Two-Sided Embedding in Treebanks: A High Coverage (99.994%) Limit

Anssi Yli-Jyrä
Department of Modern Languages, University of Helsinki

(related to a paper in FSMNLP 2017 and to collaboration with Carlos Gómez-Rodrígues in ACL 2017)

Helsinki, 21 September 2017

Motivation

Finite-State Hypothesis

- Yngve 1961: ”A model and an hypothesis for language structure”
- ”The finite state transducer as a theory of language” (Krauwer and des Tombe 1980)
- ”Constraints on multiple center-embedding...” (Karlsson 2007)

Dependency Syntax

- Inherent relevance of CFGs to (noncrossing) dependency graphs (Yli-Jyrä and Gómez-Rodrígues 2017)
- Universal Dependencies is the present day’s ”theory” – not related to competence but representation.

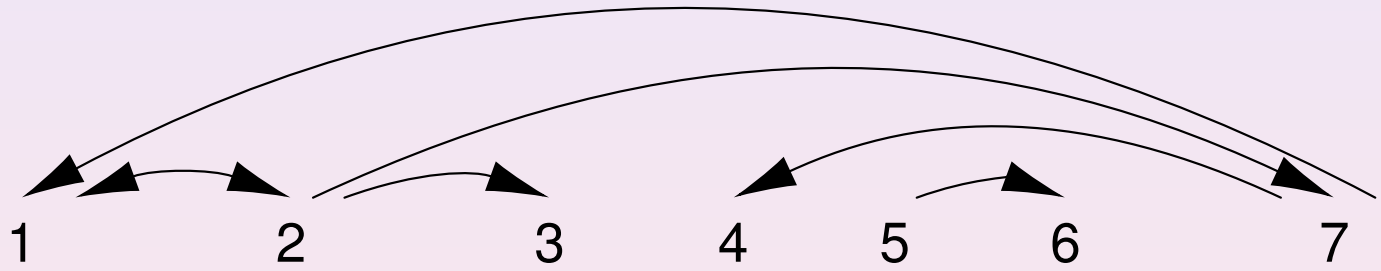
Finite Search Spaces Impact Future AI

- neural network grammars: exact inference and high quality
- interpretable and explainable Artificial Intelligence

Parsing Sentences into Noncrossing Digraphs

NC-DIGRAPH contains ordered noncrossing digraphs (G, E)

