LC Graphs for the Lambek calculus with product

Timothy A. D. Fowler

July 18, 2007

1 Introduction

The Lambek calculus, introduced in Lambek (1958), is a categorial grammar having two variants which will be considered in this paper: the Lambek calculus with Product ($L^\bullet$) and the Lambek calculus without Product (L)[1].

$L^\bullet$ can be characterized by the following six inference rules:

$$\frac{\Gamma \vdash \alpha \qquad \Delta\beta\Theta \vdash \gamma}{\Delta\Gamma\alpha\backslash\beta\Theta \vdash \gamma}\backslash L \qquad \frac{\alpha\Gamma \vdash \beta}{\Gamma \vdash \alpha\backslash\beta}\backslash R$$

$$\frac{\Gamma \vdash \alpha \qquad \Delta\beta\Theta \vdash \gamma}{\Delta\beta/\alpha\Gamma\Theta \vdash \gamma}/L \qquad \frac{\Gamma\alpha \vdash \beta}{\Gamma \vdash \beta/\alpha}/R$$

$$\frac{\Gamma\alpha\beta\Delta \vdash \gamma}{\Gamma\alpha\bullet\beta\Delta \vdash \gamma}\bullet L \qquad \frac{\Gamma \vdash \alpha \qquad \Delta \vdash \beta}{\Gamma\Delta \vdash \alpha\bullet\beta}\bullet R$$

Figure 1: Inference rules of the Lambek calculus.

L differs from $L^\bullet$ in that the rules $\bullet L$ and $\bullet R$ are prohibited in L.

A wide variety of work has contributed to the search for an algorithm for sequent derivability in L and $L^\bullet$ including Girard (1987), Danos and Regnier (1989), Roorda (1991), Retore (1996), Penn (2005) and Carpenter and Morrill (2005). In contrast to this work, Pentus (2006) proved that sequent derivability in $L^\bullet$ is NP-complete, effectively ending hope of a polynomial time algorithm. However, because the necessity of product for modeling natural language has not been firmly established, sequent derivability in L remains an important open problem.

This paper will extend the work by Girard and others with the intent of discovering the precise computational differences between L and $L^\bullet$ with an eye towards solving the problem of sequent derivability in L. The resultant formalism will be used to analyze the NP-completeness proof of Pentus (2006). An intuitive graphical presentation is made of Pentus' proof and we also show that the proof cannot be transformed into an NP-completeness proof for L.

The proofs in this paper are necessarily sketches. Full details are available in Fowler (2006).

[1]The original formulation of Lambek (1958) prohibited empty premises which we will not do here

2 Proof nets and graph representations

Evaluating the derivability of a sequent in the Lambek calculus has proven to be quite cumbersome and as a result most work in this area is done via proof nets.

Combinatory Categorial Grammar (see Steedman (2000)) is a prime example of a categorial grammar which does not use proof nets and Moot and Puite (2002) details a very different approach to proof nets for a multimodal extension of the Lambek Calculus which will not be considered here. We know that these systems are super-context free and that some languages are also super-context free but that the Lambek calculus is not. Despite this, the Lambek Calculus is interesting because it is simple enough that estimating numerical parameters in a probabilistic natural language system is easy, yet its parse trees differ drastically from those of similar context free grammar systems allowing for valuable comparisons.

Proof nets, as originally introduced by Girard (1987), are an extra-logical proof system which eliminates spurious ambiguity. A *proof structure* consists of a deterministic *proof frame* and a non-deterministic *axiomatic linkage*. First, all formulae in the sequent are assigned a polarity. Formulae in the antecedent are assigned negative polarity and the formula in the succedent is assigned positive polarity. The proof frame is a proof-like structure built on a sequent using the following rules:

$$
\begin{array}{cc}
\dfrac{\alpha^+ \qquad \beta^-}{\alpha\backslash\beta^-}\otimes & \dfrac{\beta^+ \qquad \alpha^-}{\alpha\backslash\beta^+}\wp \\
\dfrac{\alpha^- \qquad \beta^+}{\alpha/\beta^-}\otimes & \dfrac{\beta^- \qquad \alpha^+}{\alpha/\beta^+}\wp \\
\dfrac{\alpha^- \qquad \beta^-}{\alpha\bullet\beta^-}\wp & \dfrac{\beta^+ \qquad \alpha^+}{\alpha\bullet\beta^+}\otimes
\end{array}
$$

