Proof Nets for Basic Discontinuity*

Glyn Morrill
Universitat Politècnica de Catalunya
morrill@lsi.upc.edu
wwwlsi.upc.edumorrill/

Mario Fadda
Universitat Politècnica de Catalunya
mfadda@lsi.upc.edu

Abstract
The theory of continuity based on the Lambek Calculus is well-developed, but we need a compatible extension to include discontinuity. Earlier work set out ingredients: hypersequent calculus and proof nets expanded with parameter edges. This paper completes a preliminary line by finalising proof nets for basic discontinuity (that with one point of discontinuity) and proving correctness with respect to hypersequent calculus.

1 Motivation
Since Pentus [14] proved the completeness of the calculus of Lambek [6] with respect to free semigroups, the Lambek calculus can lay a seemingly unassailable claim to be the logic of concatenation. And where syntactic structure is taken to be the concrete geometrical representation of the essential structure wherein an expression is deemed to be grammatical, Girard’s proof nets ([4]) for the Lambek calculus (Roorda [16]), for their parsimony and economy, can lay a seemingly unassailable claim to be the syntactic structures of the Lambek calculus. Furthermore, Morrill’s [11] processing model of parsing as the incremental construction of proof nets provides a complexity metric concordant with a range of performance phenomena (see also Johnson [5]). Thus the questions of the logic, structure and processing of concatenation in categorial grammar appear to be essentially resolved.

However, natural grammar is not purely concatenative: it includes discontinuous phenomena. Starting with Moortgat [7], the search has been on to find for discontinuity what the Lambek calculus provides for continuity. One possible solution comes in the form of the generalised discontinuity of Morrill [12]. This extends the basic discontinuity of [13] by lifting the upper bound on the number of points of discontinuity. The question arises as to how to formulate proof nets for discontinuity; in particular, not proof nets based on procedural graph rewriting ([15]), which do not sit well with the incremental performance

*Thanks to Bob Carpenter and Oriol Valentín for comments and suggestions. All errors are our own. Work supported by CICYT project TIC2002-04019-C03-01.
model, but rather proof nets based on a declarative, intrinsic, correctness criterion, for which the incremental performance model applies; the present proposal appears to succeed in this respect.

The concatenative Lambek proof nets are planar; proof nets for discontinuity must be partially non-planar, but it is quite unclear how to tune degrees of non-planarity. Morrill [10] formulates proof nets expanded with parameter edges for discontinuity in which planarity is not a correctness condition, but is meant to be entailed in the concatenative case; that proposal was conjectural and correctness was not proved. In this paper we modify the proposal of proof nets with parameter edges for basic discontinuity and prove correctness with respect to a sequent calculus.

2 Basic discontinuity

Let there be a vocabulary $V$ which is a set, and a separator $\mathbf{1}, 1 \notin V$. The basic discontinuity prosodic structure induced by the vocabulary and the separator is the two-sorted algebra $(L_0, L_1, +, W)$ where $L_0 = V^*, L_1 = V^*1V^*$, $+$ is the operation of concatenation of functionality $L_0, L_0 \to L_0$ and $W$ is the operation of wrapping of functionality $L_1, L_0 \to L_0$ such that $(s_1 s_3) W s_2 = s_1 s_2 s_3$.

The types $F_0$ of sort 0 and $F_1$ of sort 1 are defined on the basis of a set $A$ of atomic types of sort 0 by:

\begin{align*}
F_0 & := A | F_0 \bullet F_0 | F_0 \setminus F_0 | F_0 / F_0 | F_1 \odot F_0 | F_1 \downarrow F_0 \\
F_1 & := F_0 / F_0
\end{align*}