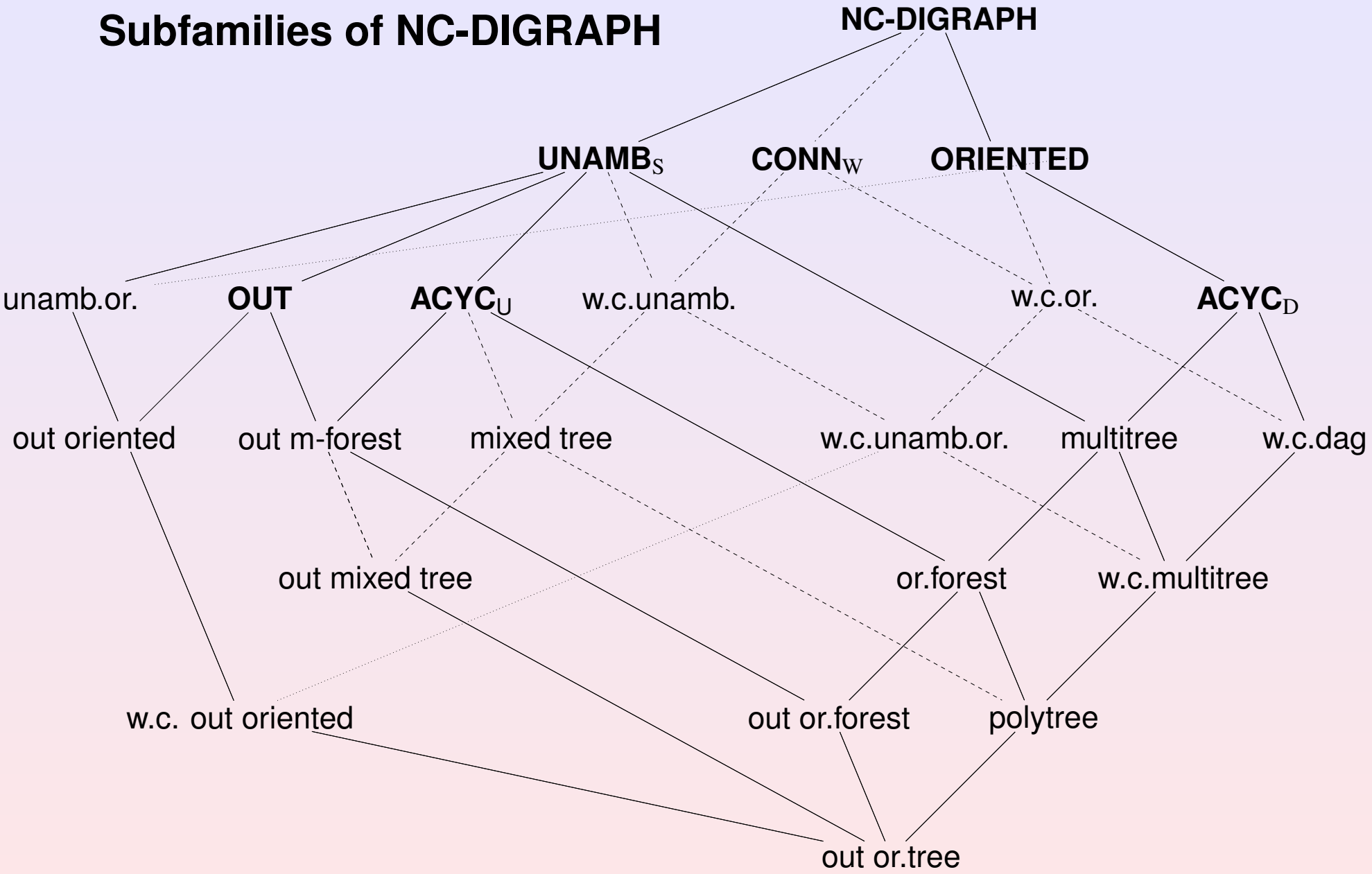
- a mixture of oriented or inverted edges
- no crossing edges when drawn above a line containing the ordered vertices



Digraph families define search spaces for parsing

- a family of noncrossing digraphs: *Family* $\subseteq$ **NC-DIGRAPH**
- parsing returns for the sentence s a digraph $g \in$ *Family*

Subfamilies of NC-DIGRAPH



Two Main Approaches to Dependency Parsing

(1) **Transition-Based:** Use an automaton (transition system) to scan the input buffer and to build a graph $g \in \mathit{Family}$.

(2) **Graph-Based:** Search for an optimal parse under an inference criteria.

- there is a sentence dependent weighting function:

$$\mathit{weight}_s : \textbf{NC-DIGRAPH} \rightarrow \mathbb{R}$$

- parsing as inference: $\arg\max_{g \in \mathit{Family}} \mathit{weight}_s(g)$

Yli-Jyrä and Gómez-Rodríguez (2017)

YG-2017 is a new parsing method that combines graph-based (and transition-based) approach with code theory:

(3) **Code-Theoretic Dependency Parsing:**

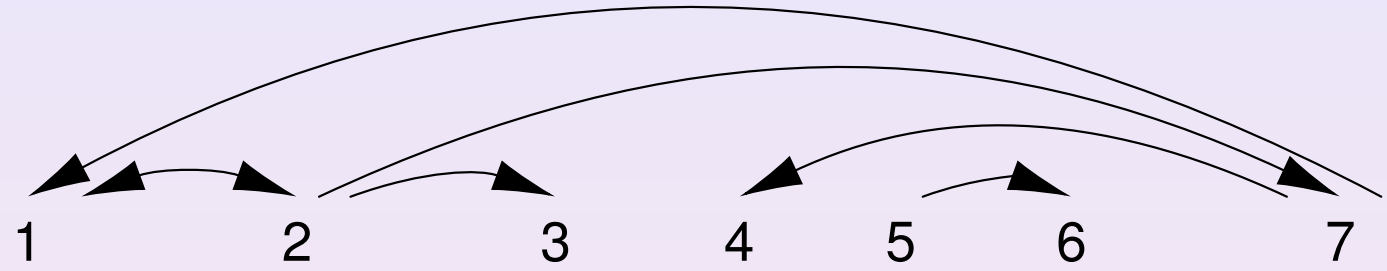
- Encode given *Family* as an unambiguous CFL L_{Family}
- Restrict to the n-node subset given the sentence s
- Search for the best weighted encoded graph using
  (a) tabular parsing ($\approx$ graph-based) or
  (b) shift-reduce automaton ($\approx$ transition-based).