Figure 2: Proof frame rules

Each connective-polarity pair has a unique rule which gives us a unique proof frame for a given sequent. The top of the proof frame will consist of basic categories with polarities which are called the *axiomatic formulae*. An *axiomatic linkage* is a bijection that matches axiomatic formulae with the same basic category but opposite polarities. See figure 4 for an example of a proof structure for a sequent.

Some proof structures correspond to proofs in the Lambek calculus and those which do are called *proof nets*. It should be noted that all proof nets for the Lambek Calculus require a *planar* axiomatic linkage. A variety of methods have been introduced to determine whether a proof structure is a proof net, all of which are based on graphs. These methods fall into two major categories described in sections 2.1 and 2.2.

2.1 Girard style correctness conditions

Presentations in this style are based on the original correctness conditions given in Girard (1987). This style is characterized by building graphs based on the proof frame rules based only on whether the rule is a $\otimes$-rule or a $\wp$-rule. Work in this style includes the graphs of Danos and Regnier (1989), *R&B* graphs of Retore (1996), quantum graphs of Roorda (1991) and switch graphs of Carpenter and Morrill (2005).

Danos and Regnier (1989) were the first to formulate graph representations of proof nets in the Girard style. The *DR*-graph of a proof structure is obtained by translating each formula in the proof structure into a vertex and then inserting edges between each parent-child pair in the proof frame and between each pair of axiomatic formulae in the axiomatic linkage.

Then, a *switching* of the *DR*-graph is obtained by finding the set of all $\wp$-rules in the proof frame and deleting exactly one of the two edges between the conclusion and the two premises of each rule in the proof frame. Danos and Regnier (1989) proved that a proof structure is a proof net if and only if every switching is acyclic.

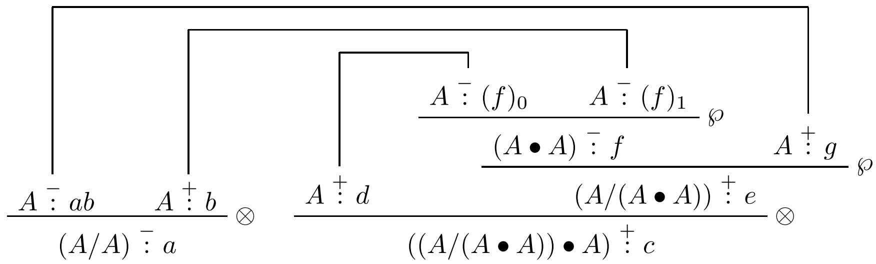
2.2 Roorda style correctness conditions

Roorda (1991) introduced a significantly different method for evaluating the correctness of proof structures. This method requires the annotation of the proof frame with lambda calculus terms as well as the creation of a set of substitutions as shown in square brackets in figure 3.

$$
\begin{array}{ll}
\dfrac{\alpha \overset{+}{:} u \qquad \beta \overset{-}{:} tu}{\alpha\backslash\beta \overset{-}{:} t} & \dfrac{\beta \overset{+}{:} v' \qquad \alpha \overset{-}{:} u}{\alpha\backslash\beta \overset{+}{:} v}\,[v := \lambda u.v'] \\
\dfrac{\alpha \overset{-}{:} tu \qquad \beta \overset{+}{:} u}{\alpha/\beta \overset{-}{:} t} & \dfrac{\beta \overset{-}{:} u \qquad \alpha \overset{+}{:} v'}{\alpha/\beta \overset{+}{:} v}\,[v := \lambda u.v'] \\
\dfrac{\alpha \overset{-}{:} (t)_0 \qquad \beta \overset{-}{:} (t)_1}{\alpha\bullet\beta \overset{-}{:} t} & \dfrac{\beta \overset{+}{:} v' \qquad \alpha \overset{+}{:} v''}{\alpha\bullet\beta \overset{+}{:} v}\,[v := \langle v', v''\rangle]
\end{array}
$$

Figure 3: Annotation of lambda terms and substitutions.

