delta}{\vdash \Gamma, \Delta}\,cut \qquad \frac{\vdash \Gamma, A \qquad \vdash B, \Delta}{\vdash \Gamma, A \otimes B, \Delta}\,\otimes \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \triangledown B}\,\triangledown$$

- **Negative** (or *asynchronous*) connectives correspond to *true determinism* in the way we apply bottom-up their corresponding inference rules: the application of $\triangledown$-rule is completely deterministic.
- **Positive** (or *synchronous*) connectives correspond to *true non-determinism* in the way we apply bottom-up their rules: there is no deterministic way to split the context Γ, Δ in the $\otimes$ (or cut) rule.

Definition (proof structure)

A *CyMLL proof-structure (PS)* is an oriented graph π, in which edges (resp., nodes) are labeled by formulas (resp., by connectives) of CyMLL and built by juxtaposing the below (bipartite) graphs, called **links**, in which incoming edges are called *premises* while outgoing edges are called *conclusions* of the link:



In a PS π:

- each premise of a link must be conclusion of exactly one link of π;
- each conclusion of a link must be premise of at most one link of π.

A *conclusion of* π is any outgoing edge that is not premises of any link.

In the following we characterize those CyMLL PSs that are images of CyMLL proofs: these are called **correct proof structures** or **proof nets**

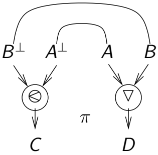
Checking Correctness of PSs

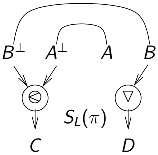
Correctness of any PS can be checked **interactively**, by “switching” (i.e. “testing”) the given PS. No need to invoke a “semantics“.

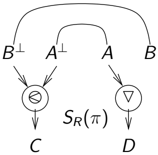
Definition (switchings)

Assume π is a CyMLL PS with conclusions Γ:

- a **Danos-Regnier switching** S for π, denoted $S(\pi)$, is the (non-oriented) graph built on nodes and edges of π with the modification that for each $\triangledown$-node we take only one premise (*left*/*right* $\triangledown$*-switch*)







Checking Correctness of PSs (continues)

"the right order of links" in a correct PS can be checked interactively.

Definition (seaweeds)

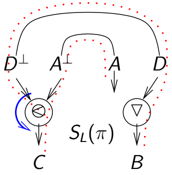
Assume π is a CyMLL PS with conclusions Γ:

- let $S(\pi)$ be an acyclic and connected switching for π;

  $S(\pi)$ is the rootless planar tree whose nodes are labeled by $\otimes$-nodes, and whose leaves $A_1, \ldots, A_n$ (with $\Gamma \subseteq A_1, \ldots, A_n$) are the terminal, i.e., pending, edges of $S(\pi)$;

  $S(\pi)$ is a ternary relation, called **seaweed**, with *support* $A_1, \ldots, A_n$; a triple (A, B, C) belongs to $S(\pi)$ iff:

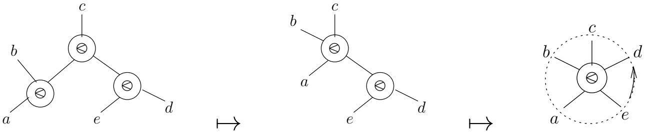
  - paths $\overline{AB}$, $\overline{BC}$ and $\overline{CA}$ intersect in the node $\otimes_i$;
  - while moving anti-clockwise around the $\otimes_i$-node, the three paths $\overline{A\otimes_i}$, $\overline{B\otimes_i}$ and $\overline{C\otimes_i}$ are in this cyclic order



Fact (seaweeds as cyclic orders)

Any seaweed $S(\pi)$ can be viewed as a cyclic total order (a cyclic, anti-reflexive, transitive and total ternary relation) on its support Γ: if a triple $(A, B, C) \in S(\pi)$, then A, B, C are in cyclic order, $A < B < C$.

Naively, we may *contract* a seaweed (by associating the $\olessthan$-nodes) until we get a single n-ary $\olessthan$-node with n incident pending edges (its support).



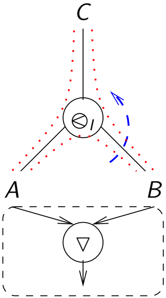
If D is an edge in $S(\pi)$, then $S_i(\pi)\downarrow^D$ is the **restriction of the seaweed** $S(\pi)$ obtained from $S(\pi)$ as follows:

1. disconnect the graph *below* (w.r.t. the orientation of π) the edge D;
2. delete the graph not containing D

Definition (CyMLL proof net)

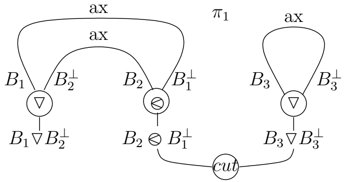
A PS π is **correct**, i.e. it is a CyMLL proof net (PN), iff:

1. π is a standard MLL PN, that is, any switching $S(\pi)$ is a connected and acyclic (**ACC**) graph (i.e., $S(\pi)$ is a seaweed);
2. for every $\triangledown$-link $\frac{A \quad B}{A \triangledown B}$, the triple (A, B, C) must occur in this cyclic order in any seaweed $S(\pi)$ restricted to A, B, i.e., $(A, B, C) \in S(\pi)\downarrow^{(A,B)}$, for every **conclusion** C of π that in the support of the restricted seaweed.

