Automated Deduction for Categorial Grammar Logics

Richard Moot*

Abstract

In this paper we will look at automated theorem proving for categorial grammar logics. Taking linear logic as a starting point, we will show how proof nets, as an optimal proof theory for the multiplicative fragment of linear logic, can be used to generate proofs for logical statements.

We will sketch how adding structural labeling to the proof nets gives us a mechanism to account for linguistically relevant distinctions like linear order and constituency structure.

Complexity analysis will show that the derivability problem is not computationally feasible in the general case. We will alleviate this by early application of the various constraints on derivability in order to give an algorithm which is nevertheless useful in practice.

1 Introduction

The logical approach to computational linguistics states that a grammar formalism should be a logical theory. The advantage of this logical approach is that we can prove our formalism has abstract properties like consistency, which will guarantee that grammars designed in our formalism will never be degenerate ones, where we can make no distinctions between expressions at all.

If we use a logical theory as a grammar formalism, a parser should be an automated theorem prover, consisting of two components

Lexicon A (possibly nondeterministic) function l from natural language expressions to formulas in the logic.

Proof Theory A consistent set of axioms and inference rules.

Our logical theory will be *descriptively adequate* if the following holds

Definition 1 *A sequence of words $w_1, \ldots, w_n$ is a grammatical sentence if and only if $l(w_1), \ldots, l(w_n) \vdash s$ is a theorem.*

*Utrecht institute of Linguistics-OTS

2 Linear Logic

We have been intentionally vague about the nature of our proof theory, but it seems clear that global availability of the structural rules of weakening and contraction is undesirable from a linguistic point of view. The rules

$$\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\ [LC] \quad \frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\ [LW]$$

would allow us to perform the following kind of inference steps (abstracting over the actual formulas assigned to the lexical entries for the moment)

$$\frac{\text{the, man, walks, across, the, street} \vdash s}{\text{the, man, walks, across, street} \vdash s}\ [LC]$$

$$\frac{\text{the, man, walks, across, the, street} \vdash s}{\text{the, man, walks, across, the, street, spanish inquisition} \vdash s}\ [LW]$$

which are not properties we would expect our logic to have.

Linear logic was developed in [Girard 87], as a logic with *control* over the use of weakening and contraction, so this seems like a good starting point.

I will give a short introduction to linear logic, and introduce proof nets as an optimal representation of proofs in the multiplicative fragment of linear logic. Proof nets will serve as the core of our algorithm for automatic proof search, first for multiplicative linear logic and later for categorial logics.

2.1 Sequent Calculus

The axiom and cut rules are the same as for classical logic

Identity

$$\frac{}{A \vdash A}\ [Ax] \quad \frac{\Gamma, A \vdash \Delta \quad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ [Cut]$$

The structural rule of permutation is global, as it is in classical logic, and we will usually apply it implicitly. Associativity is implicit in the sequent list notation. For our linguistic endeavors, neither a global rule of associativity nor a global rule of permutation seems desirable. In section 3 we will remedy these problems in a modular fashion by using labeling.

The rules of contraction and weakening, instead of being available globally, are restricted to formulas of the form $!A$ on the left hand side and to formulas of the form $?A$ on the right hand side of the sequents. The connectives ? (why not) and ! (bang) have their own left and right rules to allow interaction between formulas which allow contraction and weakening and formulas which don't. The idea is that on the left hand side $!A$ should be interpreted as an arbitrary number of occurrences of the formula A (we choose how many), whereas $?A$ specifies an unknown quantity of occurrences of the formula A.

Under this interpretation, the left rule for ! specifies that if something is derivable with a single formula A then it is also derivable if we are allowed to use an arbitrary number of occurrences of A.

The left rule for ? would then indicate that we could accommodate for an unknown quantity of A formulas only if all context formulas can be used as many times as necessary.

Structural Rules

$$\frac{\Gamma, B, A, \Gamma' \vdash \Delta}{\Gamma, A, B, \Gamma' \vdash \Delta}\,[LP] \qquad \frac{\Gamma \vdash \Delta, B, A, \Delta'}{\Gamma \vdash \Delta, A, B, \Delta'}\,[RP]$$

$$\frac{\Gamma, !A, !A \vdash \Delta}{\Gamma, !A \vdash \Delta}\,[LC] \qquad \frac{\Gamma \vdash ?A, ?A, \Delta}{\Gamma \vdash ?A, \Delta}\,[RC]$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, !A \vdash \Delta}\,[LW] \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash ?A, \Delta}\,[RW]$$

Exponentials

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, !A \vdash \Delta}\,[L!] \qquad \frac{!\Gamma \vdash A, ?\Delta}{!\Gamma \vdash !A, ?\Delta}\,[R!]$$

$$\frac{!\Gamma, A \vdash ?\Delta}{!\Gamma, ?A \vdash ?\Delta}\,[L?] \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash ?A, \Delta}\,[R?]$$

Linear negation $(.)^{\perp}$ has the obvious rules.

Negation

$$\frac{\Gamma \vdash A, \Delta}{\Gamma, A^{\perp} \vdash \Delta}\,[L\perp] \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash A^{\perp}, \Delta}\,[R\perp]$$

For the multiplicative conjunction $\otimes$ (tensor) and disjunction $\parr$ (par) the context of the conclusion is the union of the contexts of the premisses of the rule. A way to look at a formula $A \otimes B$ is that it gives *both* an A and a B resource. It is difficult to come up with a similar intuition for $\parr$ except perhaps by noting it is the De Morgan dual of $\otimes$.

