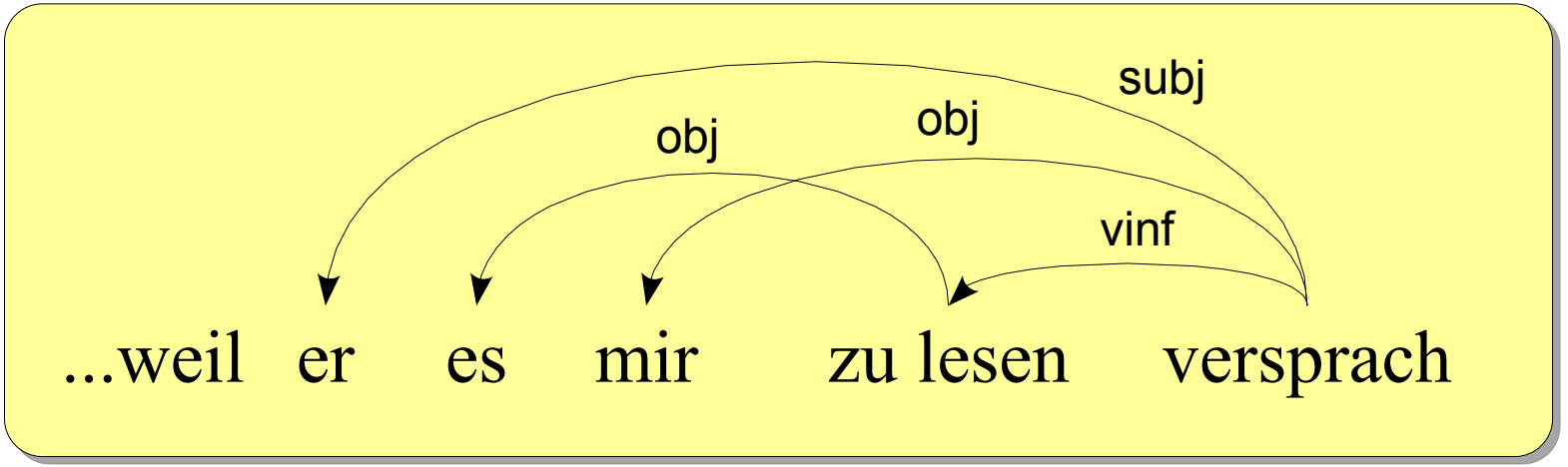
Computational Aspects of Non-Projective Dependency Grammars

by Robert Grabowski
supervised by Marco Kuhlmann

final talk of diploma thesis
Programming Systems Lab, Saarland University, Prof. Gert Smolka
26 January 2006

Declarative Dependency Grammars

- declarative dependency grammars: describing relations between heads and dependents
- well-formedness conditions on structures instead of rules
- related work: TDG, CDG



Declarative Dependency Grammars

- general dependency grammar: very flexible
- large variety of linguistic phenomena easy to encode in grammar
- but: complexity hard to analyse
- difficult to write efficient parsers

Declarative Dependency Grammars

- Gaifman (1965):
  - projective DG are strongly equivalent to lexicalised context-free grammars
  - recognizable in polynomial time

Contents of Thesis

- used Lexicalised Configuration Grammars (LCG) as a flexible framework for non-projective dependency grammars
- analysed computational aspects of LCG grammar parsing
- characterised mildly context-sensitive dependency grammars