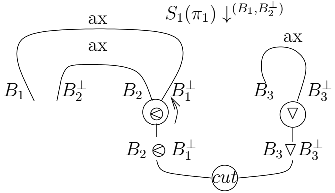


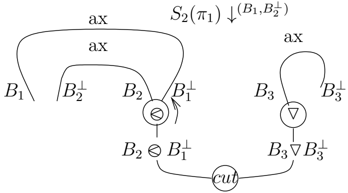
Example (correct proof structures)

the following CyMLL proof structure π_1 is correct



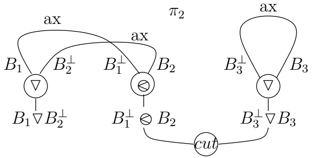
both switchings are ACC (cond. 1) and both restricted seaweeds $S_1(\pi_1)\downarrow^{(B_1,B_2^\perp)}$ and $S_2(\pi_1)\downarrow^{(B_1,B_2^\perp)}$ trivially satisfy cond. 2 (of PNs-Def.)



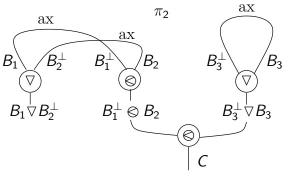


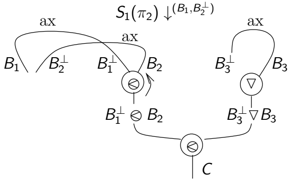
Example (more proof structures with cuts)

also the following (non-planar) proof structure π_2 is correct since both conditions, 1 and 2 (trivially), of the CyMLL PNs Definition are satisfied



By contrast, the following PS (obtained **by replacing the cut-link by a tensor $\otimes$-link**) is not correct



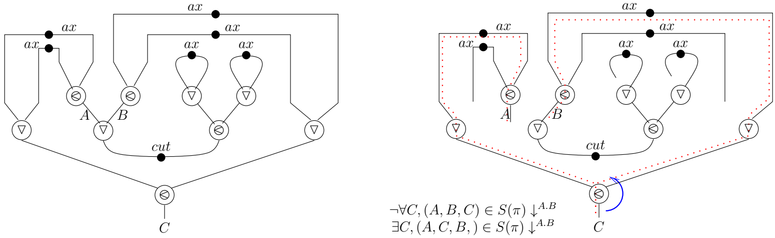


condition 2 is violated: $\exists \triangledown$-link $\dfrac{B_1 \quad B_2^\perp}{B_1 \triangledown B_2^\perp}$ and a seaweed $S_1(\pi_2)$ s.t. $\neg\forall C$ conclusion, $(B_1, B_2^\perp, C) \in S_1(\pi_2) \downarrow^{(B_1, B_2^\perp)}$.

Example (Melliès proof structure)

Unlike what happens in the commutative MLL case, the presence of cut links is "quite tricky" in the non-commutative case, since cut links are not equivalent, from a topological point of view, to tensor links: these make appear new conclusions that may disrupt the original order.

Unlike what happens with previous criteria, like Abrusci-Ruet (2000) or Pogodalla-Retoré (2004), **Melliès PS is not correct according to our correctness criterion**, since $\exists \frac{A \quad B}{A \nabla B}$ link, a seaweed $S(\pi)$ and a conclusion C s.t. $(A, C, B) \in S(\pi) \downarrow^{(A,B)}$, contradicting Cond. 2 of PN-Def.

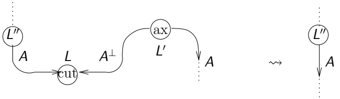


Anyway, Melliès's proof structure becomes correct after **cut reduction**.

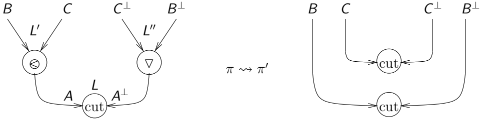
Definition (cut reduction)

Let L be a cut link in a proof net π whose premises A and $A^\perp$ are, resp., conclusions of links L', L''. Then we define the result π' (called *reductum*) of reducing this cut in π (called *redex*), as follows:

- **Ax-cut:** if L' (resp., L'') is an axiom link then π' is obtained by removing in π both formulas $A, A^\perp$ (as well as L) and giving to L'' (resp., to L') the other conclusion of L' (resp., L'') as new concl.



- $(\otimes/\triangledown)$-**cut:** if L' is a $\otimes$-link with premises B and C and L'' is a $\triangledown$-link with premises $C^\perp$ and $B^\perp$, then π' is obtained by removing in π the formulas A and $A^\perp$ as well the cut link L with L' and L'' and by adding two new cut links with premises B, $B^\perp$, resp., $C, C^\perp$

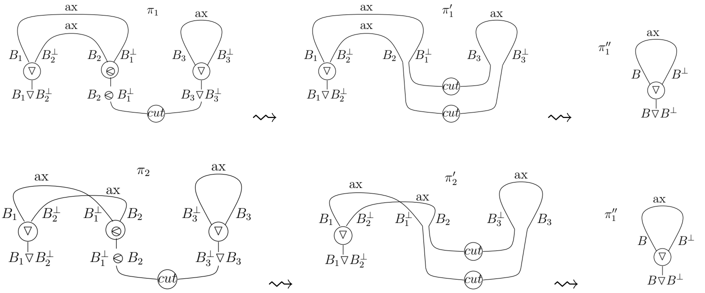


Theorem (PNs are stable under cut reduction)

If π is a CyMLL PN that reduces to π' in one step of cut reduction, $\pi \rightsquigarrow \pi'$, then π' is a CyMLL PN.

