

Type-Logical and Hyperedge Replacement Grammars

Richard Moot

Méthodes Formelles, 20 May 2008

Introduction

The Non-Associative Lambek Calculus

NL was introduced by Lambek (1961) as a restriction of the Lambek calculus. Since linguists tend to think of trees as the basic linguistic structures, a logic of trees seems (at least a priori) a good choice.

Kandulski (1988) showed that **NL** generates only context-free languages. On the positive side, though de Groot (1999) showed we can parse **NL** grammars in polynomial time.

Introduction

The Multimodal Lambek Calculus

Given that there is convincing evidence that natural languages are not context-free (Shieber 1985), various extensions to **NL** have been proposed.

Moortgat & Oehrle (1994) have given an analysis of Dutch verb clusters using the *multimodal* Lambek calculus $\mathbf{NL} \diamond_{\mathcal{R}}$, thereby showing this calculus generates more than just context-free languages.

However, the large freedom in proposing structural rules in \mathcal{R} means the logic is Turing complete in general even though a simply restriction on the structural rules gives a PSPACE formalism generating exactly the context-sensitive languages (Moot 2002).

Introduction

The Multimodal Lambek Calculus

In terms of practical parsing, a PSPACE bound is still quite high. We would like to find a restriction of the multimodal calculus which extends **NL**, but does so while keeping a polynomial parsing algorithm.

The so-called *mildly* context-sensitive languages and the different corresponding seem a good compromise between parsing complexity and descriptive accuracy.

So the question I want to answer today is what fragment of $\mathbf{NL} \diamond_{\mathcal{R}}$ allows for polynomial parsing?

Introduction

Overview of Lambek Calculi: Logic, Language, Complexity

Logic	NL	L	???	NL $\diamond_{\mathcal{R}}$
Complexity	P	NP	P	PSPACE
Languages	CFL	CFL	MCSL	CSL

Introduction

Mildly Context-Sensitive Languages and Grammars

- 1 Contains the context-free languages.
- 2 Polynomial parsing.
- 3 Constant growth.
 - Excluded: $\{a^p \mid p \text{ is prime}\}$
 - Excluded: $\{a^{2^n} \mid n \geq 0\}$
- 4 Limited cross-serial dependencies.

Introduction

Mildly Context-Sensitive Languages and Grammars

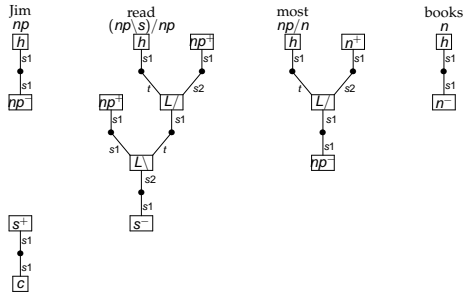
Some typical mildly context-sensitive languages are:

$\{ww \mid w \in \{a, b\}^*\}$	copy language
$\{a^n b^n c^n d^n \mid n \geq 0\}$	counting dependencies
$\{a^n b^m c^n d^m \mid n \geq 0, m \geq 0\}$	crossed dependencies

Many independently proposed frameworks in computational linguistics (Tree Adjoining Grammars, Linear Indexed Grammars, Head Grammars and Combinatory Categorical Grammars) generate exactly the same class of mildly context-sensitive languages (Vijay-Shanker & Weir 1994).