Linear implication $\multimap$ is not a primitive connective of classical linear logic, and is defined as $A \multimap B =_{def} A^{\perp} \parr B$. For completeness, I will present the rules for linear implication along with the other multiplicatives. In an intuitionistic setting $\multimap$ will replace $\parr$, as par makes essential use of multiple formulas on the right hand side of the sequent.

Multiplicatives

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \otimes B \vdash \Delta}\,[L\otimes] \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \otimes B, \Delta, \Delta'}\,[R\otimes]$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \parr B \vdash \Delta, \Delta'}\,[L\parr] \qquad \frac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash A \parr B, \Delta}\,[R\parr]$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \multimap B \vdash \Delta, \Delta'}\,[L\multimap] \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \multimap B, \Delta}\,[R\multimap]$$

For the additives the contexts of the premisses of the rule must be the same. In the presence of global contraction and weakening the additive and multiplicative rules collapse. A formula $A\&B$ represents a *choice* between A or B (as opposed to $A \otimes B$, where you get stuck with both). Similarly $A \oplus B$ means it is unknown whether we have an A or a B resource.

Additives

$$\frac{\Gamma, A_i \vdash \Delta}{\Gamma, A_0 \& A_1 \vdash \Delta}\,[L\&] \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \& B, \Delta}\,[R\&]$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \oplus B \vdash \Delta}\,[L\oplus] \qquad \frac{\Gamma \vdash A_i, \Delta}{\Gamma \vdash A_0 \oplus A_1, \Delta}\,[R\oplus]$$

Example 1 *Given the following lexicon*

$$\begin{aligned} l(Bill) &= np \\ l(is) &= np^\perp \otimes (s \otimes (np^\perp \oplus (n^\perp \otimes n))^\perp) \\ l(the) &= np \otimes n^\perp \\ l(President) &= n \\ l(fat) &= n \otimes n^\perp \end{aligned}$$

we can derive 'Bill is the President' as follows

$$\dfrac{\dfrac{\dfrac{}{np \vdash np}\,[Ax]}{np, np^\perp \vdash}\,[L\perp] \qquad \dfrac{\dfrac{\dfrac{}{s \vdash s}\,[Ax] \quad \dfrac{\dfrac{}{np \vdash np}\,[Ax]}{np \vdash np \oplus (n \otimes n^\perp)}\,[R\oplus]}{(np \oplus (n \otimes n^\perp))^\perp, np \vdash}\,[L\perp] \quad \dfrac{}{n \vdash n}\,[Ax]}{s \otimes (np^\perp \oplus (n^\perp \otimes n))^\perp, np \vdash s}\,[L\otimes] \quad \dfrac{}{n, n^\perp \vdash}\,[L\perp]}{s \otimes (np^\perp \oplus (n^\perp \otimes n))^\perp, np \otimes n^\perp, n \vdash s}\,[L\otimes]}{np, np^\perp \otimes (s \otimes (np^\perp \oplus (n \otimes n^\perp))^\perp), np \otimes n^\perp, n \vdash s}\,[L\otimes]$$

2.1.1 One-sided Sequent Calculus

Proposition 1 *The following formulas are derivably equivalent.*

$$\begin{aligned} A^{\perp\perp} &= A \\ (A \otimes B)^\perp &= A^\perp \parr B^\perp \\ (A \parr B)^\perp &= A^\perp \otimes B^\perp \\ (A \& B)^\perp &= A^\perp \oplus B^\perp \\ (A \oplus B)^\perp &= A^\perp \& B^\perp \\ (!A)^\perp &= ?A^\perp \\ (?A)^\perp &= !A^\perp \end{aligned}$$

Proof Trivial. □

Because of the properties of negation given in proposition 1 (De Morgan dualities, elimination of double negation) we can restrict negation to atomic formulas, and move all formulas to the right hand side of the sequent. This is just a matter of economy: a syntactic manipulation to reduce the number of rules.

$$\dfrac{}{\vdash A, A^\perp}\,[Ax] \qquad \dfrac{\vdash \Gamma, A \quad \vdash A^\perp, \Delta}{\vdash \Gamma, \Delta}\,[Cut]$$

$$\dfrac{\vdash \Gamma, B, A, \Delta}{\vdash \Gamma, A, B, \Delta}\,[P]$$

$$\dfrac{\vdash \Gamma, ?A, ?A}{\vdash \Gamma, ?A}\,[C] \qquad \dfrac{\vdash \Gamma}{\vdash \Gamma, ?A}\,[W]$$

$$\dfrac{\vdash ?\Gamma, A}{\vdash ?\Gamma, !A}\,[!] \qquad \dfrac{\vdash \Gamma, A}{\vdash \Gamma, ?A}\,[?]$$

$$\dfrac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}\,[\otimes] \qquad \dfrac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\,[\parr]$$

$$\dfrac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \& B}\,[\&] \qquad \dfrac{\vdash \Gamma, A_i}{\vdash \Gamma, A_0 \oplus A_1}\,[\oplus]$$

We can prove the equivalence between the one-sided and two-sided sequent calculus by simple induction.