Results

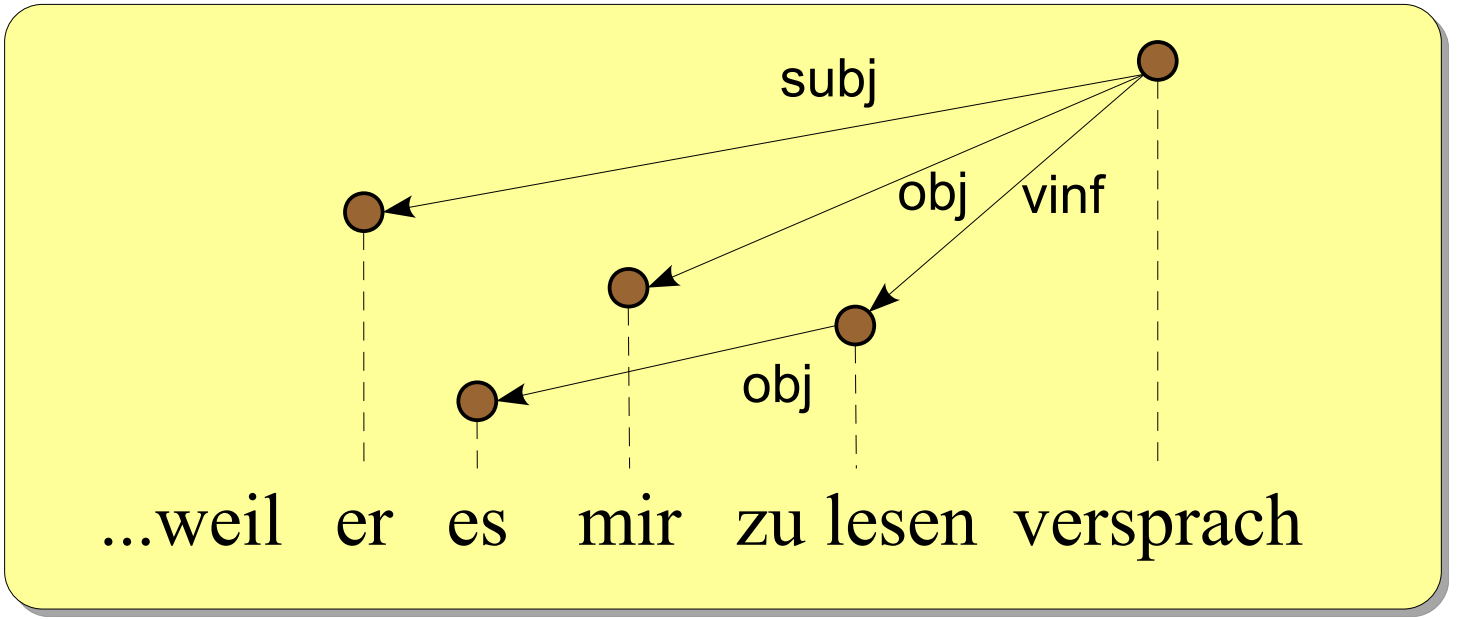
- isolated factors that contribute to parsing complexity
- stated conditions for parsing dependency grammars in polynomial time
- developed a mildly context-sensitive dependency grammar class and parsing schema
- used results to explain complexity of various formalisms encoded in LCG

Overview

- Introduction
- Lexicalised Configuration Grammars
- Computational Factors
- Mildly Context-Sensitive Dependency Languages
- Summary

Lexicalised Configuration Grammars

- LCG talks about labelled drawings
- trees with a total order on the nodes
- may be non-projective
- edges and nodes are labelled



Theories and Grammars

- theory:
  defines a model class and a lexical constraint language
  - lexical constraints are defined locally
  - must be checkable by a computable function
- grammar:
  consists of lexical entries with
  - specifications of incoming and outgoing edge labels
  - local constraints from the constraint language