Theorem (YG-2017): For every *Family*, there is an unamb. CFL, L_{Family}, with an **isomorphic encoding function** *enc*:

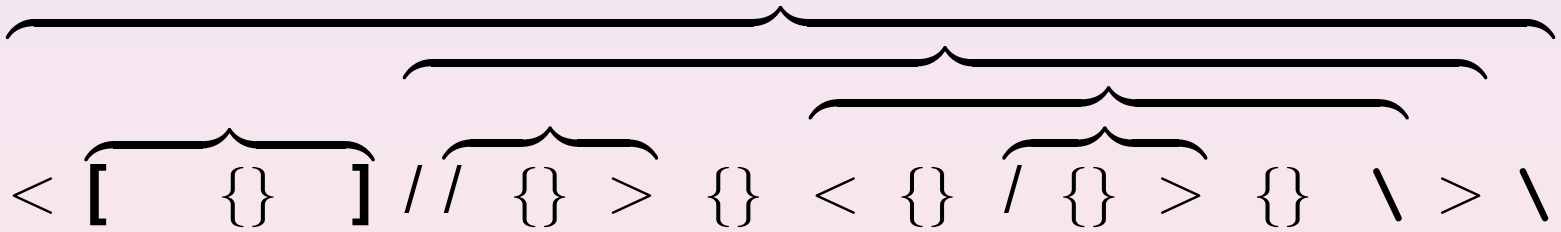
$$enc : Family \rightarrow L_{Family}$$

satisfying $(V, A) = enc^{-1}(enc((V, A)))$, $enc((\{i\}, \{\})) = \epsilon$ and $enc(([i..j], A_1))enc(([j...k], A_2)) = enc(([i..k], A_1 \cup A_2))$.

The Encoding Function



Y-G encoding:



with uniform left brackets (in the current paper):

[[{}] [[{} > {} [{} [{} > {} \ > \

Arc-Factored Parsing in YG-2017

The weight of a graph is the sum of the arc weights:

$$weight_s(\underbrace{V, A}_{\text{a digraph}}) = \sum_{(u,v)\in A} w_s(\underbrace{u, v}_{arc})$$

Inference over parses in $W^n = \overline{B}^*(\{\}\overline{B}^*)^{n-1}$

- $L_{Family} \cap W^n = h(D \cap R \cap W^n) = h'(D' \cap R' \cap W^{n\prime})$

  where ' adds position indices: $[_1\,[_1\{\}]_2\,[_2\{\}\{\}]_4]_4$

- replace D' with a weighted language D'':

  $S \to \epsilon \mid \{\};\ S \xrightarrow{w_{12}} [_1 S]_2 S;\ S \xrightarrow{w_{13}} [_1 S]_3 S;\ S \xrightarrow{w_{14}} [_1 S]_4 S;\ S \xrightarrow{w_{23}} [_2 S]_3 S;\ S \xrightarrow{w_{24}} [_2 S]_4 S;\ S \xrightarrow{w_{34}} [_3 S]_4 S.$

- construct a WCFG for $D'' \cap R' \cap W^{n\prime}$

Goal: Linear Time Inference

- arc weights computed in $O(n^2)$ time
- construction of WCFG in $O(n^3)$ time

To obtain a **linear-time parser**, we have to

- reduce the weight computation time to $O(n)$
- reduce the inference time to $O(n)$ $\Longleftarrow$ **START HERE**

Replace each *Family* with a **regular subset approximation**.

- exact linear decoder, not just beam search
- best path in an WFST over the max-plus semiring

But what is a **good** regular approximation?

Requirements

Definition

A **good subset approximation** of a parse search space has the following three properties:

- **Complete Core.** A language-independent and generic approximation for the set of digraphs should contain all digraphs over a small number ($n \leq 7$) of tokens.
- **Convenient Size.** The amount of the memory needed to carry out inference over long sentences should not stretch the limits of a convenient implementation.
- **Good coverage.** The approximation should have a good coverage of the existing analyses in treebanks.