2.2 Proof Nets

Proof nets are geometric proof objects characterizable either as parallelized sequent proofs or as natural deduction proofs with multiple conclusions, and indeed they combine the best aspects of both.

The problem is that accommodating full linear logic requires the use of 'proof boxes': concessions to the sequent calculus which destroy many of the advantages of proof nets. For the current paper we will focus on the multiplicative fragment of linear logic only, though the additives may be incorporated using weighted links and formulas as in [Girard 96].

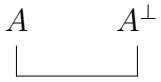
We first define a superset of proof nets, called proof structures.

Definition 2 (Proof Structure) *A* proof structure *is a collection of the following links*

Axiom

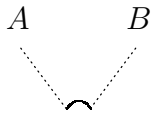
Premisses: None
Conclusions: $A, A^\perp$

Cut



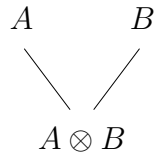
Premisses: $A, A^\perp$
Conclusions: None

Par



Premisses: A, B
Conclusion: $A \parr B$

Tensor



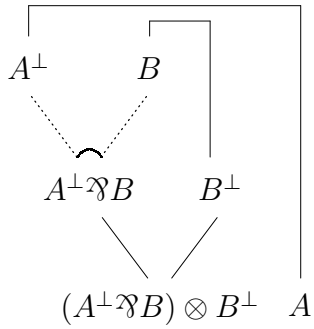
Premisses: A, B
Conclusion: $A \otimes B$

subject to the following conditions:

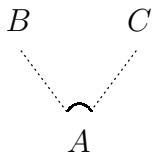
- *Each formula of the proof structure is the conclusion of one link.*
- *Each formula of the proof structure is the premiss of at most one link.*

Because proof structures don't take the difference between par and tensor into account, it should be obvious that not all proof structures are sound (i.e. correspond to a sequent proof). We need an extra soundness criterion to distinguish proof nets from other proof structures.

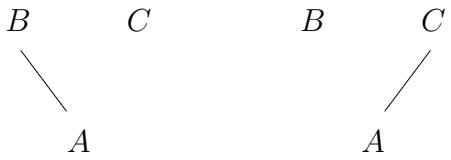
Example 2 *The sequent* $\vdash (A^\perp \parr B) \otimes B^\perp, A$ *is underivable, but has the following proof structure*



Definition 3 (Correction Graph) *From a proof structure we generate a* correction graph *by replacing all par links*



by one of the following (unary) links



Definition 4 (Proof Net) *A proof structure is a* proof net *if and only if all its correction graphs are acyclic and connected.*

We can see that by this definition the proof structure of example 2 is not a proof net as it has two correction graphs one of which is both cyclic and disconnected.

Theorem 2 ([Girard 87]) *A sequent* $\vdash A_1, \ldots, A_n$ *has a proof if and only if there is a proof net* $\mathcal{P}$ *with conclusions* $A_1, \ldots, A_n$

The above, together with cut elimination, immediately suggests the following algorithm for deciding whether or not a sequent is provable.

1. Unfold the formulas applying the par and tensor links above.

   Complexity $O(n)$ for n connectives.

2. Connect atomic formulas by axiom links.

   Complexity $O(n!)$ for $2n$ atomic formulas.

3. Check if the resulting proof structure has only acyclic and connected correction graphs.

   Complexity $O(n^2)$ for a graph with n vertices (see section 2.3).

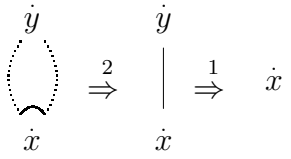
The complexity of step 2 indicates we have a very inefficient algorithm. Unfortunately, it is the best possible (complete) algorithm for the current framework.

We will, however, try to reduce the search space by detecting unsound proof structures at an early stage of the construction.

2.3 Graph Reductions

For a proof net with n par links, there will be 2^n different correction graphs, so naive application of the acyclicity and connectedness criterion is not an option. In [Danos 90], Danos gives us a better method for checking whether a proof structure is acyclic and connected. Starting with the graph of the proof structure, we apply the following reductions, until none of them is applicable

Graph Reductions



The reductions are subject to the following conditions

($\overset{1}{\Rightarrow}$) only if $x \neq y$

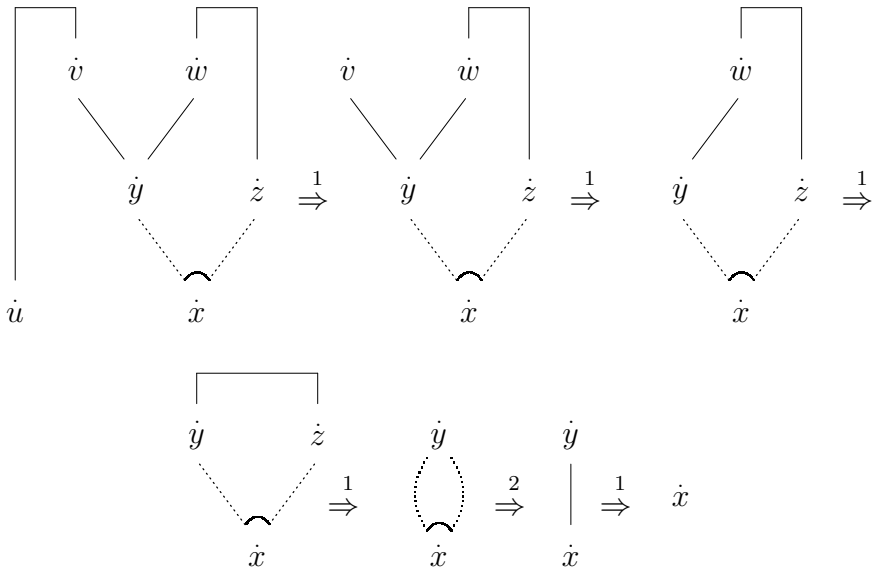
($\overset{2}{\Rightarrow}$) only if the two edges come from the same link.