FACTS. *Cut reduction is trivially* convergent *(i.e., terminating and confluent) and preserves the order conclusions of a PN.*

EXAMPLE : observe that both π_1 and π_2 (seen before) reduce to the same normal PS; each reduction step, trivially preserves the property of being a correct PS.



Proof of "stability of PNs under cut reduction".

Condition 1. (MLL-PNs) of Def. of CyMLL-PNs follows by the:
[Euler-Poicaré invariance]: in a graph $\mathcal{G}$, $(\sharp CC - \sharp Cy) = (\sharp V - \sharp E)$
Condition 2. Assume π reduces to π' after the reduction of a cut between $(X \otimes Y)$ and $(Y^\perp \nabla X^\perp)$ and assume, by absurdum, there exist a ∇-link with conclusion $A \nabla B$ s.t. the triple (A, C, B) occurs in this wrong cyclic order in a seaweed $S(\pi')$ restricted to A, B for a conclusion C, i.e.: $(A, C, B) \in S(\pi') \downarrow^{(A,B)}$.
Then, two of the three paths $A\otimes$, $B\otimes$ and $C\otimes$ must go through (i.e., they must contain) the two cut-links, $cut_1 \frac{X \quad X^\perp}{}$ and $cut_2 \frac{Y \quad Y^\perp}{}$, obtained by reduction, otherwise π would already be violating Cond. 2

This means $\exists$ a seaweed $S(\pi)$, a link $Y^\perp \nabla X^\perp$ and a triple $Y^\perp, C, X^\perp$ s.t. $(Y^\perp, C, X^\perp) \in S(\pi) \downarrow^{(Y^\perp, X^\perp)}$, violating Cond. 2 and so the correctness of π. *Remark*: since $S(\pi)$ is acyclic, deleting the subgraph "below" $Y^\perp \nabla X^\perp$ does not make disappear C. $\square$

Cyclic order conclusions of PNs

Lemma (cyclic order conclusions)

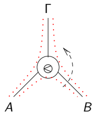
Let π be a CyMLL PN with conclusions Γ, then all seaweeds $S_i(\pi)\downarrow^{\Gamma}$ (restricted to Γ) induce the same cyclic order σ on Γ, denoted $\sigma(\Gamma)$ and called **(cyclic) order of the conclusions of** π*.*

PROOF By induction on the size $\langle \sharp V, \sharp E\rangle$ of π.

1. π is reduced to an axiom link, then obvious.
2. π contains at least a conclusion $A \triangledown B$, then $\Gamma = \Gamma', A \triangledown B$;

   by hypothesis of induction on the sub-proof net π', each $S_i(\pi')\downarrow^{(\Gamma', A, B)}$ induces the same cyclic order σ on (Γ', A, B);

   in particular, by cond. 2 of Def. of PNS, each $S(\pi')$ has this shape:

   so, by restriction $S_i(\pi')\downarrow^{(\Gamma', A)}$ (resp., $S_i(\pi')\downarrow^{(\Gamma', B)}$) and by substitution $[A/A \triangledown B]$ (resp., $[B/A \triangledown B]$) we conclude that every seaweed $S_i(\pi)\downarrow^{(\Gamma', A \triangledown B)}$ induces the same cyclic order $\sigma(\Gamma', A \triangledown B)$.

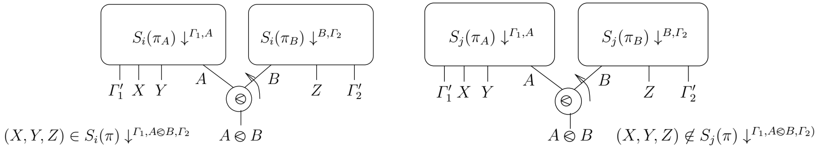
Proof of "Cyclic order conclusions Lemma" (continues).

Otherwise π must contain a terminal **splitting** $\otimes$-link $\frac{A \quad B}{A \otimes B}$ (or *cut*-link). Assume by absurdum that π is such a minimal (w.r.t. the size) PN with at least two seaweeds, $S_i(\pi)$ and $S_j(\pi)$, s.t.

$$(X, Y, Z) \in S_i(\pi) \quad \text{but} \quad (X, Y, Z) \notin S_j(\pi).$$

By **Splitting Lemma** and by Def. of Seaweed, it cannot be the case that $X \in \pi_B$, $Y \in \pi_A$ and $Z = A \otimes b$; thus, assume e.g. both X and Y belong to π_A and Z belongs to π_B and for some i, j, we have:

$$(X, Y, Z) \in S_i(\pi) \downarrow^{(\Gamma_1, A \otimes B, \Gamma_2)} \text{ and } (X, Y, Z) \notin S_j(\pi) \downarrow^{(\Gamma_1, A \otimes B, \Gamma_2)}$$



so, by restriction, $(X, Y, A) \in S_i(\pi_A) \downarrow^{\Gamma_1, A}$ and $(X, Y, A) \notin S_j(\pi_A) \downarrow^{\Gamma_1, A}$, that is absurdum, since by hypothesis of induction π_A is correct.

sequentialization of CyMLL PNs

Theorem (sequentialization of CyMLL PNs)

Any CyMLL PN with conclusions $\sigma(\Gamma)$ sequentializes *into a CyMLL sequent proof with same cyclic order conclusions* $\sigma(\Gamma)$ *and vice-versa.*

Proof.