Given a prosodic interpretation function $F$ mapping from $A$ into subsets of $L_0$, the denotation $[A_0] \subseteq L_0$ of types $A_0$ of sort 0 and $[A_1] \subseteq L_1$ of types $A_1$ of sort 1 are defined by:

\begin{align*}
[A] & := F(A) \\
[A \bullet B] & := \{s_1+s_2 \mid s_1 \in [A] \land s_2 \in [B]\} \\
[A \setminus C] & := \{s_2 \mid \forall s_1 \in [A], s_1+s_2 \in [C]\} \\
[C / B] & := \{s_1 \mid \forall s_2 \in [B], s_1+s_2 \in [C]\} \\
[A \odot B] & := \{s_1 W s_2 \mid s_1 \in [A] \land s_2 \in [B]\} \\
[[A, C]] & := \{s_2 \mid \forall s_1 \in [A], s_1 W s_2 \in [C]\} \\
[C \uparrow B] & := \{s_1 \mid \forall s_2 \in [B], s_1 W s_2 \in [C]\}
\end{align*}

3 Hypersequent calculus

In hypersequent calculus for discontinuity ([9]) a discontinuous type is represented by punctuated type occurrences at the distinct loci of its discontinuous segments. The configurations $O_0$ of sort 0 and $O_1$ of sort 1 are defined as follows, where $\Lambda$ is the empty configuration:

\begin{align*}
O_0 & ::= \Lambda | A_0 \mid O_0, O_0 \mid \sqrt{A_1}, O_0, \sqrt{A_1} \\
O_1 & ::= [1] \mid [1] O_0, O_1 | O_1, O_0 | \sqrt{A_1}, O_1, \sqrt{A_1}
\end{align*}

\[1\text{The configurations can be generated by an unambiguous grammar:}
\]

\begin{align*}
O_0 & ::= \Lambda | A_0, O_0 \mid \sqrt{A_1}, O_0, \sqrt{A_1}; O_0 \\
O_1 & ::= O_0, [1]; O_0 | O_1, \sqrt{A_1}, O_1, \sqrt{A_1}; O_0
\end{align*}
In configurations, all occurrences of types of sort 1 are pairwise matched. In a configuration \( O_1 \) of sort 1 there is a metalogical separator \([\) marking the point of the discontinuity. \textit{Sequents} are of the form \( O_0 \Rightarrow A_0 \) (sort 0) and \( O_1 \Rightarrow A_1 \) (sort 1).

We extend the interpretation of types to include configurations as follows, where \( \emptyset \) is the empty string:

\[
\begin{align*}
\llbracket A \rrbracket &= \{ \emptyset \} \\
\llbracket \{ \} \rrbracket &= \{ 1 \} \\
\llbracket O^{(1)}, O^{(2)} \rrbracket &= \{ s_1 s_2 \mid s_1 \in [O^{(1)}] \land s_2 \in [O^{(2)}] \} \\
\llbracket \sqrt{A}, O, \sqrt{A} \rrbracket &= \{ s_1 s_2 s_3 \mid s_1, s_3 \in [A] \land s_2 \in [O] \}
\end{align*}
\]

A sequent \( \Gamma \Rightarrow A \) is valid if and only if \([\Gamma] \subseteq [A] \) in all interpretations. The hypersequent calculus for basic discontinuity is as given in Figure 1 where \( \Gamma \) and \( \Delta \) range over arbitrary sequences of sorts 0 and 1 and punctuated sort 1 types, and separators.

(5) \textbf{Proposition} (Soundness of hypersequent calculus for basic discontinuity). If a sequent is derivable in the hypersequent calculus then it is valid.

\textbf{Proof}. Induction on the construction of hypersequent proofs. For example, to check the case corresponding to the \( \circ L \)-rule, one has to show that if \( [\Delta_1, \sqrt{A}, B, \sqrt{A}, \Delta_2] \subseteq [D] \) then \( [\Delta_1, A \circ B, \Delta_2] \subseteq [D] \). This is the case, since a simple proof by induction over the complexity of the configuration \( \Delta_1, \sqrt{A}, B, \sqrt{A}, \Delta_2 \) shows that \([\Delta_1, \sqrt{A}, B, \sqrt{A}, \Delta_2] = [\Delta_1, A \circ B, \Delta_2] \). Similarly, for the \( \circ R \)-rule one has to show that \([\Gamma_1, \Delta, \Gamma_2] \subseteq [A \circ B] \) follows from \([\Gamma_1, [\], \Gamma_2] \subseteq [A] \) and \([\Delta] \subseteq [B] \). Again, an induction over the complexity of \( \Gamma_1, [\], \Gamma_2 \) shows that \([\Gamma_1, \Delta, \Gamma_2] = \{ s_1 s_3 s_2 s_1 s_3 \in [\Gamma_1, [\], \Gamma_2], s_2 \in [\Delta] \} \), that is then contained in \( \{ s_1 s_3 s_2 s_1 s_3 \in [A] \}, s_2 \in [B] \} = [A \circ B] \). Q.E.D.