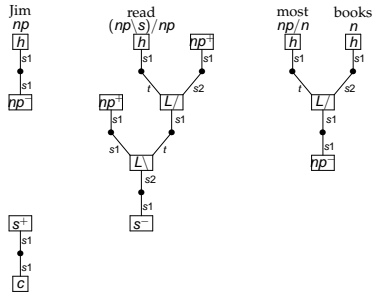
Example

AB Proof Nets



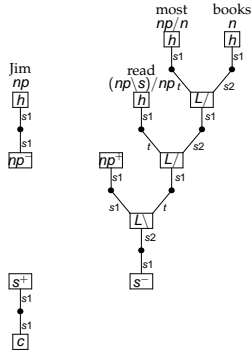
Example

AB Proof Nets



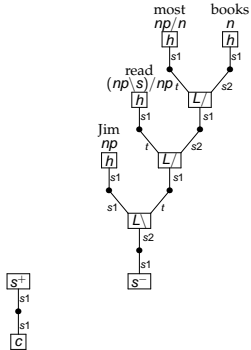
Example

AB Proof Nets



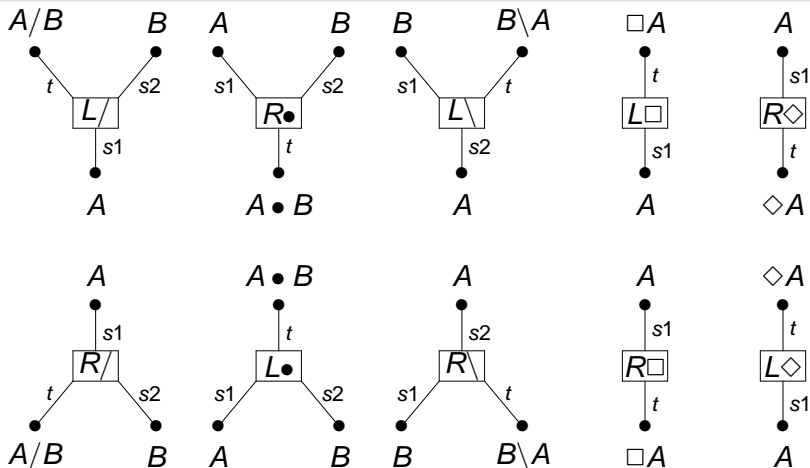
Example

AB Proof Nets



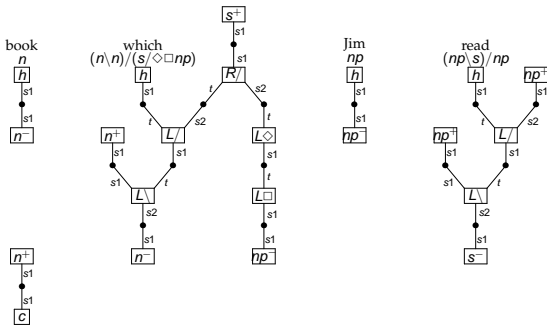
Proof Structures

Logical Links



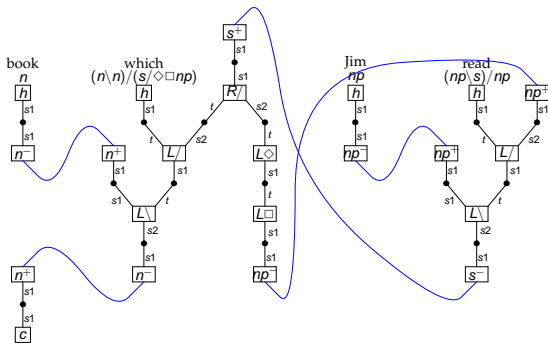
Proof Structures

Example



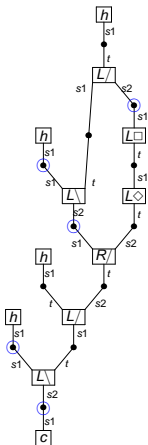
Proof Structures

Example— Connecting The Axioms



Proof Structures

Example— Finding The Axioms



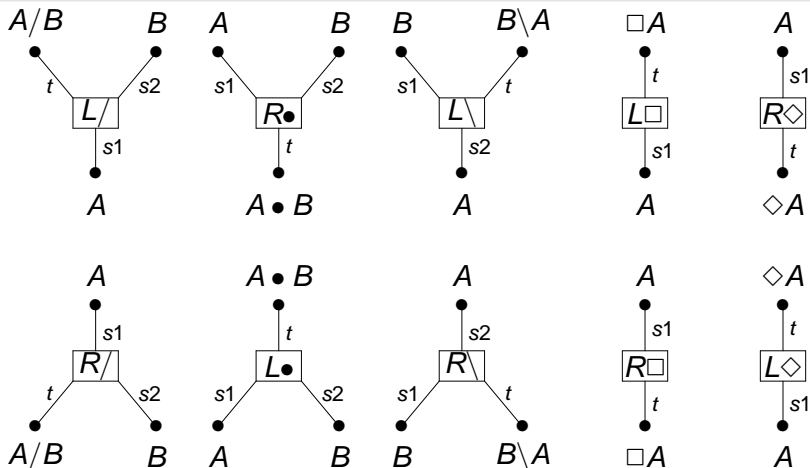
From Proof Structures to Abstract Proof Structures

To make the statement of our correctness condition easier, we abstract away over some of the structure present in proof nets to obtain *abstract* proof structures.

- We no longer distinguish between the different tensor links.
- As a consequence, we can no longer distinguish between axiom, flow and cut formulas: only the external formulas, which are linked to the h and c hyperedges are still available.
- Abstract proof structures which are trees and contain tensor links only will be called *tensor trees*.

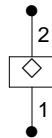
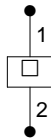
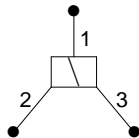
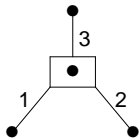
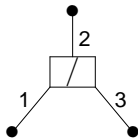
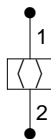
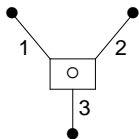
Proof Structures

Logical Links



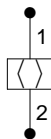
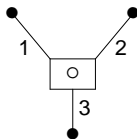
Abstract Proof Structures