By induction on the size $\langle \sharp Vertexes, \sharp Edges \rangle$ of π.

1. if π is an axiom link, then trivial case;
2. else, if π contains a terminal $\triangledown$-link, then we reason by induction via the Order Conclusions Lemma (seen before);
3. else, if π contains a terminal $\otimes$-link or a cut-link (π is in *splitting condition*), then we reason by induction via the Splitting Lemma.

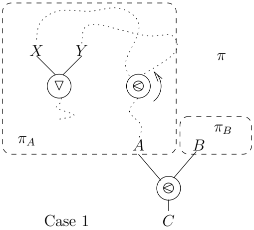
Lemma (splitting)

Let π *be a CyMLL PN with at least a* $\otimes$*-link (resp., a cut-link) and with conclusions* Γ *not containing any terminal* $\triangledown$*-link (so, we say* π *is in* splitting condition*); then, there must exist a* $\otimes$*-link* $\frac{A \quad B}{A \otimes B}$ *(resp., a cut-link* $\frac{A \quad A^{\perp}}{}$*) that splits* π *in two CyMLL PNs,* π_A *and* π_B *(resp.,* π_A *and* $\pi_{A^{\perp}}$*).*

Proof of Splitting Lemma

Assume π is a CyMLL PN in splitting condition, then by the Splitting Lemma for standard commutative MLL PNs (Girard, 1987) π must split at $\frac{A \quad B}{A \otimes B}$ in two components π_A and π_B; we know that both components satisfy Cond. 1 (they are MLL PNs).

Assume by absurdum π_A is not a CyMLL PN (violating Cond. 2 of PN-Def.); this means there exists a $\frac{X \quad Y}{X \triangledown Y}$ and a restricted seaweed $S(\pi_A)\downarrow^{(X,Y)}$ with the triple X, A, Y in the wrong order, i.e., $(X, A, Y) \in S(\pi_A)\downarrow^{(X,Y)}$



Case 1

This means there exists a restricted seaweed $S(\pi)\downarrow^{(X,Y)}$ containing X, Y and $C = A \otimes B$ in the wrong cyclic order, i.e., $(X, C, Y) \in S(\pi)\downarrow^{(X,Y)}$, contradicting the correctness of π.

CyMLL and Lambek Calculus

CyMLL can be considered as a classical extension of **Lambek Calculus** (LC, 1958) one of the ancestors of LL.

Definition (Lambek formulas and sequents)

Assume A and S are, respectively, a formula and a sequent of CyMLL.

1. A is a *(pure) Lambek formula (LF)* if it is a CyMLL formula recursively built according to the following grammar

$$A \ := \ \text{positive atoms} \mid A \otimes A \mid A^{\perp} \triangledown A \ (\equiv A \multimap A) \mid A \triangledown A^{\perp} \ (\equiv A \multimapinv A).$$

2. S is a *Lambek sequent of CyMLL* iff

$$S \ = \ (\Gamma)^{\perp}, A$$

   where A is a non void LF and $(\Gamma)^{\perp}$ is a possibly empty finite sequence of **negations of LFs** (i.e., Γ is a possibly empty sequence of LFs and $(\Gamma)^{\perp}$ is given by the negation of each formula in Γ).

3. A *(pure) Lambek proof* is any derivation built by means of the CyMLL inference rules in which premise(s) and the conclusions are Lambek sequents.

Lambek CyMLL proof nets

- The first (sound) notion of Lambek cut-free proof net, without sequentialization, was given by **Roorda** (1992).
- Then, several proposals follow, by **Retoré et alii** (1996-2004), that are stable under cut-reduction but only cut-free sequentializable (only cut-free PNs sequentialize);
- Finally a *topological correctness criterion*, proposed by **Melliès** (2004), that is both stable under cut-reduction and full sequentializable, ... but it is quite complicate!

Definition (Lambek CyMLL proof net)

A *Lambek PN* is a CyMLL PN whose edges are labeled by pure LFs or negation of pure LFs and whose conclusions is a Lambek sequent.

Corollary (stability of cut-reduction)

Cut reduction is both preserving Lambek PNs and convergent.

Theorem (full sequentialization of Lambek CyMLL PNs)

Any Lambek CyMLL proof net of $\sigma(\Gamma^{\perp}, A)$ *sequentializes into a Lambek CyMLL proof of the sequent* $\vdash \sigma(\Gamma^{\perp}, A)$ *and vice-versa.*

Parsing with Lambek Calculus

- LC represents the first attempt of **parsing as deduction**, i.e., parsing of natural language by means of a logical system.
- In LC parsing is interpreted as type checking in the form of theorem proving of Gentzen sequents.
- Types (i.e. propositional formulas) are associated to words in the lexicon; when a string $w_1 ... w_n$ is tested for grammaticality, types $t_1, ..., t_n$ are associated with these words, then parsing reduces to proving the derivability of a two-sided sequent $t_1, ..., t_n \vdash s$.
- Remind that proving a two sided Lambek derivation $t_1, ..., t_n \vdash s$ is equivalent to prove the one-sided sequent $\vdash t_n^\perp, ... t_1^\perp, s$ where $t_i^\perp$ is the dual (i.e., linear negation) of type t_i.