(6) \textbf{Theorem} (Cut-elimination for the hypersequent calculus for basic discontinuity). If a sequent is derivable in the hypersequent calculus then it has a Cut-free derivation.

\textbf{Proof}. Cut-elimination follows from the embedding result presented below, because the embedding can be used to translate derivations as well as sequents. A proof \( \pi \) of a hypersequent of the calculus for basic discontinuity can be translated into a proof of \textbf{MILL1}. Any occurrence of a Cut in the translation of \( \pi \) can be removed, since \textbf{MILL1} enjoys the Cut-elimination property. The Cut-free proof can be translated back into a proof of the hypersequent calculus for basic discontinuity. Q.E.D.

(7) \textbf{Corollary} (Decidability of basic discontinuity). It is decidable whether a hypersequent of basic discontinuity is a theorem.

\textbf{Proof}. By backward-chaining in the finite Cut-free hypersequent search space. Q.E.D.

---

2O. Valentin, p.c. A simple proof by induction on the complexity of configurations shows that the same semantics can be defined on the basis of the given unambiguous grammar:

\[
\begin{align*}
[\{ A \}, \Gamma_1] &= \{ s_2 s_1 \mid s_1 \in [A] \land s_2 \in [\Gamma_1] \} \\
[\sqrt{\Delta}, \Gamma_1, \sqrt{\Delta}, \Delta_1] &= \{ s_1 s_2 s_3 s_4 \mid s_1, s_2, s_3 \in [\Delta] \land s_4 \in [A] \} \\
[\Gamma_1, [\], \Delta_1] &= \{ s_3 s_2 s_1 \mid s_1 \in [\Gamma_1] \land s_2 \in [A] \land s_3 \in [\Delta_1] \} \\
[\Gamma_1, \sqrt{\Delta}, \Gamma_1, \sqrt{\Delta}, \Delta_1] &= \{ s_2 s_3 s_4 s_5 s_6 \mid s_1 \in [\Gamma_1] \land s_2 \in [A] \land s_3 \in [\Delta_1] \}
\end{align*}
\]

The application of the inductive step in the definition does not depend on the way the configuration is decomposed, because the operation of concatenation is associative.
Figure 1: Hypersequent calculus for basic discontinuity
4 Embeddings in MILL1

The hypersequent calculus for basic discontinuity can be embedded into MILL1, the multiplicative and exponential fragment of intuitionistic linear logic with first order quantification, along the lines of the embedding of the Lambek Calculus into MILL1 of Moot and Piazza [8]. In the translation, a type of sort 0 is turned into a binary predicate, and a type of sort 1 into a quaternary predicate. The arguments encode the start and end positions of segments of a type in a hypersequent.

Let us call the length of a configuration $\mathcal{O}$ the number $l(\mathcal{O})$ of type occurrences and separators that form it. To a configuration $\mathcal{O}[1]$ of sort 1 we furthermore associate the number $p\mathcal{O}$ of type occurrences that precede the separator. A sequent $\mathcal{O} \Rightarrow B$ is translated as $\|\mathcal{O}\|^\varphi \Rightarrow \|B\|^\varphi$ where for distinct constants $\epsilon_0, \ldots, \epsilon_{p\mathcal{O}}$, $\varphi = \langle \epsilon_0, \epsilon_i(\mathcal{O}) \rangle$ if the sequent is of sort 0, and $\varphi = \langle \epsilon_0, \epsilon_0, \epsilon_0, \epsilon_{i+1}, \epsilon_i(\mathcal{O}) \rangle$ if it is of sort 1. The translation of the configurations and of the types are defined by induction in Tables 1 and 2.