It is important to note the edges of a par link are paired, as suggested by the arc connecting them. This means that when multiple par links have the same vertex as a base, which can happen after applying some reductions, we keep track of which pairs belong together.

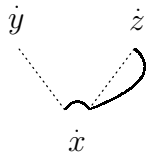
It is immediate that whenever it is possible to apply more than one reduction to a graph, the results will converge as the conflicting reductions will either 1) produce isomorphic graphs immediately, or 2) the order in which the reductions are applied can be reversed.

Reduction of a proof net will result in the trivial acyclic connected graph: a single vertex.

Example 3 *We can reduce the proof net corresponding to the sequent* $\vdash A, (A^{\perp} \otimes B) \parr B^{\perp}$ *as follows*



Reduction of a proof structure which is not a proof net will result in a graph which is not a single vertex. The unsound proof structure of example 2 will reduce in 4 steps to



which can be further reduced by a single 1 reduction after which no reductions are possible.

Theorem 3 ([Danos 90]) *A proof structure $\mathcal{S}$ is a proof net iff it reduces to a single vertex by applying reductions 1 and 2 above.*

Beginning with a proof structure with n par links and m tensor links, we can first reduce all tensor links in $O(m)$ time. Then we can find a par link to which reduction 2 and 1 can be applied in at most $O(n)$ time. If such a par link cannot be found, we fail because the proof structure is disconnected. Reducing all par links will then take $n + (n-1) + \ldots + 1 (= \frac{1}{2}n(n+1))$ time. The maximum time for determining a proof structure is a proof net will then be $O(\frac{1}{2}n(n+1) + m) = O(n^2)$.

2.4 Incremental Graph Reductions

Another important advantage of the graph reduction strategy described above, is that we can incrementally reduce the proof structure we are generating. After each axiom link, we reduce the proof structure as far as possible and check whether we have cyclic or disconnected parts.

1. (a) Starting with the graph of decomposed formulas we assign to each vertex a *multiset* of atomic formulas at that vertex. At this point the leaves will have a singleton multiset assigned to them, and all other vertices the empty multiset.
   (b) We apply all 1 reductions. The multiset assigned to the result of the reduction will be the *union* of the multisets of the reduced nodes. From this point there are only par links in the graph.
2. (a) We (nondeterministically) remove an atomic formula and its negation from the multisets of two different vertices and add an axiom link to the graph. We make sure the axiom link does not produce a cycle, and apply a 1 reduction to the new link.
   (b) We apply a combination of reduction 2 and 1 to all 2 redexes in the graph. This can result in new 2 redexes, so we repeat this step until no 2 redexes remain.
   (c) We check for connectedness, and repeat from step 2a until we have a single vertex with an empty multiset of atomic formulas.

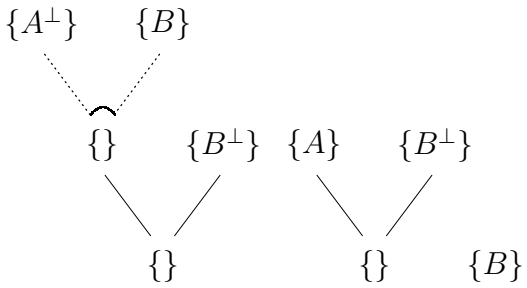
We can check for cycles and disconnectedness in the following way

- When a vertex, which is not the base of a par link and not the only vertex in the graph, has an empty set of atomic formulas assigned to it we know the proof structure will not be connected, as no future axiom link can connect it to its sister or to the rest of the proof structure.
- When we apply an axiom link between two formulas at vertices one of which is a descendant of the other, the resulting proof structure will be cyclic as when all par links between the atoms have been reduced we will have produced a cycle.

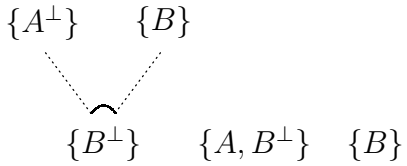
The algorithm described above still leaves us a degree of freedom in the way we select the atomic formulas at step 2a.

As an illustration, we show how this algorithm gives a derivation of the simple theorem $\vdash (A^\perp \parr B) \otimes B^\perp, A \otimes B^\perp, B$.

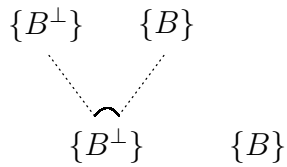
After step 1 we have the following graph



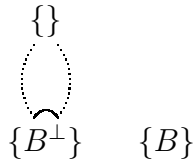
Reduction of all tensor links gives us



at step 2. Step 2a is nondeterministic, but we can see that linking the B formulas of the par link is ruled out by both the acyclicity and connectedness check. We decide to link both A formulas, which after reduction and set union gives us



There are no 2 redexes to reduce, and it is still possible to make the graph connected, so we repeat from step 2a. Should we decide to link the left $B^\perp$ of the par link to the rightmost B, the result would be disconnected. We link it to the right B of the par link instead, which results (after reducing the axiom link and taking the union of the sets) in



where we have a single 2 redex. Reducing it will produce

which is connected as both vertices still have a nonempty atom set. Back at step 2a only one pair can be selected, and reduction gives us the single vertex with empty atom set we want.