In addition to the substitutions specified above, for each pair $\langle X \overset{+}{:} \alpha, X \overset{-}{:} \Delta\rangle$ in the axiomatic linkage, we add a substitution of the form $[\alpha := \Delta]$.



Substitutions $:= [c := \langle e, d\rangle], [e := \lambda f.g], [g := ab], [b := (f)_1], [d := (f)_0]$

Figure 4: A proof structure for $(A/A) \vdash ((A/(A \bullet A)) \bullet A)$ with annotations.

Roorda (1991) then provides a method for determining proof structure correctness based on variable substitutions for both L and $L^\bullet$. Penn (2005) provides a graph representation of this method for L as follows. Construct a graph by creating a vertex for each lambda variable occurring in the proof frame. Then, introduce a directed edge from the lambda variable on the left of a substitution to the lambda variables on the right of the substitution.

Penn (2005) then gives the following correctness conditions for these LC graphs:

- **I(1)** There is a unique node s in G with in-degree 0 such that for all $v \in V$, $s \rightsquigarrow$[2] v.
- **I(2)** G is acyclic.
- **I(3)** For every substitution of the form $[v := \lambda u.v']$, $v' \rightsquigarrow u$.

2.3 Evaluation of Girard and Roorda style correctness conditions

Given our goal of investigating the difference between L and $L^\bullet$, we must evaluate the two correctness styles.

The Girard style conditions have the advantage that they have been defined for both L and $L^\bullet$ but the significant disadvantage that by virtue of the fact that it ignores the differences among $\otimes$ rules and among $\wp$ rules, removing product does not simplify the complexity of these conditions.

On the other hand, the Roorda style conditions do become simplified with the removal of product, given that projections and pairings are removed. However, no graph formalism has been introduced for $L^\bullet$ in this style until now.

3 LC Graphs for $L^\bullet$

We will construct our LC graph for $L^\bullet$ sequents in exactly the same way as for L with the obvious difference that we will have the two $\bullet$ rules and the substitutions associated with the positive $\bullet$ rule. It turns out that this is all that is necessary to accommodate $L^\bullet$. Then, we will add the following correctness condition.

- **I(4)** For every substitution of the form $[v := \lambda u.v']$ and for every $x \in V$, either every path from x to u contains v or $v \rightsquigarrow x$.

We can prove that these correctness conditions are sound and complete relative to the correctness conditions for variable substitutions in Roorda (1991) in a very similar way to the proofs for LC graphs in Penn (2005). Most proofs follow from the close mirroring between the correctness conditions for LC graphs

[2] $\rightsquigarrow$ denotes path accessibility

and the correctness conditions for variable substitutions. To prove that $I(1)$ is necessary requires an application of structural induction and some facts about projections in the lambda calculus. Details of these proofs can be found in Fowler (2006).

It is important to notice that the only difference between LC graphs for L and those for $L^\bullet$ is a single correctness condition. This simple difference does not appear in the treatment of proof nets in Roorda (1991) with the result that we now have a new tool for examining how different the two calculi are in terms of their parsing complexity.

Figure 4 shows a proof structure for a sequent which is potentially in $L^\bullet$. Figure 5 shows the corresponding LC graph. The path from d to f violates $I(4)$ causing this proof structure to not qualify as a proof net.