Size as $|\mathcal{M}_A|$ = # of states in the minimal DFA for A.
Coverage$_1$ as % of the search space
Coverage$_2$ as % of the Universal Dependencies treebank.

Baseline

Intersection as the product: $|\mathcal{M}_{A\cap B}| \leq |\mathcal{M}_A| \times |\mathcal{M}_B|$

n	encoded digraphs	$\lvert\mathcal{M}_{W^n}\rvert$	$\lvert\mathcal{M}_{L_{\text{NC-DIGRAPH}}\cap W^n}\rvert$
1	1	2	1
2	4	4	12
3	64	6	80
4	1,792	8	490
5	62,464	10	2,952
6	2,437,120	12	27,040
CORE $\Rightarrow$ 7	101,859,328	14	106,372

Complete Core: requires $\geq 106,372/14 = 7,598$ states.
Inconvenient: $\mathcal{M}_{L_{\text{NC-DIGRAPH}}\cap W^{500}}$ has $\geq 7,598,000$ states.
Bad Coverage: manages only 6 levels of nested edges.

Weak Dependency Bracketing

Edges that share the same endpoint use a *superbracket*:

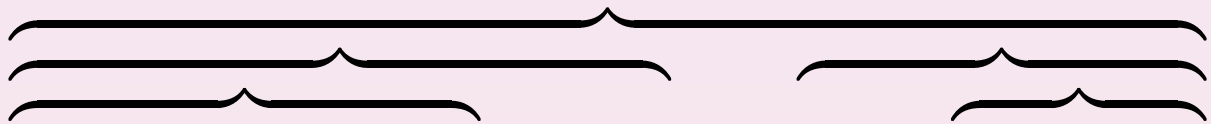
[[[[[{}] {}] {}] {}] {}]

[! {}] {}] {}] {}] {}] !

Both ends of the outermost edges can be shared:

[! {}] {}] {} [{} [{}] !

The edges can have an orientation or remain inverted:

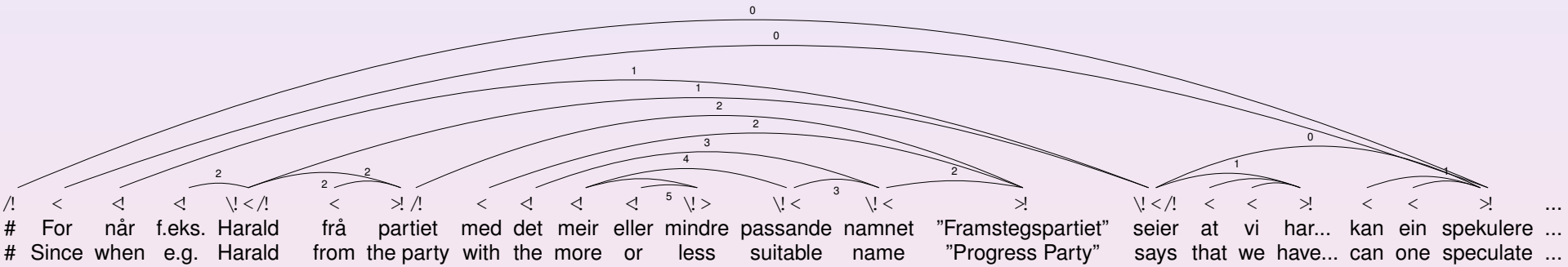


< ! {} > {} / {} < {} \ {} \ !