Links



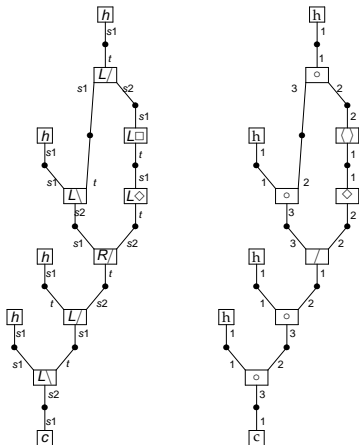
Tensor Trees

Links

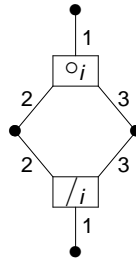
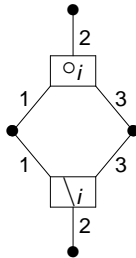
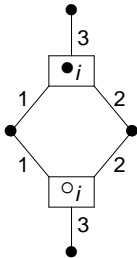


Abstract Proof Structures

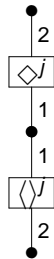
Example



Contractions

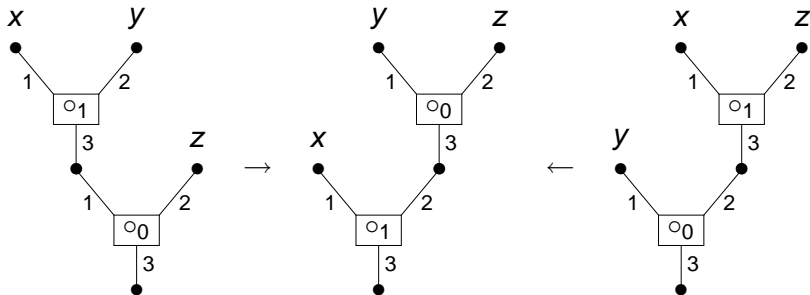


Contractions



Structural Rules

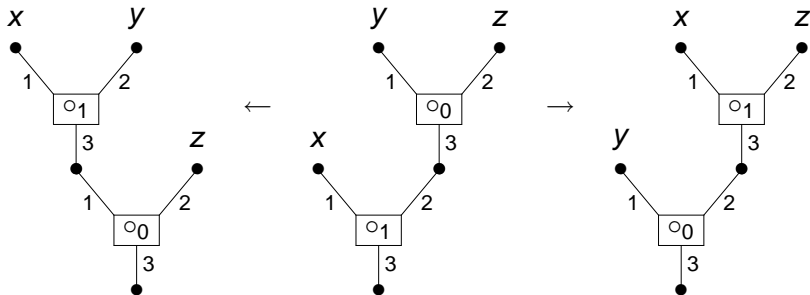
Mixed Associativity and Commutativity— Extraction



Structural Rules

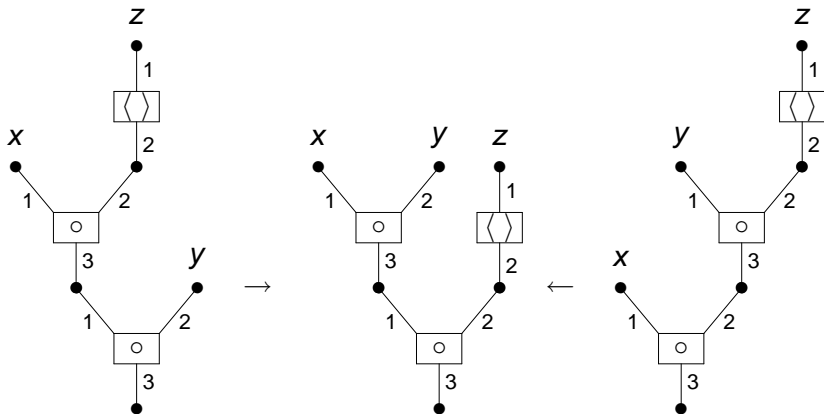
Mixed Associativity and Commutativity

— Infixation



Structural Rules

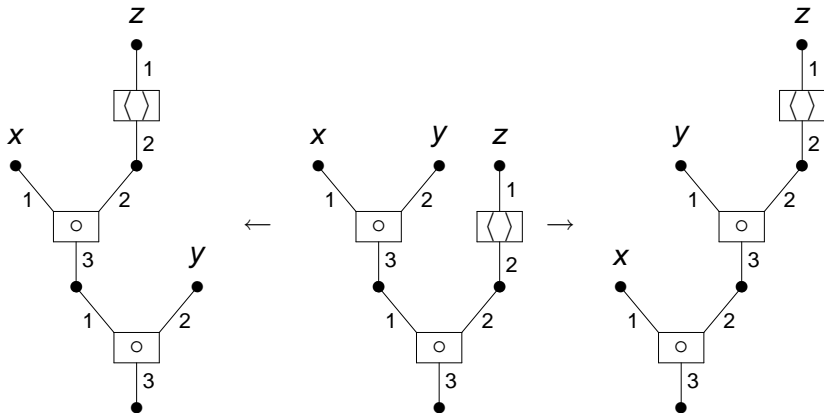
Unary Control—Extraction



Structural Rules

Unary Control

— Infixion



Proof Nets

Definition

Definition

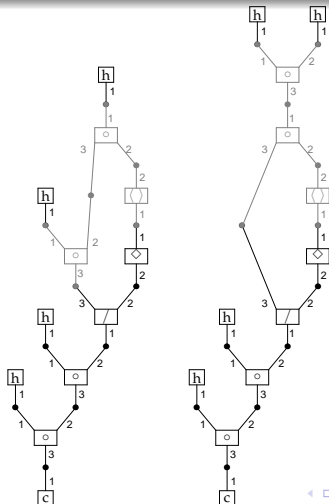
An $\mathbf{NL}\diamond$ proof structure \mathcal{P} is a *proof net* iff its underlying abstract proof structure \mathcal{A} contracts to a tensor tree.