  In one-sided sequent calculus, phrases or sentences should be read "like in a mirror" (following opposite direction to the natural one).
- Forcing some constraints on the Exchange rule (e.g., by allowing only *cyclic permutations* over sequents of formulas) gives the required computational control needed to view theorem proving (or PN construction) as parsing in Lambek Categorial Grammar style.

main syntactical ambiguity problems with LC parsing

LC parsing presents some syntactical ambiguity problems; there may be:

(non canonical proofs) more than one (cut-free) proof for the same sequent conclusion;

(lexical polymorphism) more than one type associated with a single word.

- PNs are commonly considered an elegant solution to the first problem of representing canonical proofs; under this respect:
  - we (previously) gave an embedding of pure LC into CyMLL PNs;
  - we now show how to parse some linguistic examples by LC PNs.
- Unfortunately, there is not an equally brilliant solution to the second problem; by the way, we will sketch a possible solution (FG2015).

linguistic parsing examples, via PNs

Assume the following lexicon, where *linear implication* $\multimap$ (resp., $\circ\!-$) is traditionally used for expressing types in two-sided sequent parsing:

1. $\mathit{Sollozzo, Sam, Vito} \quad = \quad np;$
2. $$\begin{array}{lclcl} \mathit{trusts} & = & np \multimap (s \circ\!- np) & = & np^{\perp} \triangledown (s \triangledown np^{\perp}) \\ & \equiv & (np \multimap s) \circ\!- np & = & (np^{\perp} \triangledown s) \triangledown np^{\perp}; \end{array}$$
3. $\mathit{him} \quad = \quad (s \circ\!- np) \multimap s \; = \; (s \triangledown np^{\perp})^{\perp} \triangledown s \; = \; (np \otimes s^{\perp}) \triangledown s;$

Cases of lexical ambiguity follow to words with several possible formulas A and B assigned it. For example, a verb like "*to believe*" can express a relation between two persons (interpreted as np) like in S1, or between a person and a statement (interpreted as s) like in S2 or S3:

$$\textit{Sollozzo believes Vito.} \tag{1}$$

$$\textit{Sollozzo believes Vito trusts Sam.} \tag{2}$$

$$\textit{Sollozzo believes Vito trusts him.} \tag{3}$$

We can express this verb ambiguity by two lexical assignments as follows:

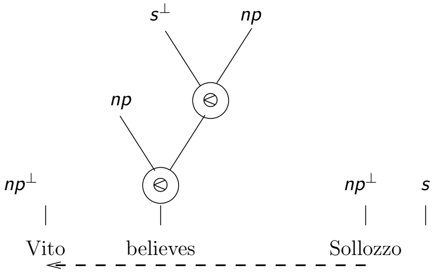
3. $\mathit{believes} = (np \multimap s) \circ\!- np \; = \; (np^{\perp} \triangledown s) \triangledown np^{\perp};$
4. $\mathit{believes} = (np \multimap s) \circ\!- s \; = \; (np^{\perp} \triangledown s) \triangledown s^{\perp}.$

parsing of S1: “Sollozzo believes Vito”

- **via derivation in the sequent calculus**:

$$\dfrac{\dfrac{}{np^{\perp}, np}\,id_1 \qquad \dfrac{\dfrac{}{s^{\perp}, s}\,id_2 \qquad \dfrac{}{np, np^{\perp}}\,id_3}{s^{\perp} \olessthan np, np^{\perp}, s}\,\olessthan}{np^{\perp}, np \olessthan (s^{\perp} \olessthan np), np^{\perp}, s}\,\olessthan$$

- **via proof net construction**: we start with the formula tree of each conclusion (no matter the order!) including s (type for sentence)

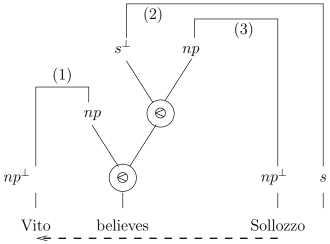


parsing of S1: “Sollozzo believes Vito” (continues)

- via derivation in the sequent calculus:

$$\dfrac{\dfrac{}{np^{\perp}, np}\,id_1 \qquad \dfrac{\dfrac{}{s^{\perp}, s}\,id_2 \qquad \dfrac{}{np, np^{\perp}}\,id_3}{s^{\perp} \olessthan np, np^{\perp}, s}\,\olessthan}{np^{\perp}, np \olessthan (s^{\perp} \olessthan np), np^{\perp}, s}\,\olessthan$$

- via proof net construction:
  – we start with the formula tree of each conclusions
  – then we “incrementally put” the axiom links on the top.