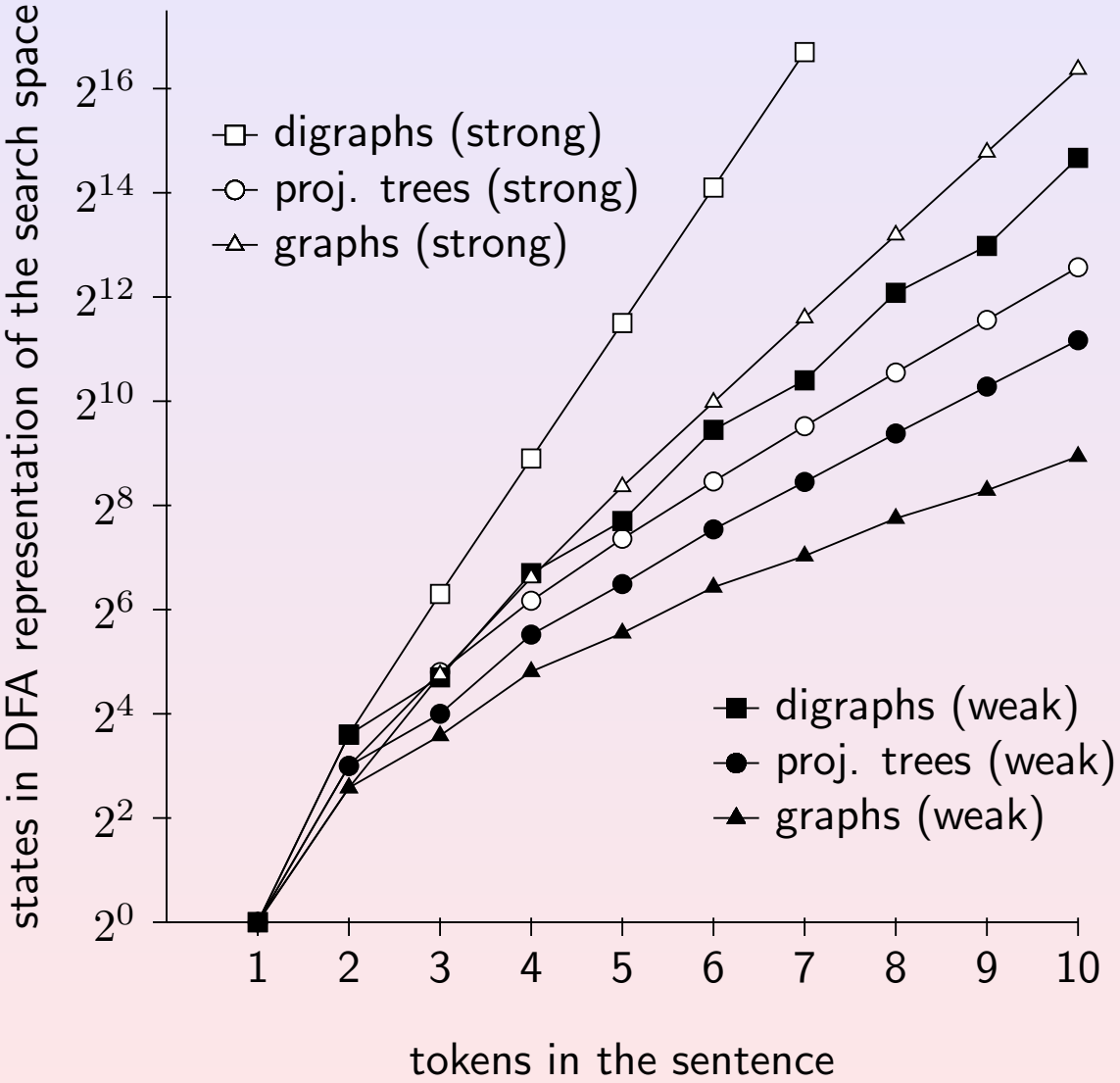
And the opening superbrackets can be simplified:

[! {} > {} / {} < {} \ {} \ !

Example


0
0
1
1
2
2
3
4
5
2
2
2
2
3
0
1
1
/! < <! <! \! < /! < >! /! < <! <! <! \! > \! < \! < >! \! < /! < < >! < < >! ...
For når f.eks. Harald frå partiet med det meir eller mindre passande namnet ”Framstegspartiet” seier at vi har... kan ein spekulere ...
Since when e.g. Harald from the party with the more or less suitable name ”Progress Party” says that we have... can one speculate ...