Definition

An $\mathbf{NL}\diamond_{\mathcal{R}}$ proof structure \mathcal{P} is a *proof net* iff its underlying abstract proof structure \mathcal{A} converts to a tensor tree using the contractions *and the structural conversions in \mathcal{R}* .

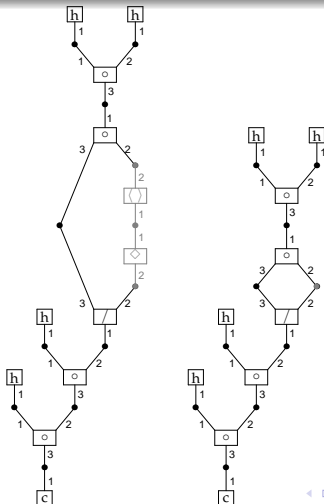
Contractions and Structural Rules

Example



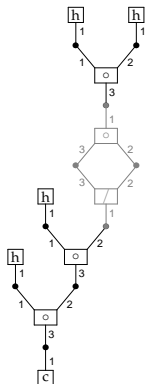
Contractions and Structural Rules

Example



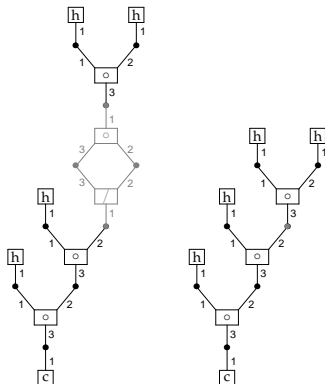
Contractions and Structural Rules

Example



Contractions and Structural Rules

Example

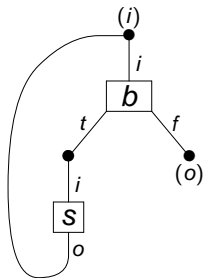


Hyperedge Replacement Grammars

Hyperedge replacement grammars were introduced by Bauderon & Courcelle (1987) and Habel & Kreowski (1987) as a type of context-free graph grammars. Few people have studied the link between hyperedge replacement grammars and proof nets, however.