This small example already shows how a number of linkings which would never have resulted in a proof net have been prevented.

3 Categorial Grammar

Though the proof net approach described above gives us both an elegant proof theory and a simple, transparent algorithm it does so only for associative, commutative logics. For serious linguistics, however, we will want associativity and commutativity, like weakening and contraction, to be *optional*.

Categorial grammars, as described in [Moortgat 97], deal with this issue. Comparing categorial logic with multiplicative linear logic, we will notice the following differences:

Intuitionism We restrict ourselves to the *intuitionistic* fragment of linear logic, because we are interested in the Curry-Howard interpretation of proofs as (semantic) terms. In the multiplicative intuitionistic setting we only have the connectives $\otimes$ and $\multimap$. A notion of polarity (negative for antecedent formulas, positive for succedent formulas) will play the role of negation for intuitionistic proof nets.

Associativity/Commutativity We drop commutativity and associativity as global options. Without commutativity the connective $\multimap$ will split into two versions, depending on whether the implication looks for its argument to the left or right. We will write A/B (resp. $B\backslash A$) for the formula which yields an A when it finds a B to its right (resp. left). Noncommutative $\otimes$ will be written as $\bullet$.

Unary Connectives In the full logic presented in [Moortgat 97] we also have unary connectives $\Diamond$ and $\Box^{\downarrow}$ which fulfill the same kind of role as the exponentials ? and ! in linear logic in that they can be used to license or constrain the access to structural rules. You can find an exposition on the unary connectives and the embedding results in [KM 95]. We will not treat the unary connectives in this paper, but adding them is a simple generalization of the rules for the binary connectives, as shown in [Moot 96].

Multimodality Finally, we want to be able to distinguish between different modes of composition which, individually or in combination with other modes, have access to different packages of structural rules. This will enable us to have different types of composition in one logic. We will indicate the modes of connectives by using an index as subscript (e.g. $A/_i B$, $A \bullet_i B$, $A\backslash_i B$).

A way to extend the multiplicative intuitionistic fragment of linear logic to account for these differences is by using Gabbay's [Gabbay 94] labeled deduction; instead of using formulas A as our basic declarative unit, we use *labeled* formulas $\mathbf{x} : A$. The label $\mathbf{x}$ represents a piece of structural information. The rules will be adapted to operate on both the formulas and the labels.

3.1 Labeled Sequent Calculus

A labeled deductive version of the sequent calculus for multimodal categorial grammar is presented below. Labels are the following

Definition 5 (Structure Labels) *Over a countably infinite set* $\mathbf{x}, \mathbf{y}, \mathbf{z}, \ldots$ *of structure variables* $\mathcal{V}$*, we define the set of structure labels inductively as follows*

$\mathcal{L} ::= \mathcal{V} \mid (\mathcal{L}, \mathcal{L})^i$

with i *ranging over the set of modes* $\mathcal{I}$*.*

All antecedent formulas are assigned a fresh structure variable, and the succedent formula a metavariable Z which will get fully instantiated during the proof. The succedent label will represent the way the antecedent resources are configured with respect to linear order and constituency structure.

The notation $Z[X]$ will be interpreted as a label Z with a distinguished occurrence of a sublabel X. Newly introduced structure variables are assumed to be fresh.

Without the labels, the rules are essentially the intuitionistic versions of the multiplicative rules presented in section 2.1.

Identity

$$\frac{}{\mathbf{x} : A \Rightarrow \mathbf{x} : A}\ [Ax] \qquad \frac{\Gamma, \mathbf{y} : B \Rightarrow Z[\mathbf{y}] : C \quad \Delta \Rightarrow Y : B}{\Gamma, \Delta \Rightarrow Z[Y] : C}\ [Cut]$$

Multiplicatives

$$\frac{\Gamma, \mathbf{x} : A, \mathbf{y} : B \Rightarrow Z[(\mathbf{x}, \mathbf{y})^i] : C}{\Gamma, \mathbf{z} : A \bullet_i B \Rightarrow Z[\mathbf{z}] : C}\ [L\bullet] \qquad \frac{\Gamma \Rightarrow X : A \quad \Delta \Rightarrow Y : B}{\Gamma, \Delta \Rightarrow (X, Y)^i : A \bullet_i B}\ [R\bullet]$$

$$\frac{\Delta \Rightarrow Y : B \quad \Gamma, \mathbf{x} : A \Rightarrow Z[\mathbf{x}] : C}{\Gamma, \Delta, \mathbf{y} : A/_i B \Rightarrow Z[(\mathbf{y}, Y)^i] : C}\ [L/] \qquad \frac{\Gamma, \mathbf{y} : B \Rightarrow (X, \mathbf{y})^i : A}{\Gamma \Rightarrow X : A/_i B}\ [R/]$$

$$\frac{\Delta \Rightarrow Y : B \quad \Gamma, \mathbf{x} : A \Rightarrow Z[\mathbf{x}] : C}{\Gamma, \Delta, \mathbf{y} : B\backslash_i A \Rightarrow Z[(Y, \mathbf{y})^i] : C}\ [L\backslash] \qquad \frac{\Gamma, \mathbf{y} : B \Rightarrow (\mathbf{y}, X)^i : A}{\Gamma \Rightarrow X : B\backslash_i A}\ [R\backslash]$$

