Work in progress ...

# Can We Preserve the Polynomial Parsability of Restricted Graph Grammars when Adding Contextuality?

**Johanna Björklund, Frank Drewes, and Anna Jonsson**
{johanna,drewes,aj}@cs.umu.se

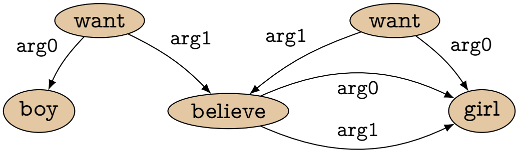
UMEÅ UNIVERSITY

# Introduction

Algorithmic processing of natural language requires formal semantic representations.

Graph-based representations combine versatility with transparency.

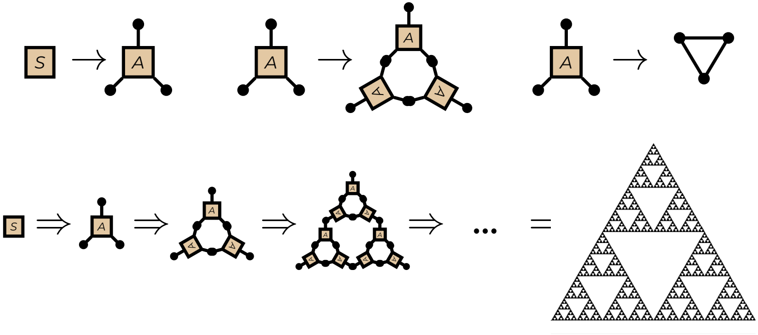
A popular example of semantic graphs is abstract meaning representation (Banarescu et al. 2013).



# Hyperedge replacement grammars

Most previous work set out from hyperedge replacement grammars (Habel 1992)

Weighted graph-generating devices over the Boolean semiring ;-)



# Related work

**Parsing is NP-complete $\Rightarrow$ we need suitable restrictions.**

- AMR parsing with HRGs (Chiang, Andreas, et al. 2013)
- DAG grammar (Chiang, Drewes, et al. 2016)
- Predictive Top-Down parsing for HRGs (Drewes, Hoffmann, and Minas 2015)
- Predictive Shift-Reduce parsing for HRGs (Drewes, Hoffmann, and Minas 2017)
- Constrained Graph Algebras (Groschwitz et al. 2017)
- Order Preserving HRGs (Björklund et al. 2021)

# Our approach

**Despite NP-completeness, HRGs are too weak ...**

- Only a constant number of nodes can be “remembered”; new edges can be attached only to those.
- In contrast, cross references created by, e.g., pronouns, give rise to edges that “can point anywhere”.
- We make use of the idea of contextual hyperedge replacement (Drewes and Hoffmann 2015).
- Allows us to insert edges to nodes not explicitly remembered.

# Tree-based graph generation

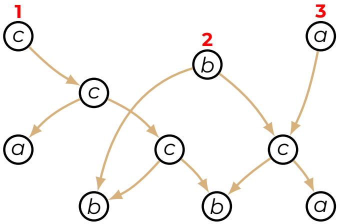
Like others before (Courcelle, Engelfriet, D.), we use the principle of tree-based generation (Mezei and Wright 1967)

**General idea:** A regular tree grammar generates trees interpreted as expressions by an algebra which evaluates them into objects.

**In our case:**

- The domain is the set of directed graphs.
- Graphs have a sequence of distinguished nodes called *ports*, that indicate where new edges can be attached in a controlled manner.
- The operations graph extension and union are used to combine and extend graphs into larger ones.