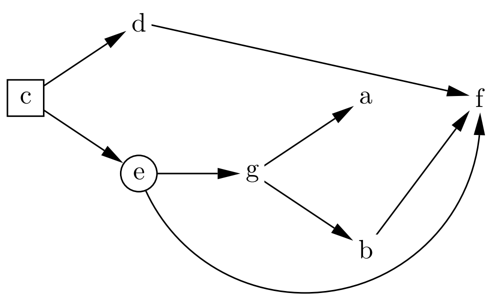


Figure 5: The LC Graph for the proof structure in figure 4.

4 The NP Completeness proof

Pentus (2006) showed that derivability in $L^\bullet$ is NP-complete and we wish to analyze that proof using LC graphs to determine whether it can be adapted to an NP-completeness proof for derivability in L.

Given a SAT instance $c_1 \wedge \ldots \wedge c_m$, Pentus (2006) introduced the following categories:

$$E_i^0(t) = p_{i-1}^0 \backslash p_i^0 \text{ for } t \in \{0,1\} \quad (1)$$

$$E_i^j(t) = (p_{i-1}^j \backslash E_i^{j-1}(t)) \bullet p_i^j \text{ if } \neg_t x_i{}^3 \in c_j \quad (2)$$

$$E_i^j(t) = p_{i-1}^j \backslash (E_i^{j-1}(t) \bullet p_i^j) \text{ otherwise} \quad (3)$$

$$F_i = (E_i^m(1)/H_i^m) \bullet H_i^m \bullet (H_i^m \backslash E_i^m(0)) \quad (4)$$

$$G^0 = p_0^0 \backslash p_n^0 \quad (5)$$

$$G^j = (p_0^j \backslash G^{j-1}) \bullet p_n^j \quad (6)$$

$$H_i^0 = p_{i-1}^0 \backslash p_i^0 \quad (7)$$

$$H_i^j = p_{i-1}^j \backslash (H_i^{j-1} \bullet p_i^j) \quad (8)$$

These categories are then used to construct the sequent $F_1, \ldots, F_n \vdash G^m$. Pentus (2006) then proved that $F_1, \ldots, F_n \vdash G^m$ is derivable in $L^\bullet$ if and only if $E_1(t_1), \ldots, E_n(t_n) \vdash G^m$ is derivable in $L^\bullet$ for some truth assignment $\langle t_1, \ldots t_n \rangle \in \{0,1\}^n$.

We now want to consider all possible LC graphs for $E_1(t_1), \ldots, E_n(t_n) \vdash G^m$. It can be shown that for a fixed $\langle t_1, \ldots, t_n \rangle$, there is exactly one proof structure for $E_1(t_1), \ldots, E_n(t_n) \vdash G^m$. Given the similarity of these sequents, we can show that the LC graph for an arbitrary $\langle t_1, \ldots, t_n \rangle$ is as seen in figure 6.

The LC graph is independent of $\langle t_1, \ldots, t_n \rangle$ except for an m by n chart of edges (shown as dotted lines). Then, the edge in column j, row i is *not* present if and only if $\neg_{t_i} x_i$ appears in c_j.

With a simple check, we can see that no proof structure for this sequent can ever violate $I(1)$, $I(2)$ or $I(3)$ by checking its LC graph.

Since $\neg_{t_i} x_i$ is present in c_j if and only if the presence of that variable causes c_j to be true for truth assignment $\langle t_1, \ldots, t_n \rangle$, all of the edges in a column are present if and only if c_j is necessarily false under $\langle t_1, \ldots, t_n \rangle$. As can be seen this occurs if and only if an $I(4)$ violation is caused by the path from b^j to d^j.

With this result, we can see that not only is $I(4)$ an important part of Pentus' NP-completeness proof, but that it is the only correctness condition with any influence on the derivability of $F_1, \ldots, F_n \vDash G^m$ and consequently on the satisfiability of $c_1 \wedge \ldots \wedge c_m$.

This also leads us to the inevitable conclusion that this proof cannot be transformed into a similar proof for L because of its absolute dependence on $I(4)$ which is not present in LC graphs for L.