The logical rules are the same for all modes $i \in \mathcal{I}$. In addition to these logical rules, individual modes may have language dependent structural rules available, which operate only on the labels. To a commutative mode c, for example, the following rule would be applicable

$$\frac{\Gamma \Rightarrow Z[(Y, X)^c] : C}{\Gamma \Rightarrow Z[(X, Y)^c] : C}\ [Com]$$

Example 4 *We can now, given the following lexicon*

$l(Bill) = \mathbf{b}{:}np$
$l(is) = \mathbf{i}{:}(np\backslash_n s)/_n np$
$= \mathbf{i}{:}(np\backslash_n s)/_n (n/_n n)$
$l(the) = \mathbf{t}{:}np/_n n$
$l(President) = \mathbf{p}{:}n$
$l(fat) = \mathbf{f}{:}n/_n n$

give the following derivation for 'Bill is the President'

$$\dfrac{\dfrac{}{\mathbf{p}{:}n \Rightarrow \mathbf{p}{:}n}\ [Ax] \quad \dfrac{\dfrac{}{\mathbf{x}{:}np \Rightarrow \mathbf{x}{:}np}\ [Ax] \quad \dfrac{\dfrac{}{\mathbf{b}{:}np \Rightarrow \mathbf{b}{:}np}\ [Ax] \quad \dfrac{}{\mathbf{z}{:}s \Rightarrow \mathbf{z}{:}s}\ [Ax]}{\mathbf{b}{:}np, \mathbf{y}{:}np\backslash_n s \Rightarrow (\mathbf{b}, \mathbf{y})^n{:}s}\ [L\backslash]}{\mathbf{b}{:}np, \mathbf{i}{:}(np\backslash_n s)/_n np, \mathbf{x}{:}np \Rightarrow (\mathbf{b}, (\mathbf{i}, \mathbf{x})^n)^n{:}s}\ [L/]}{\mathbf{b}{:}np, \mathbf{i}{:}(np\backslash_n s)/_n np, \mathbf{t}{:}np/_n n, \mathbf{p}{:}n \Rightarrow (\mathbf{b}, (\mathbf{i}, (\mathbf{t}, \mathbf{p})^n)^n)^n{:}s}\ [L/]$$

In contrast to the derivation of example 1, permutations of this sentence, like 'is the Bill President', are underivable without additional assumptions about structural rules for mode n.

3.2 Labeled Proof Nets

Moortgat [Moortgat 97] proposes to add structural labeling to proof nets in the following way. This proposal should be contrasted to earlier labeling proposals like [Moortgat 90] and [Morrill 95] which were incomplete for the product formulas.

Definition 6 (Structure Labels) *Over a countably infinite set $\mathcal{V}$ of structure variables, we define the set of structure labels $\mathcal{L}$ as follows*

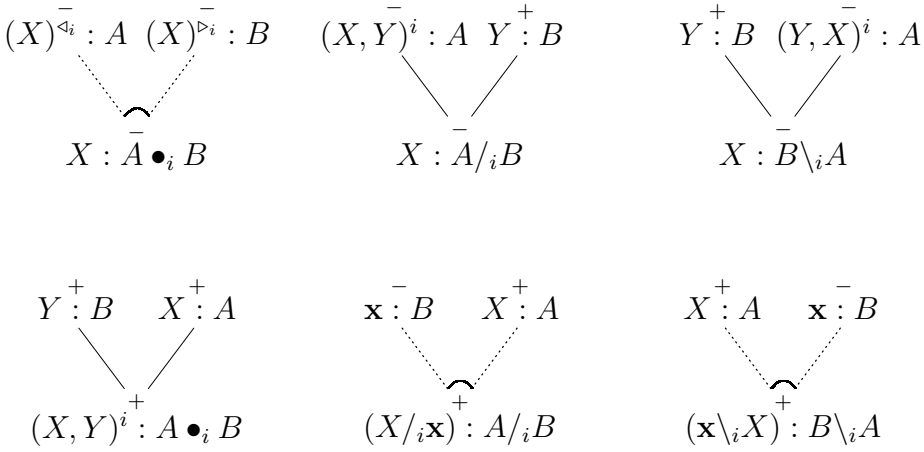
$\mathcal{L}::=\mathcal{V}$
$\mid (\mathcal{L},\mathcal{L})^i$ (*Structural counterpart of* $\bullet_i$)
$\mid (\mathcal{V}\backslash_i\mathcal{L})$ (*Auxiliary constructor for* $\backslash_i$)
$\mid (\mathcal{L}/_i\mathcal{V})$ (*Auxiliary constructor for* $/_i$)
$\mid (\mathcal{L})^{\triangleleft_i}$ (*Auxiliary constructor for* $\bullet_i$)
$\mid (\mathcal{L})^{\triangleright_i}$ (*Auxiliary constructor for* $\bullet_i$)

Definition 7 (Normal Labels) *The set of* normal *labels $\mathcal{N}$ is a subset of $\mathcal{L}$ defined as follows*

$\mathcal{N}::=\mathcal{V} \mid (\mathcal{N},\mathcal{N})^i$

As can be seen from the definition above, there are two kinds of label constructors: one *structural* and several *auxiliary*. The structural connective is the label constructor we used for the labeled sequents. In addition, we have an auxiliary constructor for each of the connectives in our formula language. The purpose of these constructors will be to check the sublinear constraints on derivability for that specific connective.