<table>
<thead>
<tr>
<th>configuration $\mathcal{O}$ of sort 0</th>
<th>translation $|\mathcal{O}|^\varphi$ where $l(\mathcal{O}) = j - i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Lambda$</td>
<td>$|\Lambda|^\varphi$</td>
</tr>
<tr>
<td>$\mathcal{O}_0^{(1)}, \mathcal{O}_0^{(2)}$</td>
<td>$|\mathcal{O}_0^{(1)}|^\varphi, |\mathcal{O}_0^{(2)}|^\varphi$ where $l(\mathcal{O}_0^{(1)}) = h - i$</td>
</tr>
<tr>
<td>$\sqrt{\mathcal{A}_1}, \mathcal{O}_0, \sqrt{\mathcal{A}_1}$</td>
<td>$|\mathcal{A}_1|^\varphi, |\mathcal{O}_0|^\varphi$ where $l(\mathcal{O}_0) = h - i$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>configuration $\mathcal{O}$ of sort 1</th>
<th>translation $|\mathcal{O}|^\varphi$ where $l(\mathcal{O}) + 1 = j - i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{O}_0, \mathcal{O}_1$</td>
<td>$|\mathcal{O}_0|^\varphi, |\mathcal{O}_1|^\varphi$ where $l(\mathcal{O}_0) = h - i$</td>
</tr>
<tr>
<td>$\sqrt{\mathcal{A}_1}, \mathcal{O}_1, \sqrt{\mathcal{A}_1}$</td>
<td>$|\mathcal{A}_1|^\varphi, |\mathcal{O}_1|^\varphi$ where $l(\mathcal{O}_1) = j - h$</td>
</tr>
</tbody>
</table>

Table 1: Translation of hypersequent configurations into MILL1

<table>
<thead>
<tr>
<th>Continuous Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>$|A \bullet B|^{\varphi, \psi}$</td>
</tr>
<tr>
<td>$|A / B|^{\varphi, \psi}$</td>
</tr>
<tr>
<td>$|A \div B|^{\varphi, \psi}$</td>
</tr>
</tbody>
</table>

Discontinuous Types

| $\|A \circ B\|^{\varphi, \psi}$ | $\exists x, y (\|A\|^{\varphi, \psi}, \|B\|^{\psi, \varphi})$ |
| $\|A \downarrow B\|^{\varphi, \psi}$ | $\forall x, y (\|A\|^{\varphi, \psi}, \|B\|^{\psi, \varphi})$ |

Table 2: Translation of types into MILL1
(8) **Theorem.** A sequent $\Gamma \Rightarrow B$ is a theorem of the hypersequent calculus for basic discontinuity if and only if its translation $||\Gamma||^\varphi \Rightarrow ||B||^\varphi$ is a theorem of MILL1, where $\varphi = \langle e_0, e_i(\Gamma) \rangle$ if the sequent is of sort 0, and $\varphi = \langle e_0, e_2^{\Gamma_1}, e_2^{\Gamma_2_1}, e_i(\Gamma) \rangle$ if it is of sort 1.

**Proof.** This is a straightforward generalization of the analogous result for the embedding of the Lambek Calculus into MILL1 of [8]. On the one hand, the translation of the conclusion of any hypersequent calculus rule can be derived in MILL1 from the translation of its premise(s). On the other hand, without loss of generality, in proving that the translation of a hypersequent is a theorem of MILL1, the order of application of rules can be modified so as to view the derivation in MILL1 as a translation of a derivation in the hypersequent calculus for basic discontinuity. Q.E.D.

5 **Proof nets**

There are the standard notions of polarity, logical link and proof frame as the arranged formula trees of a sequent, proof structure as a result of connecting complementary literals in a proof frame by axiom links, and proof net as a proof structure which corresponds to a sequent proof. Keeping the usual Danos-Regnier condition ([2]) for multiplicative linear validity, we use the proof nets expanded with parameter edges of [10] to encode the sublinear structure. We replace the two “resolution criteria” with a single new geometric condition of unicity, for which we prove correctness.