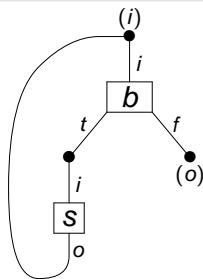
Hyperedge Replacement Grammars

Hypergraphs With External Nodes

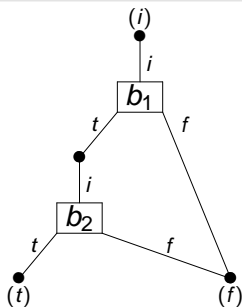


Hyperedge Replacement Grammars

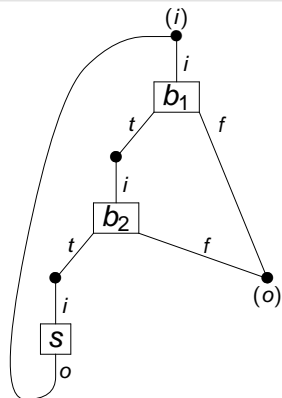
Hyperedge Replacement



H



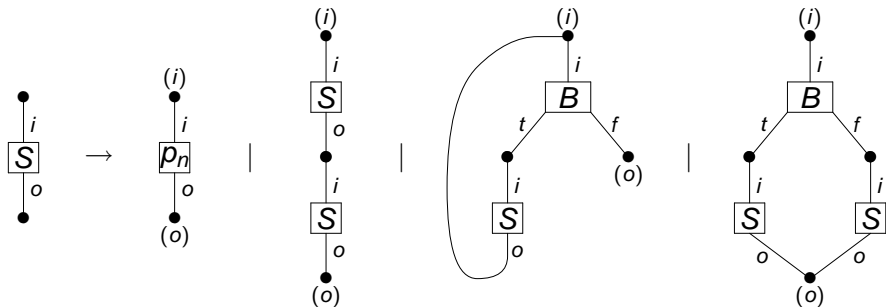
K



$H[e := K]$

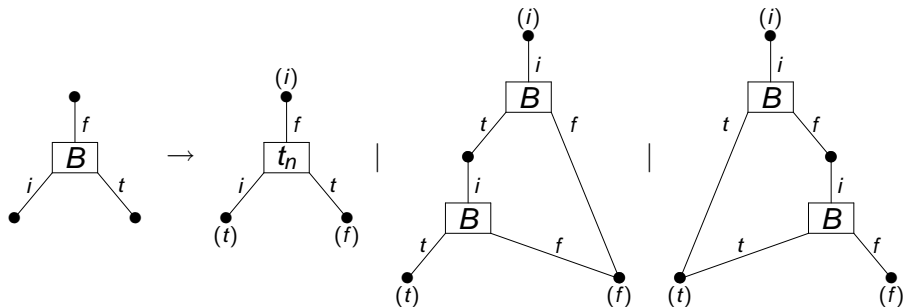
Hyperedge Replacement Grammars

Example Grammar



Hyperedge Replacement Grammars

Example Grammar



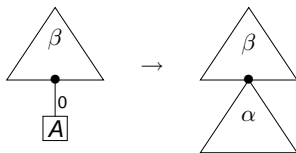
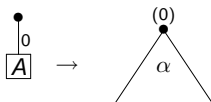
Tree Adjoining Grammars as HR Grammars

Tree Adjoining Grammars (Joshi, Levi & Takahashi 1975) are a mildly context-sensitive grammar formalism. In context of HR grammars, they can be seen as a special case of hyperedge replacement grammars where:

- every non-terminal hyperedge label has at most two tentacles.
- every right-hand side of a HR rule is either:
 - a tree with the root as its sole external node.
 - a tree with a root and a leaf as its external nodes.