For the details behind the LC graphs for these sequents see Fowler (2006).

[3] $\neg_1 v$ is a shorthand for v and $\neg_0 v$ is a shorthand for $\neg v$.

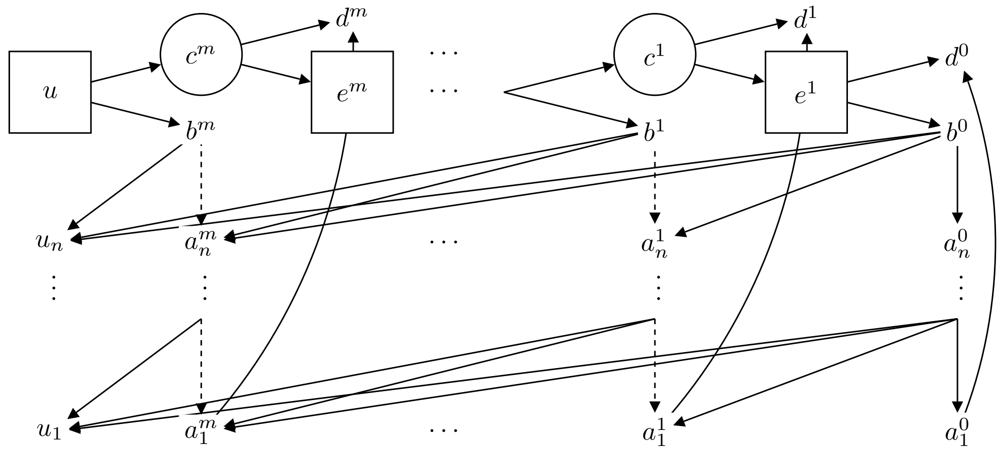


Figure 6: LC graph of $E_1(t_1), \ldots, E_n(t_n) \vdash G$.

5 Conclusion

Having introduced LC graphs for $L^\bullet$, comparing them with LC graphs for L reveals that the difference is only a single path condition on certain vertices in the graph. Furthermore, we can see by applying this observation to the NP-completeness proof of Pentus (2006) that this path condition is absolutely essential to that proof. This has given us a graphical insight into the precise differences between L and $L^\bullet$ as well as having shown that manipulating the proof of Pentus (2006) will likely be impossible to prove NP-completeness for L.

References

Bob Carpenter and Glyn Morrill. Switch graphs for parsing type logical grammars. *Proceedings of IWPT*, 2005.

Vincent Danos and Laurent Regnier. The Structure of Multiplicatives. *Archive for Mathematical Logic*, 28:181–203, 1989.

Timothy Fowler. A graph formalism for proofs in the Lambek calculus with product. Master's thesis, University of Toronto, 2006.

Jean-Yves Girard. Linear Logic. *Theoretical Computer Science*, 50:1–102, 1987.

Joachim Lambek. The mathematics of sentence structure. *American Mathematical Monthly*, 65: 154–170, 1958.

R. Moot and Q. Puite. Proof Nets for the Multimodal Lambek Calculus. *Studia Logica*, 71(3):415–442, 2002.

Gerald Penn. A Graph-Theoretic Approach to Sequent Derivability in the Lambek Calculus. *ENTCS*, 53:274–295, 2005.

Mati Pentus. Lambek calculus is NP-complete. *Theoretical Computer Science*, 357:186–201, 2006.

Christian Retore. Perfect matchings and series-parallel graphs: multiplicatives proof nets as R&B-graphs. In M. Okada & A. Scedrov J.-Y. Girard, editor, *Linear '96*, volume 3 of *Electronic Notes in Theoretical Science*. Elsevier, http://www.elsevier.nl/, 1996.

Dirk Roorda. *Resource Logics: Proof-theoretical Investigations*. PhD thesis, Universiteit van Amsterdam, 1991.

Mark Steedman. *The Syntactic Process*. The MIT Press, 2000.