Example: Lexicalised CFG

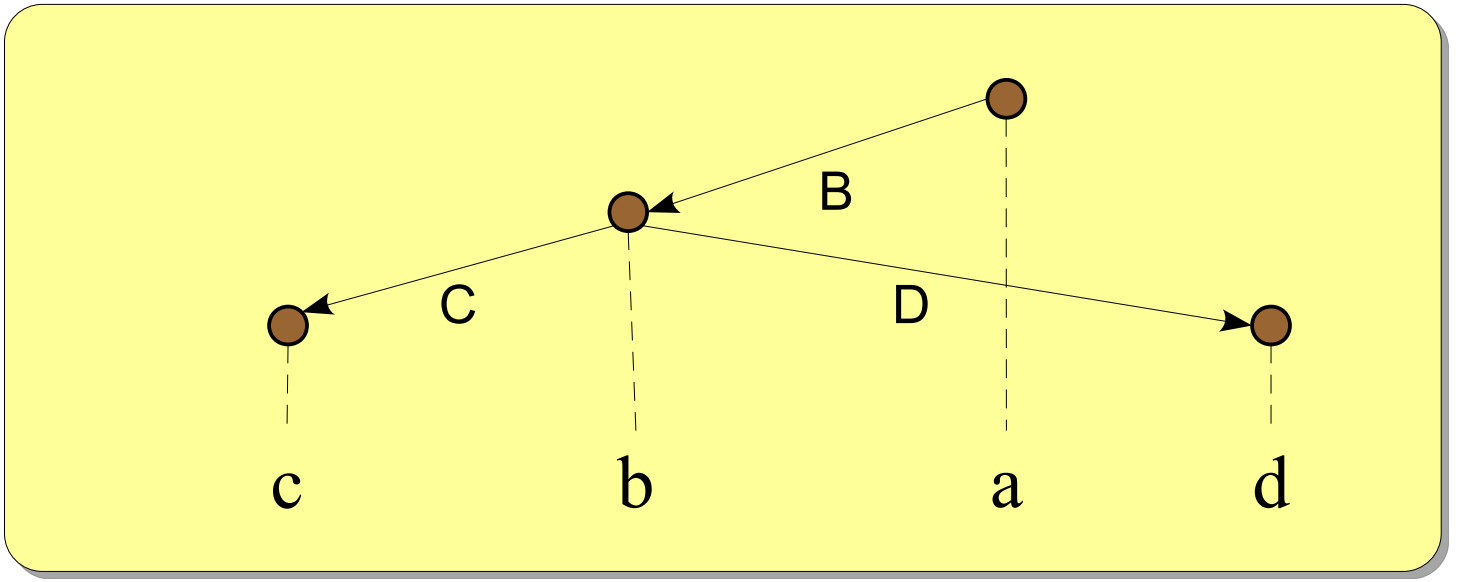
- theory:
  - projective drawings
  - precedence constraint ($\prec$) on child nodes

- grammar:
  - a grammar rule $B \rightarrow C\ b\ D$ is encoded as a lexical entry for *b*:

word: b	in:	{ B }	constraints: C $\prec$ b
	out:	{ C, D }	b $\prec$ D

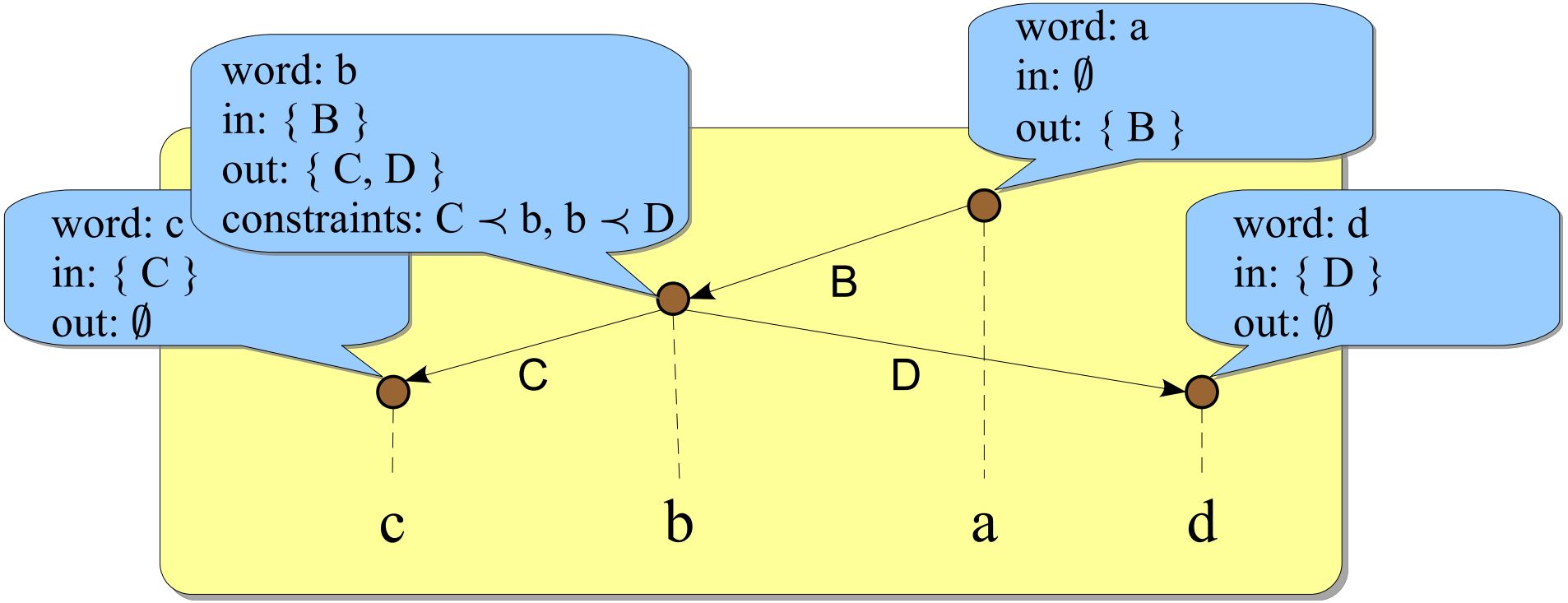
Word Problem

- Is a sentence in the language defined by a grammar?
- also known as membership problem
- can be seen as constraint satisfaction problem



Word Problem

- Is a sentence in the language defined by a grammar?
- also known as membership problem
- can be seen as constraint satisfaction problem



Overview

- Introduction
- Lexicalised Configuration Grammars
- Computational Factors
  - Dimensions of Complexity
  - Chart Parsers
  - Isolating Computational Factors
  - Improving Parsing Efficiency
  - Complexity Matrix
- Mildly Context-Sensitive Dependency Languages
- Summary

Dimensions of Complexity

- deciding the word problem depends on
  - grammar size $|G|$
  - input sentence length n

Parsing Techniques

- LCG parser as constraint solver
- generic algorithms: inherent exponential time
- improvements not to be expected
- factors that influence recognition complexity hard to isolate
- yet: for specific languages, there are efficient parsing techniques

Parsing Techniques

- valency descriptions in LCG form regular tree grammar
- decision:
  exploit tree structure for chart-based parser,
  constraints as filters only
- later:
  - hard-wire constraints
  - "blind" parser that automatically satisfies constraints

Chart Parsers

- efficient parsing technique
- parse items represent already recognized substructures
- inference system
  - defines how to derive new items by combining existing ones
- derived items are written to chart
  - efficient retrieval
- no inference is done twice
- dynamic programming

A Chart Parsing Schema for LCG

$$\frac{\text{<in: } I\text{, out: } \Omega\text{>} \in Lex(w_i)}{\{i\}\text{: <}I\text{, } \Omega\text{>}}\ \text{LOOKUP}$$

$$\frac{s\text{: <}I\text{, } I_1 \uplus \cdots \uplus I_k\text{>} \qquad s_1\text{: <}I_1\text{, } \emptyset\text{>} \quad \cdots \quad s_k\text{: <}I_k\text{, } \emptyset\text{>}}{s \oplus s_1 \oplus \cdots \oplus s_k\text{: <}I\text{, } \emptyset\text{>}}\ \text{COMBINE}$$

- parse item s: $\langle I, \Omega \rangle$ represents partial drawing
  - s: covered sentence positions (span)
  - I: label of incoming edge
  - Ω: labels of outgoing edges (valencies)
  - sentence recognized iff $\{1,...,n\}$: $\langle \emptyset, \emptyset \rangle$ derived

A Chart Parser for LCG

- approximating complexity:
  - combination rule dominates time complexity

$$\frac{P \qquad P_1 \qquad P_2 \qquad \cdots \qquad P_k}{C} \text{ COMBINE}$$

  - at most $|P|^{k+1}$ possibilities for instantiation, where $|P|$ is the number of possible parse items

Isolating Computational Factors

number of possible items depends on:

- span representation
  - worst case: set of integers (exponential in n)
- valency representation
  - worst case: sets (exponential in $|G|$)

Isolating Computational Factors

time complexity is influenced by:

- number of possible items
- number of items to combine
  - worst case: "wide" rules (exponential in $|G|$)
- constraint behavior
  - worst case: no restriction (exponential in $|G|$ and n)

Improving Efficiency

achieving polynomiality with respect to n:

- restrict gap degree to g
  - spans representable as $2*(g+1)$ integers
  - number of possible spans: $n^{2*(g+1)}$

$$\{\ 2,3,4,5,8,9\ \} \rightarrow (2,5,8,9)$$

2^n $\qquad\qquad$ n^4

Improving Efficiency

achieving polynomiality with respect to $|G|$

- combine constant number of items at once
  - store intermediate items in chart
- define a combination order for valencies
  - valencies representable as list

$$\frac{\{1,2\}\ \ \{3\}\ \ \{4,5\}}{\{1,2,3,4,5\}} \rightarrow \frac{\{1,2\}\ \{3\}}{\{1,2,3\}\quad \{4,5\}}\ \{1,2,3,4,5\} \qquad \frac{\{1,2\}\ \{4,5\}}{\{1,2,4,5\}\quad \{3\}}\ \{1,2,3,4,5\} \qquad \frac{\{3\}\ \{4,5\}}{\{3,4,5\}\quad \{1,2\}}\ \{1,2,3,4,5\}$$