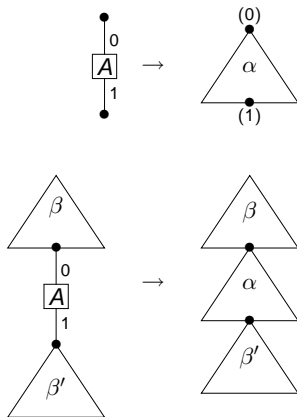
Tree Adjoining Grammars as HR Grammars

Substitution



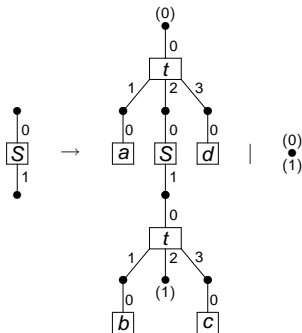
Tree Adjoining Grammars as HR Grammars

Adjunction



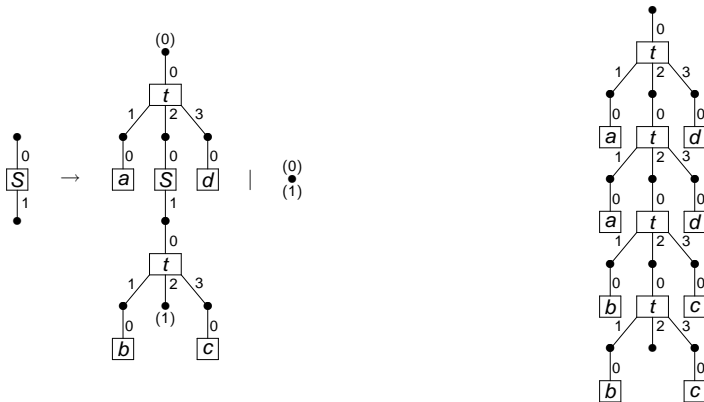
Tree Adjoining Grammars as HR Grammars

Example: $a^n b^n c^n d^n$



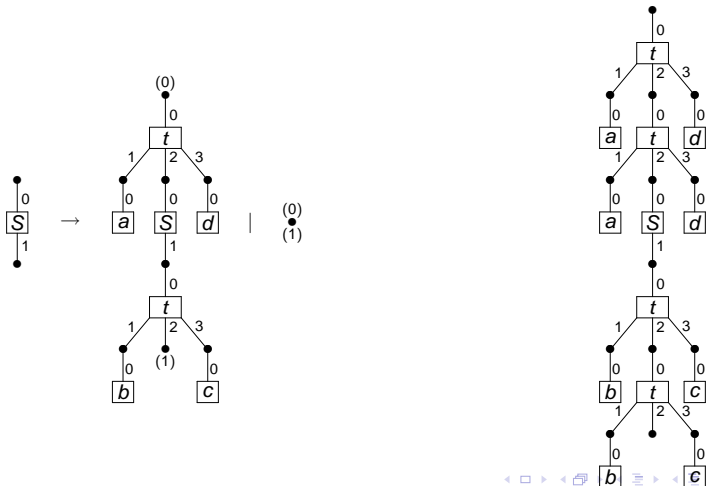
Tree Adjoining Grammars as HR Grammars

Example: $a^n b^n c^n d^n$



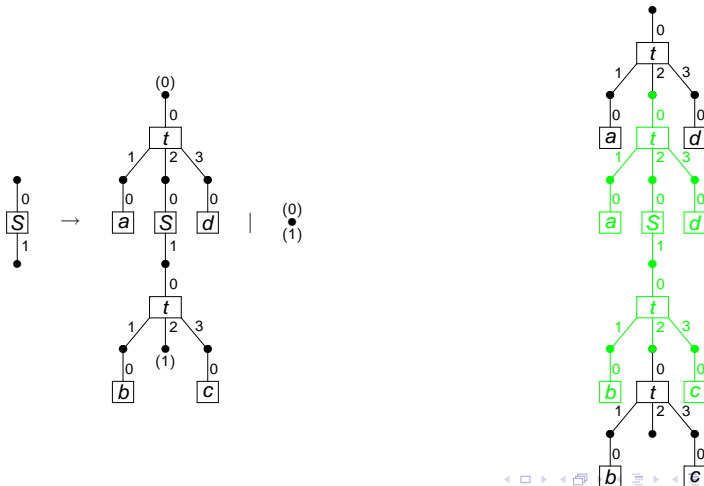
Tree Adjoining Grammars as HR Grammars

Example: $a^n b^n c^n d^n$



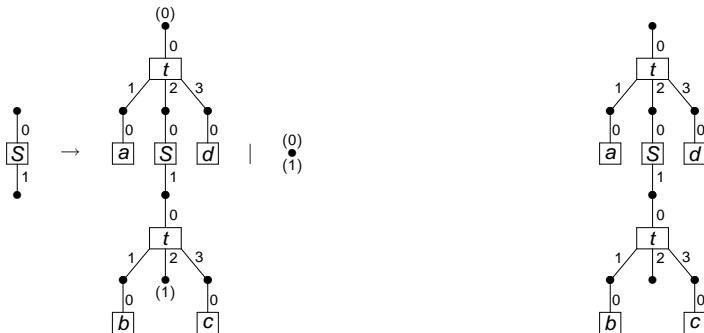
Tree Adjoining Grammars as HR Grammars

Example: $a^n b^n c^n d^n$



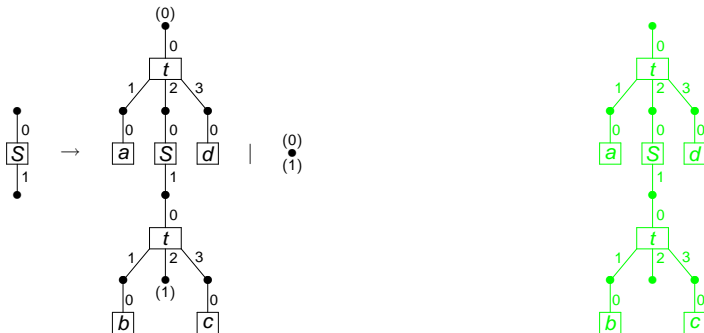
Tree Adjoining Grammars as HR Grammars

Example: $a^n b^n c^n d^n$



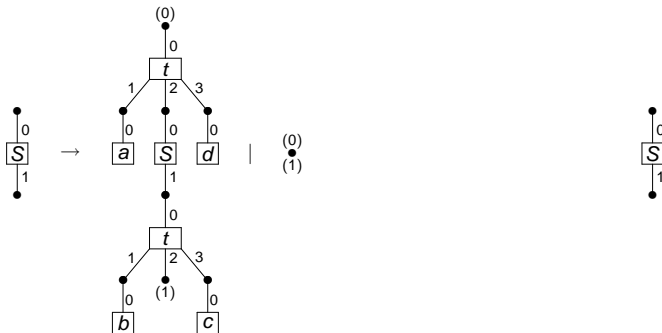
Tree Adjoining Grammars as HR Grammars

Example: $a^n b^n c^n d^n$



Tree Adjoining Grammars as HR Grammars

Example: $a^n b^n c^n d^n$



HRG and Proof Nets

Nonterminal Symbols

- S (start),

HRG and Proof Nets

Nonterminal Symbols

- S (start),
- T_{00} (tree, cut),
- T_{01} (tree, flow down),
- T_{10} (tree, flow up),
- T_{11} (tree, axiom),

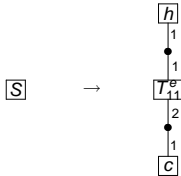
HRG and Proof Nets

Nonterminal Symbols

- S (start),
- T_{00} (tree, cut),
- T_{01} (tree, flow down),
- T_{10} (tree, flow up),
- T_{11} (tree, axiom),
- V_{00} (vertex, cut),
- V_{01} (vertex, flow down),
- V_{10} (vertex, flow up),
- V_{11} (vertex, axiom).