We decompose labeled formulas by applying the following links, where all newly occurring structural or metavariables are fresh. Negative formulas are initially assigned distinct structural variables $\mathbf{x},\mathbf{y},\ldots$, the positive formula a metavariable Z.



From the decomposed formulas we generate a labeled proof structure as usual by connecting the atomic formulas by axiom links, unifying the labels. For proof nets this unification will always be possible.

We now define a set of *conversions* on the succedent label, which will check the sublinear constraints on derivability. We have one conversion for each connective.

We call the label on the right hand side of the conversion a *redex* and the label on the left hand side its *contractum*. Converting a label X to a label Y ($X \rightarrow Y$) consists of

replacing all occurrences of a redex by its contractum. We will write $\twoheadrightarrow$ (reduces to) for the transitive, reflexive closure of $\rightarrow$.

Residuation Conversions

$$((X)^{\triangleleft_i}, (X)^{\triangleright_i})^i \rightarrow X \quad [\mathrm{Res}\bullet_i]$$
$$((X, Y)^i /_i Y) \rightarrow X \quad [\mathrm{Res}/_i]$$
$$(Y \backslash_i (Y, X)^i) \rightarrow X \quad [\mathrm{Res}\backslash_i]$$

We can read the residuation conversions as checking *constraints*. The residuation conversion for A/B specifies the constraint that the structural variable assigned to the subformula B should be the right daughter of the outermost structural connective. It should also have the same index. This is similar to the right rule for / in the labeled sequent calculus.

The residuation conversion for $A \bullet B$ checks if the two labels assigned to the subformulas A and B occur with A to the direct left of B and composed with the right mode of composition (compare this to the left rule for $\bullet$ in the labeled sequent calculus).

Like with the sequent calculus, we can add structural rules to our label conversions. The structural rule of commutativity for mode c, for example, would translate into the following label conversion

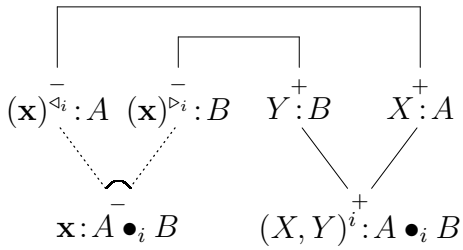
$$(X, Y)^c \rightarrow (Y, X)^c$$

Definition 8 (Reducibility) *We will call a label L* reducible *if and only if there is a normal label N such that $L \twoheadrightarrow N$.*

If the label assigned to the succedent formula is irreducible, some of the conditions on the labels in the sequent calculus can not be met, and the proof structure is not a proof net.

Example 5 *A standard well-behavedness property for a sequent or proof net calculus is that we can restrict ourselves to atomic instances of the axiom rule. The reader can easily check that we can derive all non-atomic instances of this rule at the logical level.*

The label reductions should therefore also allow us to produce a normal label for those. For the product formula, we can do so as follows



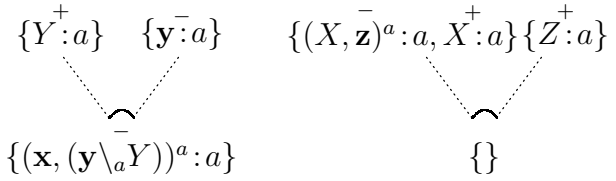
Using a non-atomic axiom link would produce a label $\mathbf{x}$ for the succedent formula immediately. The unifications $X = (\mathbf{x})^{\triangleleft_i}$ and $Y = (\mathbf{x})^{\triangleright_i}$ will give us a succedent label $((\mathbf{x})^{\triangleleft_i}, (\mathbf{x})^{\triangleright_i})^i$. We can now use the residuation conversion for the product formula to this label to get $((\mathbf{x})^{\triangleleft_i}, (\mathbf{x})^{\triangleright_i})^i \rightarrow \mathbf{x}$.

The algorithm for the labeled proof net calculus is only a slight modification of the previous algorithm, the only addition being item 2b.

1. (a) Starting with the graph of decomposed formulas we assign to each vertex a *multiset* of atomic formulas at that vertex. At this point the leaves will have a singleton multiset assigned to them, and all other vertices the empty multiset.
   (b) We apply all 1 reductions. The multiset assigned to the result of the reduction will be the *union* of the multisets of the reduced nodes. From this point there are only par links in the graph.
2. (a) We (nondeterministically) remove an atomic formula and its negation from the multisets of two different vertices and add an axiom link to the graph. We make sure the axiom link does not produce a cycle, and apply a 1 reduction to the new link.
   (b) We unify the labels assigned to the two formulas, and make sure the constraints specified by the labels can still be satisfied after this unification.
   (c) We apply a combination of reduction 2 and 1 to all 2 redexes in the graph. This can result in new 2 redexes, so we repeat this step until no 2 redexes remain.
   (d) We check for connectedness, and repeat from step 2a until we have a single vertex with an empty multiset of atomic formulas.