Improving Efficiency – A Trade-Off

achieving polynomiality with respect to n and $|G|$:

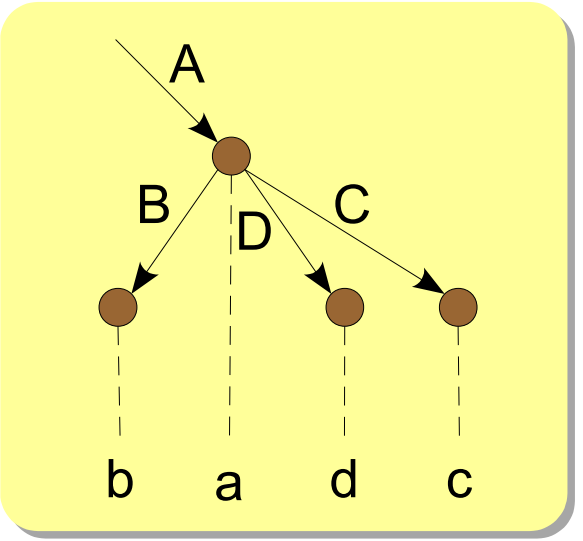
- combine a constant number at once
- a) combine in defined order
  - intermediate results may have unrestricted gap, exponential in n
- b) require gap restriction
  - parser may have to search for a combination order, exponential in $|G|$
  - result: for well-nested drawings, there is always a combination order

Example for Trade-Off

- unordered context-free grammars (UCFG):
  CFG with free valency precedence
  - lexical entry for word *a*:

    in: { A }
    out:{ B, C, D }

  - parse items for drawings
    under B,C,D already in chart



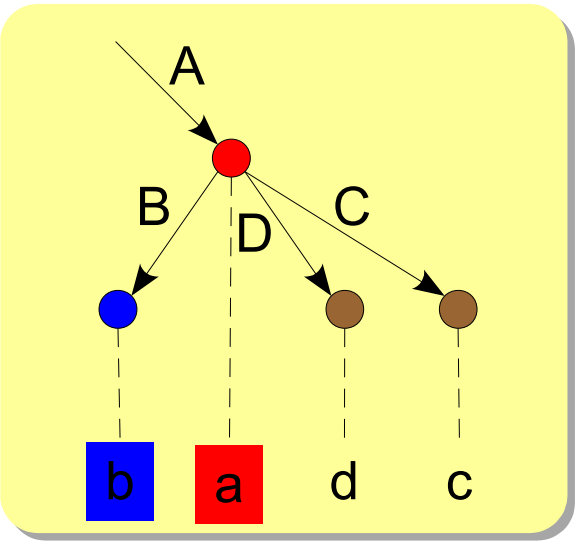
- combine order B,C,D: items with gap degree 1
- or force gap degree 0 for all items:
  have to find a fitting combination order

Example for Trade-Off

- unordered context-free grammars (UCFG):
  CFG with free valency precedence
  - lexical entry for word *a*:

    in: { A }
    out:{ B, C, D }

  - parse items for drawings
    under B,C,D already in chart



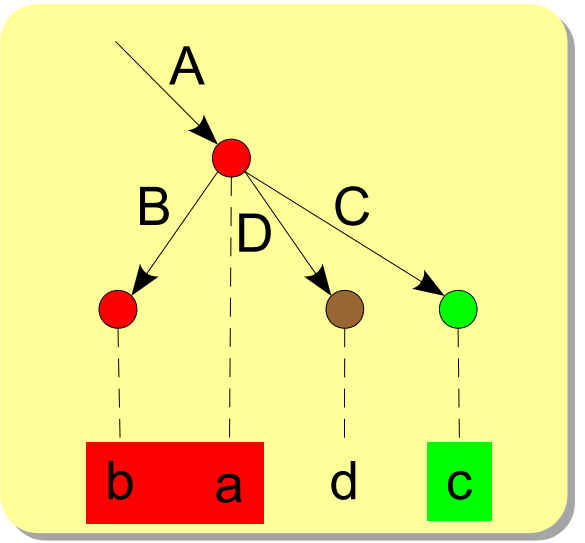
- combine order B,C,D: items with gap degree 1
- or force gap degree 0 for all items:
  have to find a fitting combination order