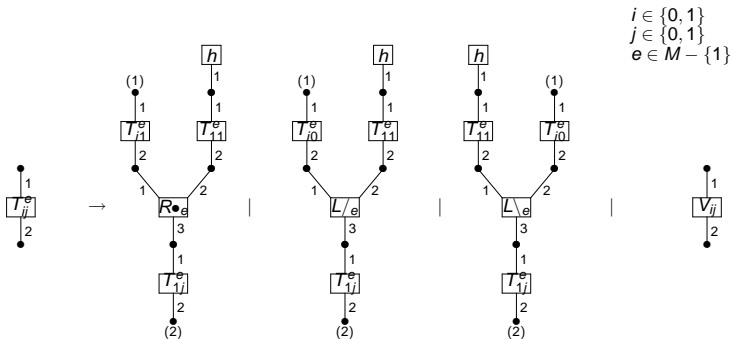
HRG and Proof Nets

Initial Axiom



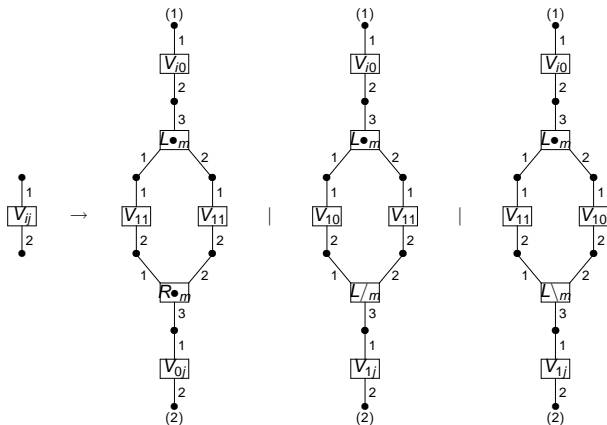
HRG and Proof Nets

Tensor Trees



HRG and Proof Nets

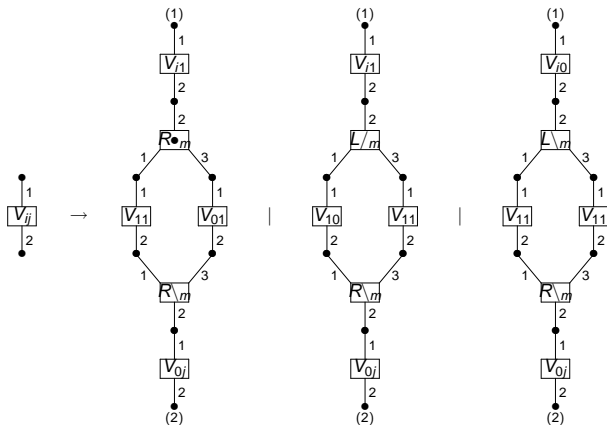
Contractions: $L \bullet$



$i \in \{0, 1\}$
 $j \in \{0, 1\}$
 $m \in M$

HRG and Proof Nets

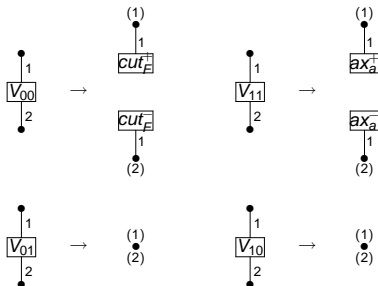
Contractions: $R \setminus$



$i \in \{0, 1\}$
 $j \in \{0, 1\}$
 $m \in M$

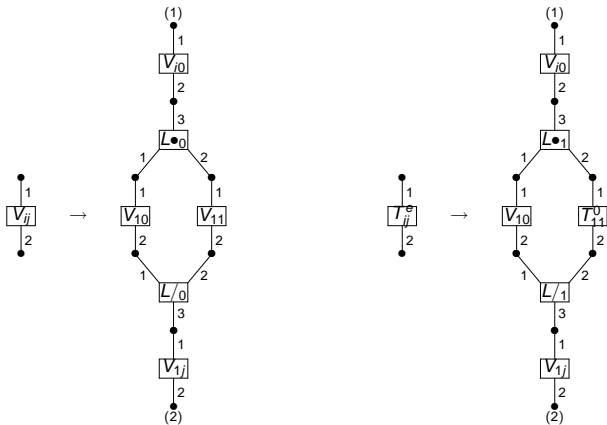
HRG and Proof Nets

Cut, Flow, Axiom



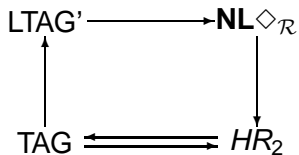
HRG and Proof Nets

Structural Rules



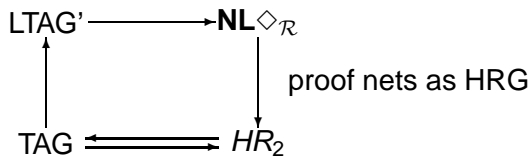
Intermezzo

Relations so far



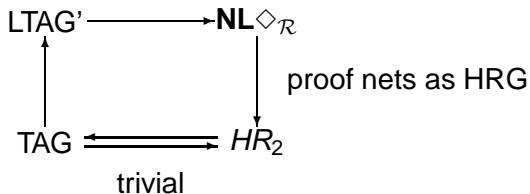
Intermezzo

Relations so far



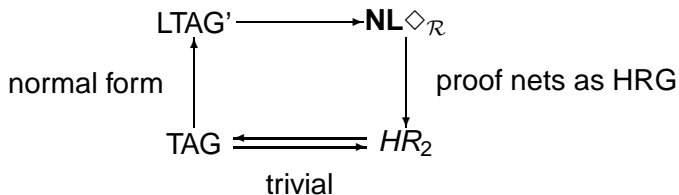
Intermezzo

Relations so far



Intermezzo

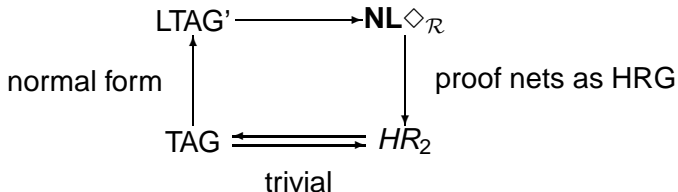
Relations so far



Intermezzo

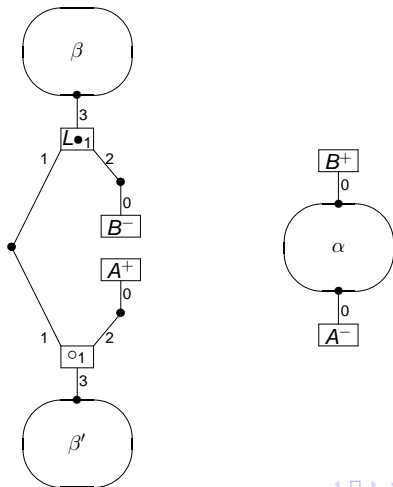
Relations so far

adjunction as contraction
and structural rules



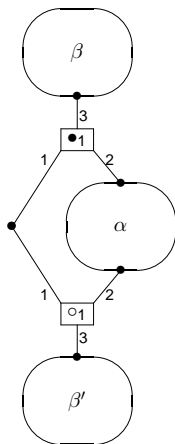
Simulating Adjunction

Initial Configuration of Adjunction Point



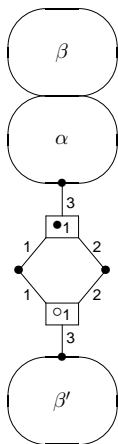
Simulating Adjunction

Axioms



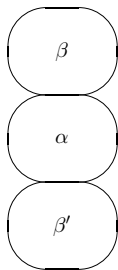
Simulating Adjunction

Structural Rules



Simulating Adjunction

Contraction and Final Result



Conclusions...









- Hyperedge replacement grammars, tree adjoining grammars and proof nets for $\mathbf{NL}\diamond$ (with mixed associativity and mixed commutativity) all generate the same string languages.
- These string languages are in the class of mildly context-sensitive languages.
- As a consequence this restricted class of categorial grammars is polynomially parseable.
- Thanks to the simplicity of their basic operations, hyperedge replacement grammars and tree adjoining grammars have played a mayor role in establishing this correspondance.