The proofs are based on the theory for LL1, i.e. multiplicative linear logic with first order quantification, expounded in [1], and enriched with the following result:

(9) **Definition.** Let $\Pi$ be an LL1 proof structure and $V$ a set of variables.

An $V$-path in $\Pi$ is a path that goes exclusively through nodes labelled by types containing free occurrences of all variables that appear in $V$.

(10) **Theorem.** Let $\Pi$ be a proof structure of LL1 all of the Danos-Regnier $\varphi$-switchings of which are connected and acyclic. Then the following conditions are equivalent:

1. $\Pi$ is correct with respect to $\varphi$- and $\gamma$-switchings in the sense of [1].

2. There are no $\gamma$-links with conclusions $\forall x A$ and $\forall y B$ the premisses of which can be joined by an \{x, y\}-path.

**Proof.** By induction over the size of the proof net. The two mentioned properties are preserved by removal of final $\varphi$- and $\gamma$-links. Moreover, if there are no such final links, the Splitting Lemma can be applied to $\Pi$ thought of as a proof structure of multiplicative linear logic. Again, the mentioned properties are preserved in each substructure. Q.E.D.

Consider, among all LL1 proof structures, only those that can be built unfoiling the $||\cdot||$-translation of a Lambek sequent. The unfoiling of, say, the
translation \( \langle A \bullet B \rangle \mid (x, t) \) of an input type \( A \bullet B \) will create a tree rooted by a \( \forall \)-link with conclusion \( \forall x (\langle A \rangle \mid (x, t) \mid \langle B \rangle \mid (x, t)) \) and with branches leading to the nodes \( \langle A \rangle \mid (x, t) \) and \( \langle B \rangle \mid (x, t) \). The latter types are joined by an \( \{x\}\)-path (that does not go through the premise of the \( \forall \)-link immediately below them). A simple application of the previous theorem will show that any such path does not go through the premise of another \( \forall \)-link. Furthermore, this condition, together with acyclicity and connectedness of switchings, is sufficient for the correctness of the proof structure. This gives rise to an alternative characterization of sequentializable Lambek proof structures. In addition to the usual (solid) edges (which we refer to as predicate edges), we decorate links with (nonsolid) parameter edges, see Figures 2 and 3, which are in essence trip instructions on the proof structure. To obtain a proof frame, we add moreover parameter edges joining pairwise the starts and ends of the roots as illustrated in Figure 4 for sort 0 types (these edges stand for the parameters that freeze
Figure 3: Logical proof links of the Lambek calculus and their expansions, II

\[
[A^* \Rightarrow \mathcal{V} \Rightarrow A_1^* \Rightarrow \mathcal{V} \Rightarrow A_2^* \Rightarrow \mathcal{V} \Rightarrow \cdots \Rightarrow \mathcal{V} \Rightarrow A_n^*] \\
\mathcal{V} \mathcal{V} \cdots \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V} \mathcal{V}
\]

Figure 4: Base parameter edges for a sequent of sort 0 types \( A_1, A_2, \ldots, A_n \Rightarrow A \).

the order of the types in the sequent).

(11) **Definition.** A proof structure expanded with parameter edges is *correct* if and only if the following conditions are satisfied:

1. **Danos-Regnier acyclicity.** Every predicate edge cycle crosses both edges of some \( \mathcal{V} \)-link.
2. **Unicity.** Every parameter edge cycle contains exactly one \( \mathcal{V} \).

Consider an expanded proof structure \( \Pi \) that can be sequentialized as the derivation of a sequent \( \Phi \), and let \( \Pi' \) be the proof structure associated with a derivation of the translation \( [\Phi] \). The parameter edges in Figures 2 and 3 can be joined to form parameter paths (cycles). The name is motivated by the fact that parameter paths in \( \Pi \) correspond to a way of travelling along \( \{x\} \)-paths in \( \Pi' \), where \( x \) stands for a variable occurring in \( \Pi' \).