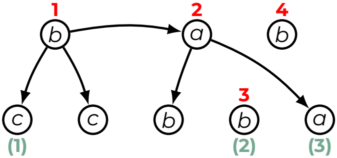
# Graph extension operations

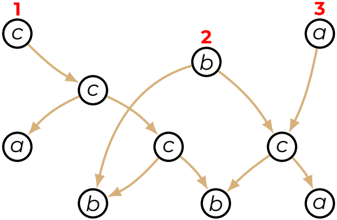



Graph

# Graph extension operations

Operation

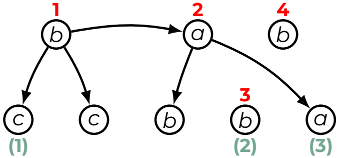


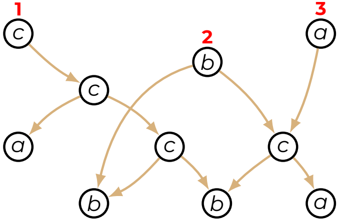




Graph

# Graph extension operations

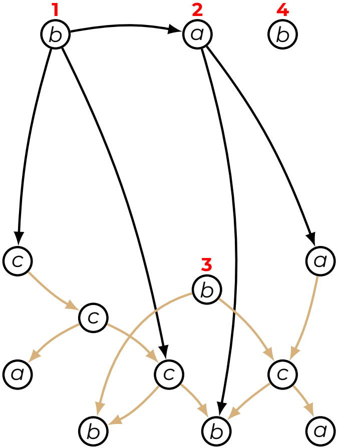
Operation




Graph

Result



# One more operation

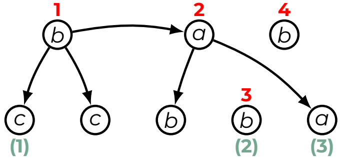
**We need one more operation: union.**

- The union of two graphs puts them next to each other, concatenating the port sequences.

# A closer look at graph extension

**Two crucial requirements on graph extension operations:**

- Edges originante only from newly generated ports.
- Every port of the input graph that is “forgotten” has one or more incoming edges.



# The basis for polynomial parsing

**Lemma**

Let $G = val(s)$ for a tree $s$ over graph extensions and graph union, and let $t$ be a subtree of $s$.

Then the subgraph $G'$ of $G$ resulting from the evaluation of $t$ is the graph induced by the nodes reachable in $G$ from the ports of $G'$.

# Parsing algorithm in brief

1. Let $k$ be the maximum length of port sequences.
2. Use dynamic programming to determine, for all nonterminals $A$ and all node sequences $p$, $|p| \leq k$, whether $A$ can generate $G \triangledown p$.
3. If there is a rule $A \to \Xi(B)$ for an extension operation $\Xi$, matching $ports_{\Xi}$ to $p$ yields zero or more possibilities for a sequence $p'$ of ports of the argument.
4. Check recursively whether $B$ can generate any of those $G \triangledown p'$, memoizing results.
5. If so, then $A$ can generate $G \triangledown p$, otherwise not.
6. Union can be handled similarly, but is simpler.

**Time complexity**

The running time of the algorithm is $O(n^k)$.

# Conclusion and future work

We have:

- Simple formalism for expressing languages of semantic graphs
- Non-context-free by the use of contextual rules
- Polynomial time parsing (though non-uniformly)

We now want to:

- Make the underlying grammar non-regular
- Refine the analysis of the parsing complexity
- Investigate whether a weighted version makes sense and can efficiently be parsed.

# THANK YOU!

drewes@cs.umu.se

# Bibliography I


Laura Banarescu et al. (2013). “Abstract Meaning Representation for Sembanking”. In: *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pp. 178–186.

Henrik Björklund et al. (2021). “Uniform parsing for hyperedge replacement grammars”. In: *Journal of Computer and System Sciences* 118, pp. 1–27.

David Chiang, Jacob Andreas, et al. (2013). “Parsing Graphs with Hyperedge Replacement Grammars”. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 924–932.

David Chiang, Frank Drewes, et al. (2016). “Weighted DAG Automata for Semantic Graphs”. Submitted.

Frank Drewes and Berthold Hoffmann (2015). “Contextual hyperedge replacement”. In: *Acta Informatica* 52.6, pp. 497–524.

# Bibliography II


Frank Drewes, Berthold Hoffmann, and Mark Minas (2015). "Predictive Top-down Parsing for Hyperedge Replacement Grammars". In: *Proceedings of the 8th International Conference on Graph Transformation*, pp. 19–34.

– (2017). "Predictive shift-reduce parsing for hyperedge replacement grammars". In: *Proc. 10th Intl. Conf. on Graph Transformation (ICGT'17)*. Lecture Notes in Computer Science.

Jonas Groschwitz et al. (2017). "A constrained graph algebra for semantic parsing with AMRs". In: *IWCS 2017 - 12th International Conference on Computational Semantics - Long papers*. URL: https://www.aclweb.org/anthology/W17-6810.

Annegret Habel (1992). *Hyperedge replacement: grammars and languages*. Vol. 643. Springer Science & Business Media.

# Bibliography III

Jorge Mezei and Jesse B. Wright (1967). “Algebraic Automata and Context-Free Sets”. In: *Information and Control* 11, pp. 3–29.