Conclusions...

And Future Work!

- Implement the polynomial algorithms!
- More classes of structural rules seem to permit a treatment by hyperedge replacement grammars. Identify as many as possible.
- It is well-knowns that augmenting the rank of the hyperedge replacement grammar augments the string (and tree) generating power. For example, an tree-generating HRG of rank n can generate $2n$ counting dependencies. What are the corresponding categorial grammars?

References

-  Baderon, M. & Courcelle, B. (1987), 'Graph expressions and graph rewritings', *Mathematical Systems Theory* **20**(1), 83–127.
-  Habel, A. & Kreowski, H.-J. (1987), May we introduce to you: Hyperedge replacement, in 'Graph Grammars and Their Application to Computer Science', Vol. 291 of *Lecture Notes in Computer Science*, Springer, pp. 15–26.
-  Joshi, A., Levi, L. S. & Takahashi, M. (1975), 'Tree adjunct grammars', *Journal of Computer and System Science* **10**, 136–163.
-  Lambek, J. (1961), On the calculus of syntactic types, in R. Jacobson, ed., 'Structure of Language and its Mathematical Aspects, Proceedings of the Symposia in Applied Mathematics', Vol. XII, American Mathematical Society, pp. 166–178.
-  Mortgat, M. & Oehle, R. T. (1994), Adjacency, dependency and order, in 'Proceedings 9th Amsterdam Colloquium', pp. 447–466.
-  Moot, R. (2002), Proof Nets for Linguistic Analysis, PhD thesis, Utrecht Institute of Linguistics OTS, Utrecht University.
-  Shieber, S. (1985), 'Evidence against the context-freeness of natural language', *Linguistics & Philosophy* **8**, 333–343.
-  Vijay-Shanker, K. & Weir, D. (1994), 'The equivalence of four extensions of context free grammars', *Mathematical Systems Theory* **27**(6), 511–546.