In expanded proof nets for the Lambek Calculus, polar type trees reflect the binary relational interpretation clauses of [17] and the translation above. Each node labelled by a polar type has two incident dashed edges referred to as its start and its end parameter edges. The start comes on the left and the end comes on the right; for an output type this is reversed:

(12) start \( A^* \) end \( A^0 \) start
Figure 5: Expanded proof links for the basic discontinuity operators

These parameter edges are connected to quantifiers in the expanded proof structures which bind the parameters of types regarded as binary predicates. Extending to discontinuity, while types of sort $\emptyset$ have two incident parameter edges, types of sort 1 have four incident parameter edges, corresponding to a quaternary relational predication, notated in expanded proof nets as in (13):

(13) start$_1$ end$_2$ $A^*$ start$_2$ end$_1$ end$_1$ start$_2$ $A^o$ end$_2$ start$_1$

The subscripts refer to the first (left) and second (right) segments of a string containing a separator; note that as for types of sort $\emptyset$ the input and output orderings are mirror-images, which promotes visual symmetry. The expanded links for the discontinuity operators are given in Figure 5.

(14) **Proposition** (Completeness of expanded proof nets with respect to hypersequent calculus for basic discontinuity). If a hypersequent is derivable, there is a proof net for it.

**Proof.** Induction over the length of a derivation. Q.E.D.

(15) **Theorem** (Soundness of expanded proof nets with respect to hypersequent calculus for basic discontinuity). Every proof net corresponds to a derivable hypersequent.
Proof. Consider a proof net $\Pi$ of the discontinuity calculus. $\Pi$ can be turned into a proof structure of $\text{MILL1}$ thus:

1. Associate to each base parameter edge a distinct constant.
2. If $X$ is a root type of sort 0, replace $X$ with $[|X|][e_i \rightarrow e_j]$ where $e_i$ ($e_j$) is the constant associated to the start (end) of $X$.
3. If $X$ is a root type of sort 1, replace $X$ with $[|X|][e_i_1 \rightarrow e_i_2 \rightarrow \cdots \rightarrow e_i_h]$ where $e_i_k$ ($e_i_h$) is the constant associated with the start (end) of the $h$-th segment of $X$.
4. In logical links, replace parameter edges by quantified variables.

This yields an $\text{MILL1}$ proof structure $\Pi'$ the roots of which form a sequent $\Sigma'$ which is the translation of a sequent $\Sigma$ of the hypersequent calculus. $\Pi'$ inherits correctness from $\Pi$ because of Theorem (10). Thus $\Pi'$ corresponds to a $\text{MILL1}$ derivation of $\Sigma'$, and, because of Theorem (8) $\Pi$ can be sequentialized as a derivation of $\Sigma$. Q.E.D.

6 Examples

Figure 6 shows an example of a proof net with a discontinuous functor for the following type assignments:

(16) \text{gave}+1+\text{the}+\text{cold}+\text{shoulder} - \text{shun}: (N \downarrow S) \downarrow N
\text{John} - \ddot{j} N
\text{Mary} - \dddot{m} N

The semantic reading of the proof net according to the semantic trip (see [3, 11]) is $((\text{shun } \dddot{m}) \ddot{j})$.

Figures 7 and 8 show subject wide scope and object wide scope analyses respectively for ‘everyone loves someone’, where the quantifier words are assigned type $(S \downarrow N) \downarrow S$. The former includes parameter edges; in the latter these are omitted. The type assignments are as follows:

(17) \text{everyone} - \forall: (S \downarrow N) \downarrow S
\text{loves} - \forall: (N \downarrow S) / N
\text{someone} - \exists: (S \uparrow N) \downarrow S

The results of the semantic trips are respectively $(\forall \lambda x (\exists \lambda y ((\forall y) x)))$ and $(\exists \lambda y (\forall \lambda x ((\forall y) x)))$. 
Figure 6: Proof net for the discontinuous idiom example ‘John gave Mary the cold shoulder’ via a wrapping functor.
Figure 7: Proof net for 'everyone loves someone', subject wide scope
Figure 8: Proof net for ‘everyone loves someone’, object wide scope

References