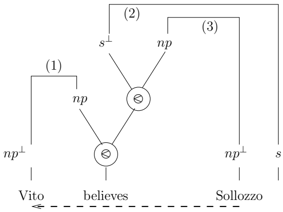


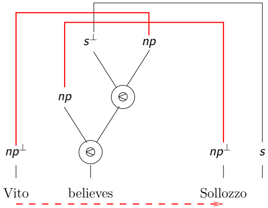
parsing of S1: “Sollozzo believes Vito” (continues)

- via derivation in the sequent calculus:

$$
\dfrac{\dfrac{}{np^{\perp}, np}\,id_1 \qquad \dfrac{\dfrac{}{s^{\perp}, s}\,id_2 \qquad \dfrac{}{np, np^{\perp}}\,id_3}{s^{\perp} \otimes np, np^{\perp}, s}\,\otimes}{np^{\perp}, np \otimes (s^{\perp} \otimes np), np^{\perp}, s}\,\otimes
$$

- via proof net construction:
  – we start with the formula tree of each conclusions
  – then we “incrementally put” the axiom links on the top.



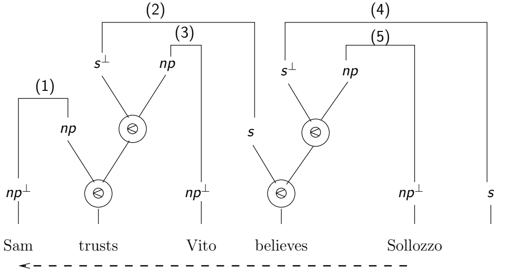


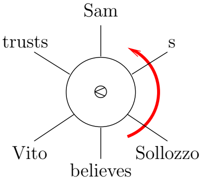
Remarks

there are two ways of linking dual pairs of literals ($np, np^{\perp}$) both of them leading to correct PNs; but only one of them corresponds to S1.

parsing of S2: “Sollozzo believes Vito trusts Sam”

$$\dfrac{\dfrac{\dfrac{}{np^{\perp}, np}\,id_1 \qquad \dfrac{\dfrac{}{s, s^{\perp}}\,id_2 \qquad \dfrac{}{np, np^{\perp}}\,id_3}{s^{\perp} \olessthan np, np^{\perp}, s}\,\olessthan}{np^{\perp}, np \olessthan (s^{\perp} \olessthan np), np^{\perp}, s}\,\olessthan \qquad \dfrac{\dfrac{}{s, s^{\perp}}\,id_4 \qquad \dfrac{}{np, np^{\perp}}\,id_5}{s^{\perp} \olessthan np, np^{\perp}, s}\,\olessthan}{np^{\perp}, np \olessthan (s^{\perp} \olessthan np), np^{\perp}, s \olessthan (s^{\perp} \olessthan np), np^{\perp}, s}\,\olessthan$$

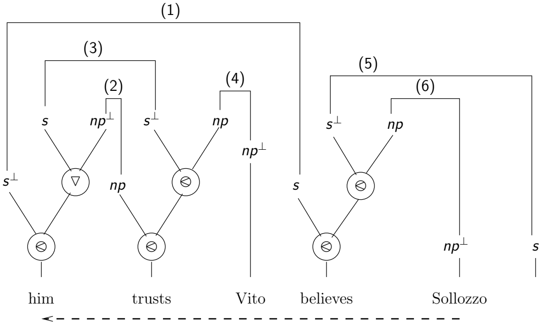




parsing of S3: “Sollozzo believes Vito trusts him”

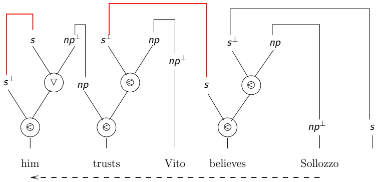
$$
\dfrac{
\dfrac{}{s^{\perp}, s}\,id_1
\qquad
\dfrac{
\dfrac{
\dfrac{}{np^{\perp}, np}\,id_2
\qquad
\dfrac{\dfrac{}{s^{\perp}, s}\,id_3 \qquad \dfrac{}{np, np^{\perp}}\,id_4}{s^{\perp} \olessthan np, np^{\perp}, s}\,\olessthan
}{s, np^{\perp}, np \olessthan (s^{\perp} \olessthan np), np^{\perp}}\,\olessthan
}{s \nabla np^{\perp}, np \olessthan (s^{\perp} \olessthan np), np^{\perp}}\,\nabla
}{s^{\perp} \olessthan (s \nabla np^{\perp}), np \olessthan (s^{\perp} \olessthan np), np^{\perp}, s}\,\olessthan
\qquad
\dfrac{\dfrac{}{s^{\perp}, s}\,id_5 \qquad \dfrac{}{np, np^{\perp}}\,id_6}{s^{\perp} \olessthan np, np^{\perp}, s}\,\olessthan
$$

$$
\dfrac{\qquad\qquad}{s^{\perp} \olessthan (s \nabla np^{\perp}), np \olessthan (s^{\perp} \olessthan np), np^{\perp}, s \olessthan (s^{\perp} \olessthan np), np^{\perp}, s}\,\olessthan
$$

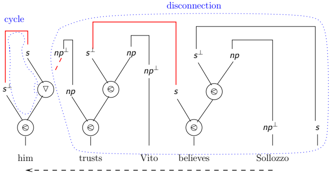


a wrong solution for “Sollozzo believes Vito trusts him”

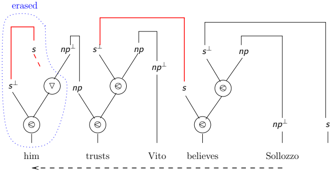
a wrong solution with an un-correct axiom linkings for literal pairs $s, s^{\perp}$



a multiplicative ∇L-switching is enough!