Example for Trade-Off

- unordered context-free grammars (UCFG):
  CFG with free valency precedence
  - lexical entry for word *a*:

    in: { A }
    out:{ B, C, D }

  - parse items for drawings
    under B,C,D already in chart



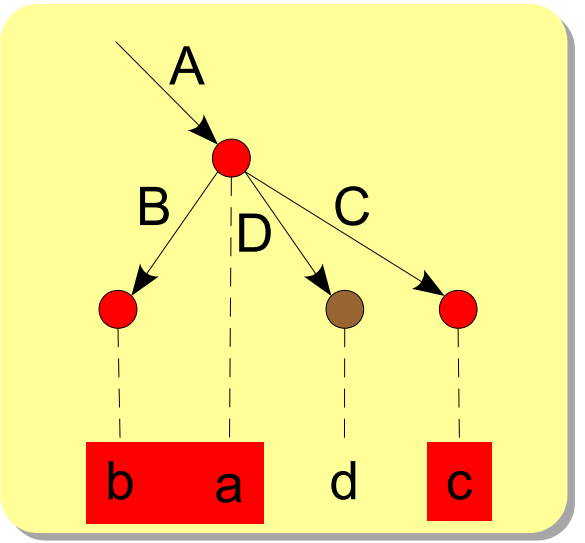
- combine order B,C,D: items with gap degree 1
- or force gap degree 0 for all items:
  have to find a fitting combination order

Example for Trade-Off

- unordered context-free grammars (UCFG):
  CFG with free valency precedence
  - lexical entry for word *a*:

    in: { A }
    out:{ B, C, D }

  - parse items for drawings
    under B,C,D already in chart



- combine order B,C,D: items with gap degree 1
- or force gap degree 0 for all items:
  have to find a fitting combination order

Constraint Complexity

- constraints check all combined subdrawings
- problem: each parse item may represent exponential number of found subdrawings
- achieving polynomiality:
  - constraints may only check parse items directly

Making Use of Constraints

- constraints as filter
- strong constraints: built into parser
  - context-free grammars:
    fully specified local precedence
  - used as combination order, require adjacent spans
- weaker constraints: solutions built into parser
  - grammars with underspecified valency precedences
  - find all constraint solutions: exponential in $|G|$
  - satisfying precedences as combination orders

Complexity Matrix

- encoded formalisms in LCG
- analysed which restrictions can be made on parsing schema without losing completeness
- positioned formalisms in a matrix according to complexity
- schema can explain complexity of formalisms

Complexity Matrix

complexity with respect to...		sentence length *n*: exponential	sentence length *n*: polynomial
grammar size |G|: exponential		LSL SCR	LUCFL
grammar size |G|: polynomial		LUCFL	LCFL