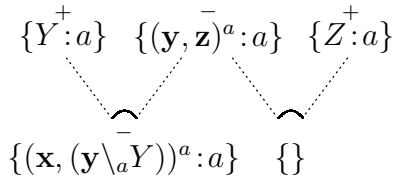
As a final example we will show how this algorithm derives the theorem $a/_a(a\backslash_a a) \Rightarrow a/_a(a\backslash_a a)$.

After step 1 of the algorithm we will be left with the following graph.



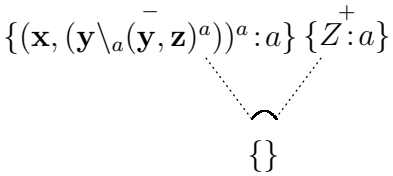
Constraints:
$Y \twoheadrightarrow (\mathbf{y}, Y')^a$
$Z \twoheadrightarrow (Z', \mathbf{z})^a$

We start linking the $\mathbf{y} : a$ formula. Of the three positive formulas we could link it to, only linking it to $X : a$ allows us to satisfy both label constraints. Performing this link gives us the following graph



Constraints:
$Y \twoheadrightarrow (\mathbf{y}, Y')^a$
$Z \twoheadrightarrow (Z', \mathbf{z})^a$

We link the $(\mathbf{y}, \mathbf{z})^a : a$ formula. We can satisfy the label constraints both by linking it to $Y : a$ and by linking it to $Z : a$, but graph reductions after linking it to $Z : a$ would produce a disconnected graph. We add the only possible axiom link to the graph, satisfying the constraint on Y, and reduce the graph to



Constraints:
$Z \twoheadrightarrow (Z', \mathbf{z})^a$

where we can perform the last axiom link, unifying Z to $(\mathbf{x}, (\mathbf{y}\backslash_a (\mathbf{y}, \mathbf{z})^a))^a$, which we can reduce to $(\mathbf{x}, \mathbf{z})^a$ satisfying the last constraint.

4 Conclusion

We have shown how labeled proof nets give us a transparent algorithm for automated theorem proving in a categorial setting. Though its complexity indicates that there will be cases in which it will not produce its output in a reasonable amount of time, the combination of early failure on logical and structural grounds gives us an automated deduction system which is nonetheless very useful in practice.

References

[BtM 97] Benthem, J. van, and A. ter Meulen (eds.), *Handbook of Logic and Language*, Elsevier, 1997.

[Danos 90] Danos, V. *La Logique Linéaire Appliquée à l'Étude de Divers Processus de Normalisation at Principalement du Lambda-Calcul*, Thèse de Doctorat, Université de Paris VII, 1990.

[Gabbay 94] Gabbay, D. *Labeled Deductive Systems*, Report MPI-I-94-223, Max-Planck-Institut für Informatik, Saarbrücken, 1994.

[Girard 87] Girard, J.Y. *Linear Logic*, Theoretical Computer Science 50, 1987, pp. 1-102.

[Girard 96] Girard, J.Y. *Proof-nets: The parallel syntax for proof-theory*, In P. Agliano and A. Ursini (eds.), *Logic and Algebra*, Marcel Dekker, New York, 1996.

[Girard e.a. 89] Girard, J.Y., Y. Lafont and P. Taylor, *Proof and Types*, Cambridge University Press, 1989.

[Girard e.a. 95] Girard, J.Y., Y. Lafont and L. Regnier (eds.), *Advances in Linear Logic*, London Mathematical Society Lecture Notes, Cambridge University Press, 1995

[Kurtonina 95] Kurtonina, N. *Frames and Labels. A Modal Analysis of Categorial Inference*, PhD Thesis, OTS Utrecht, ILLC Amsterdam, 1995.

[KM 95] Kurtonina, N. and M. Moortgat *Structural Control*, DYANA deliverable R1.1.C, BRA 6852, Utrecht, 1995.

[Moortgat 90] Moortgat, M. *Unambiguous Proof Representations for the Lambek Calculus*, Proceedings 7th Amsterdam Colloquium, 1990.

[Moortgat 96] Moortgat, M. *Labeled Deduction in the Composition of Form and Meaning*, in [OR 96].

[Moortgat 97] Moortgat, M. *Categorial Type Logics*, chapter 2 of [BtM 97].

[MO 93] Moortgat, M. and R. Oehrle *Logical Parameters and Linguistic Variation. Lecture Notes on Categorial Grammar*, Fifth European Summer School in Logic, Language and Information, Lisbon, 1993.

[Moot 96] Moot, R. *Proof Nets and Labeling for Categorial Grammar Logics*, MA Thesis, Utrecht University, 1996.

[Morrill 94] Morrill, G. *Type Logical Grammar. Categorial Logic of Signs*, Kluwer, Dordrecht, 1994.

[Morrill 95] Morrill, G. *Clausal Proofs and Discontinuity*, Bulletin of the IGPL 3(2,3). Special Issue on Deduction and Language (ed. R. Kempson), 1995, pp. 403-417.

[OR 96] Ohlbach, H.J. and U. Reyle (eds.), *Logic, Language and Reasoning. Essays in Honor of Dov Gabbay, Part I*, Kluwer, Dordrecht, to appear.

[Roorda 91] Roorda, D. *Resource Logics: A Proof-theoretical Study*, PhD Thesis, University of Amsterdam, 1991.