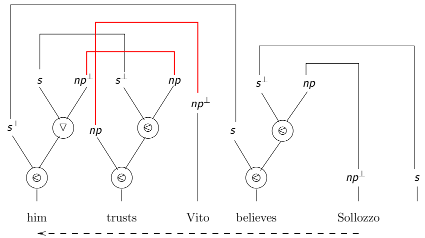


∇R-switching:

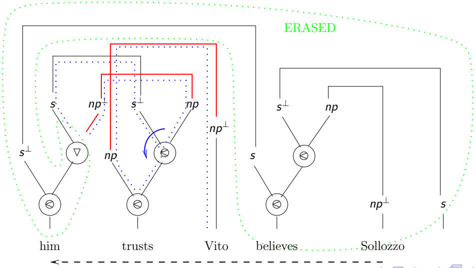


an other wrong solution: MLL switchings do not suffice!

a wrong solution with an un-correct axiom linkings for pairs $np, np^{\perp}$

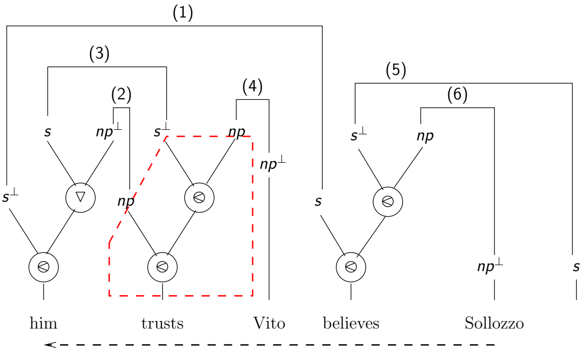


standard multiplicative switchings do not suffice;
we need a seaweed testing violation of cond. 2 on CyMLL PNs:



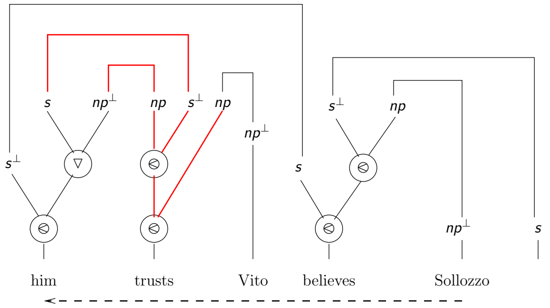
alternative parsing for “Sollozzo believes Vito trusts him”

PNs are modular! in a correct PN we can replace/interchange “modules” with same “behavior” and get still a correct PN.



alternative parsing for “Sollozzo believes Vito trusts him”

PNs are modular! in a correct PN we can replace/interchange “modules” with same “behavior” and get still a correct PN.



Here is an alternative parsing solution for sentence 3 with same matching for the axiom links but different (even though equivalent) type for the lexical item “trusts”$=np\multimap(s\circ\!\!-np) \ = \ np^\perp\triangledown(s\triangledown np^\perp)$ (*take the dual!*)

Further Works: lexical ambiguity

In order to capture lexical ambiguity we may extend Lambek PNs to the Multiplicative and Additive fragment of LL (MALL);
Ref.: Girard 1996, Hughes-van Glabbeek 2003, Maieli 2007.

Additive connectives, & and $\oplus$, allow superpositions of types;
in particular we can collapse the previous assignments 3 and 4

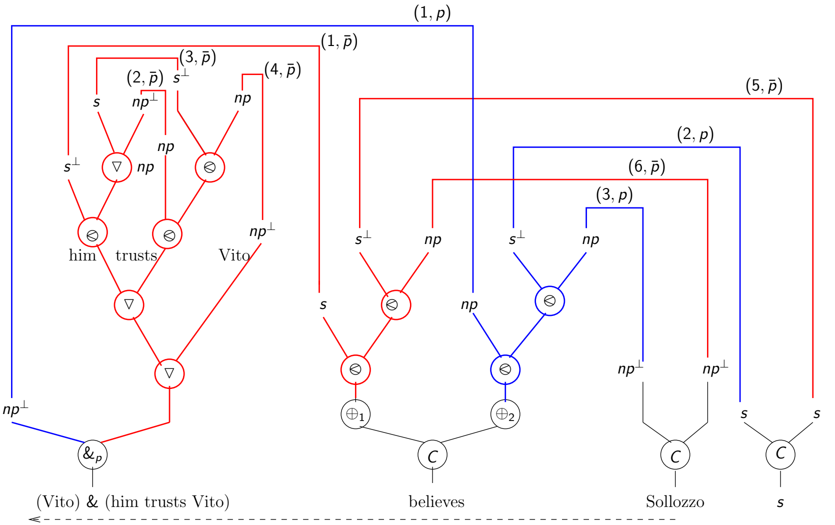
3. *believes* $= (np \multimap s) \circ\!\!- np \ = \ (np^{\perp} \nabla s) \nabla np^{\perp}$;
4. *believes* $= (np \multimap s) \circ\!\!- s \ = \ (np^{\perp} \nabla s) \nabla s^{\perp}$.

into a single additive assignment:

5. believes $= ((np \multimap s) \circ\!\!- np) \& ((np \multimap s) \circ\!\!- s) = ((np^{\perp} \nabla s) \nabla np^{\perp}) \& ((np^{\perp} \nabla s) \nabla s^{\perp})$.

and get an unique PN parsing the superposition of both sentences:
(S1) *Sollozzo believes Vito & Sollozzo believes Vito trusts him* (S3)

Further Works: lexical ambiguity



Perspective: recognizing power of Lambek Calculus

1. **Mati Pentus** proved (LICS, 1993) the *Chomsky conjecture*, that is, the languages recognized by basic Lambek Categorial Grammars (CyMLL) are precisely the Context-free ones.
2. OPEN QUESTION: in contrast, the recognizing power of CyMALL has not been precisely characterized.

   However, there is a lower bound due to **Makoto Kanazawa** who proved (JLLI, 1992) that the class of languages recognizable by the Lambek Calculus with added intersective conjunction (additive &) properly includes the class of finite intersections of CFLs.

Example

assume two CFLs

$$(a^* b^n c^n) \text{ and } (a^n b^n c^*)$$

then, their intersection is not CFL

$$(a^* b^n c^n) \cap (a^n b^n c^*) = (a^n b^n c^n)$$

Perspective: recognizing power of LC (continues)

The complexity of a logic K is the property of a *single* language associated with K. In contrast, the *recognizing power* of K has to do with a *class* of languages. Given a fixed alphabet Σ, K is said to *recognize* a language $L \subseteq \Sigma^+$ if there exist some finite subset $\mathbf{A}$ of Form_K, some $B \in \mathrm{Form}_K$, and some relation $R \subseteq \Sigma \times \mathbf{A}$ such that

$$L = \{\, w \in \Sigma^+ \mid \exists\Gamma(w\,\widetilde{R}\,\Gamma \wedge \Gamma \vdash B \in \mathrm{Prov}_K)\,\},$$

where $\widetilde{R} \subseteq \Sigma^* \times \mathrm{Form}_K^*$ is the extension of R to a relation between strings of symbols and strings of formulas:

$$\begin{aligned} w\,\widetilde{R}\,\Gamma \Leftrightarrow w = \Gamma = \epsilon \vee \\ &\exists c \in \Sigma\, \exists v \in \Sigma^*\, \exists A \in \mathrm{Form}_K\, \exists \Delta \in \mathrm{Form}_K^* \\ &(w = cv \wedge \Gamma = A\Delta \wedge c\,R\,A \wedge v\,\widetilde{R}\,\Delta). \end{aligned}$$

Then Rec_K is defined to be the class of all languages (over Σ) recognized by K.

References

- Abrusci, V. M.:
  Classical conservative extensions of Lambek calculus.
  Studia Logica vol.71(3) (2002).
- Abrusci, V. M. and Ruet, P.:
  Non-commutative logic I: the multiplicative fragment.
  Annals of Pure and Applied Logic, vol. 101(1) (2000).
- Andreoli J.-M. and Pareschi, R.:
  From Lambek Calculus to word-based parsing.
  Workshop on Substruct. Logic and Cat. Grammar, Munchen (1991).
- Danos, V. and Regnier, L.
  The Structure of Multiplicatives.
  Archive for Mathematical Logic, vol. 28 (1989)
- Girard, J.-Y.:
  Linear Logic.
  Theoretical Computer Science 50 (1987).

Girard, J.-Y.:
Proof-nets: The parallel syntax for proof-theory
Ursini, Agliano (Eds.), Logic and Algebra (1996).

Girard, J-Y.:
Le point aveugle.
Cours de Logique. Vol. I, Vers la Perfection. Paris (2006).

Hughes D., van Glabbeek, R.:
Proof Nets for Unit-free Multiplicative-Additive Linear Logic.
In: Proc. of LICS 2003, Los Alamitos (2003).

Lambek, J.:
The mathematics of sentence structure.
American Mathematical Montly, 65 (1958).

Melliès, P.-A.:
A topological correctness criterion for Mult. Non-comm. Logic.
Linear Logic in Computer Science. Cambridge Univ. Press (2004).

Maieli, R.:
A new correctness criterion for Mult. Non-comm. proof-nets.
Archive for Mathematical Logic, vol.42, Springer-Verlag (2003).

Maieli, R.:
Retractile Proof Nets of the Purely Mult. and Addit. fragment of LL.
Proc. of the 14th Int'l Conf. LPAR. LNAI 4790, Springer (2007).

Moot, R. and Retoré, Ch.:
The logic of categorial grammars.
Springer LNCS 6850 (2012).

Moot, R.:
Proof nets for linguistic analysis.
PhD thesis, Utrecht University (2002).

Pogodalla, S. and Retoré, C.:
Handsome Non-Commutative Proof-Nets.
In: Proc. of Categorial Grammars, Montpellier, France (2004).

Retoré, C.:
A semantic characterization of the correctness of a proof net.
Mathematical Structures in Computer Science, vol. 7(5), (1997).

Retoré, C.:
Calcul de Lambek et logique linéaire.
Traitement Automatique des Langues, vol. 37(2), (1996).

Roorda, D.:
Proof nets for Lambek calculus.
Journal of Logic and Computation, vol. 2(2), (1992).