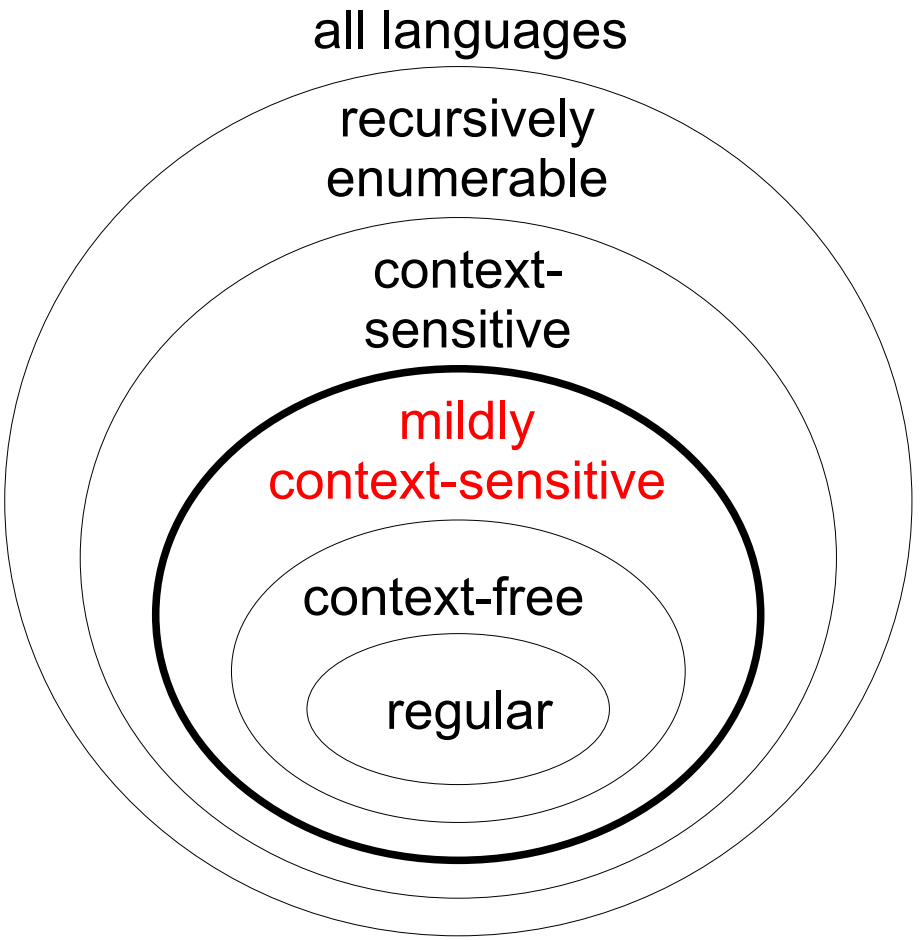
- projective languages are polynomial in *n*
- what about non-projective languages?

Overview

- Introduction
- Lexicalised Configuration Grammars
- Computational Factors
- Mildly Context-Sensitive Dependency Languages
- Summary

Mildly Context-Sensitive Languages

- an important subclass of context-sensitive languages
- believed to sufficiently account for many natural languages
- examples: TAG, MCTAG, CCG, HG

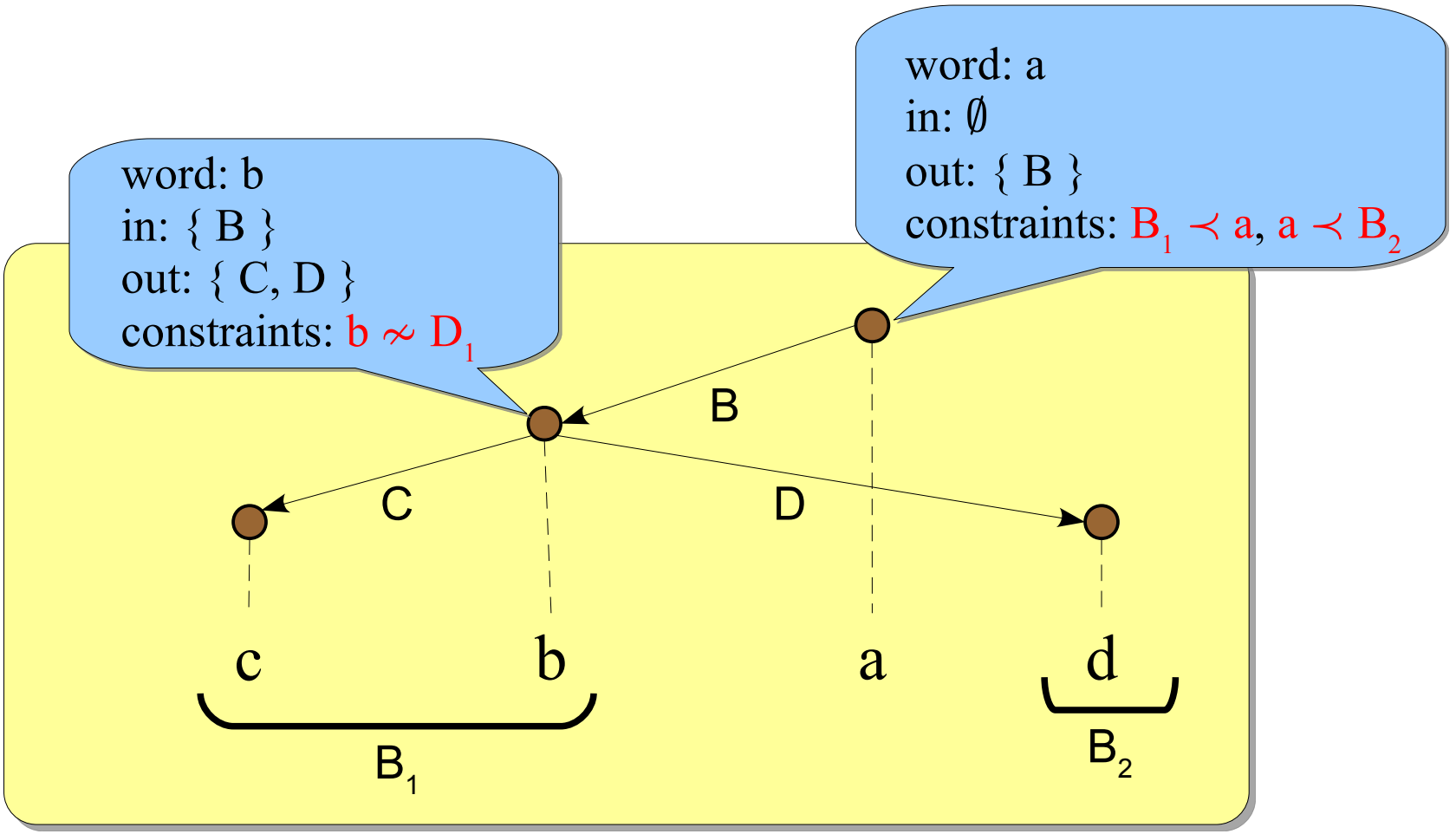


Characterisation

- result:
  characterisation of mildly context-sensitive depedency languages
  - bounded gap degree
  - underlying tree language is regular
  - recognizable in deterministic polynomial time
- no sufficient conditions

MCS-LCG

- developed a class of mildly context-sensitive LCG grammars, and a polynomial time parsing schema



Results

- context-free valency constraints are a special case of valency part constraints
- TAG grammars can be encoded with fully specified valency part precedences

Complexity Matrix

complexity with respect to...		sentence length *n*: exponential	sentence length *n*: polynomial
grammar size |*G*|: exponential		LSL SCR	LUCFL MCSL
grammar size |*G*|: polynomial		LUCFL	LCFL TAG

Overview

- Introduction
- Lexicalised Configuration Grammars
- Computational Factors
- Mildly Context-Sensitive Dependency Languages
- Summary

Summary

- isolated factors that contribute to complexity of recognizing depedency languages
- restricted factors to improve efficiency
- characterised a class of efficiently recognizable non-projective dependency grammars

Future Work

- find fragment for any PTIME language
- encode other grammar formalisms
  - MCTAG
  - CCG
- make use of results in XDG

References

- Robert Grabowski, Marco Kuhlmann, Mathias Möhl: *Lexicalised Configuration Grammars*. In: Second International Workshop on Constraint Solving and Language Processing, Sitges, Spain (2005).
- G. Edward Barton, Jr.: *On the Complexity of ID/LP Parsing*. Computational Linguistics, volume 11, number 4 (1985), 205-218.
- Oliver Suhre: *Computational aspects of a grammar formalism for languages with freer word order*. Diploma thesis, Universität Tübingen (1999).
- Mike Daniels, W. Detmar Meurers: *Improving the efficiency of parsing with discontinuous constituents*. In: Proceedings of the 7th International Workshop on Natural Language Understanding and Logic Programming, Copenhagen (2002), 49–68.
- Denys Duchier, Joachim Niehren: *Dominance constraints with set operators*. In: Proceedings of the First International Conference on Computational Logic, volume 1861 of Lecture Notes in Computer Science, Springer (2000), 326–341.

References

- K. Vijay-Shanker, David J. Weir, Aravind K. Joshi: *Characterizing structural descriptions produced by various grammatical formalisms*. In: 25th Annual Meeting of the ACL, Stanford, USA (1987), 104-111.
- Marcus Kracht: *The Mathematics of Language*. de Gruyter, Berlin (2003).
- James D. McCawley: *Concerning the base component of a transformational grammar*. Foundations of Language, volume 4 (1968), 243–269.
- Hiroshi Maruyama: *Structural Disambiguation with Constraint Propagation*. In: 28th Annual Meeting of the ACL, Pittsburgh, USA (1990), 31-38.