State Complexity of Strong/Weak Bracketing



n	encoded graphs	states
10	21 292 032	490
19	29 312 424 612 462 592	12 395
20	314 739 971 287 154 688	18 276
21	3 393 951 437 605 044 224	24 925
30	$\approx 7.681 \cdot 10^{27}$	589 598

Subset Approximation

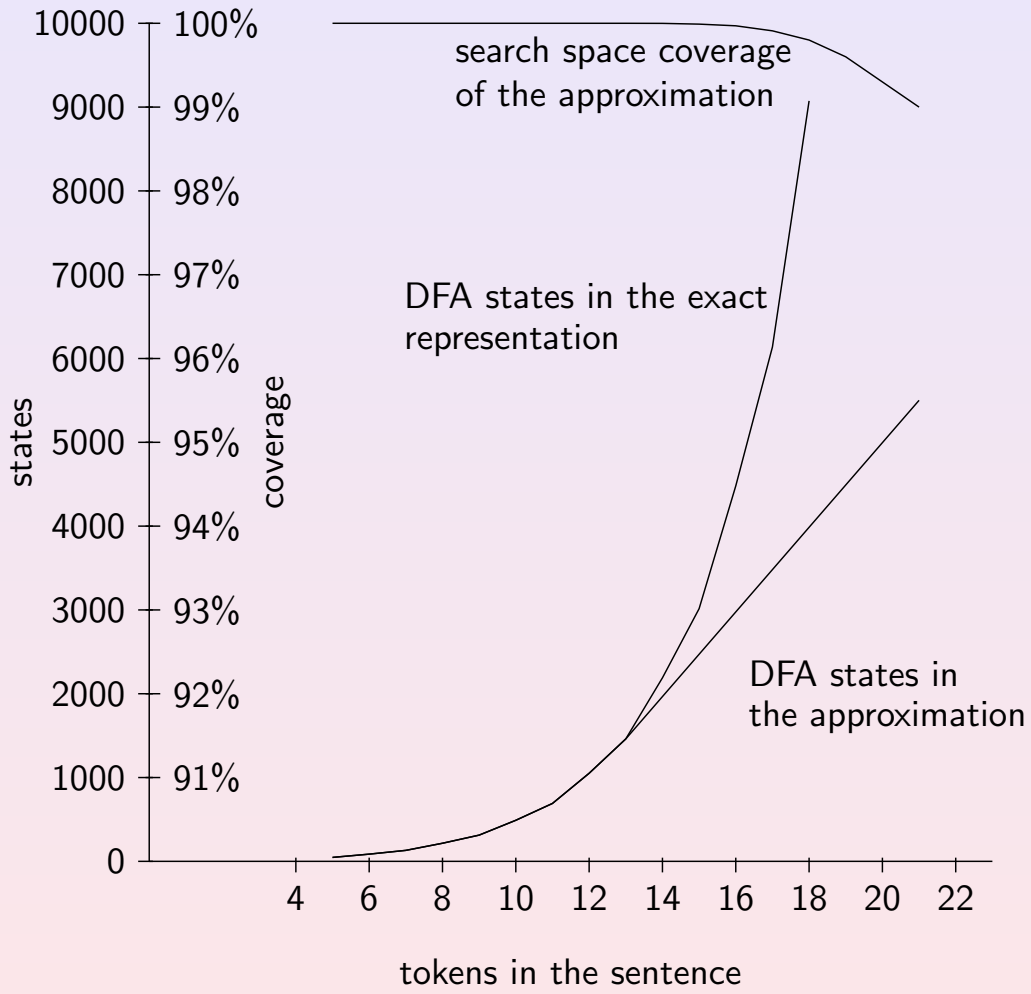
Exact CFG for weak bracketing:

$$S' \rightarrow \quad (\{\} \mid S)^*$$
$$S \rightarrow \quad [! \left(\{\} \mid T(\{\} \mid S)^* U\right)]! \text{ where}$$
$$T = \{\} \mid \{\}(\{\} \mid S)^*] \left(([\{\} \mid S)^+]\right)^*$$
$$U = \{\} \mid \left([(\{\} \mid S)^+\right)^* [(\{\} \mid S)^* \{\}$$

Regular subset approximation:
Limit recursion to 5 nested levels of S.
This requires only 442 DFA states.

For W^{21}, the intersection has only 5501 states and still covers 99% of all the exact search space.

When intersected with W^{500} the bounded-depth search space requires $\leq$ 442,000 states.



Experiments with Data

Data: the Universal Dependencies treebanks (v.2)

- the multilingual data set with .6 million sentences
- nonprojective dependency trees
- convert to noncrossing (planar) graphs

Models of data:

- P(missed tree | depth)
- P(sentence lengths)
- P(missed tree | length)
- P(length | depth)
- P(dept, length)

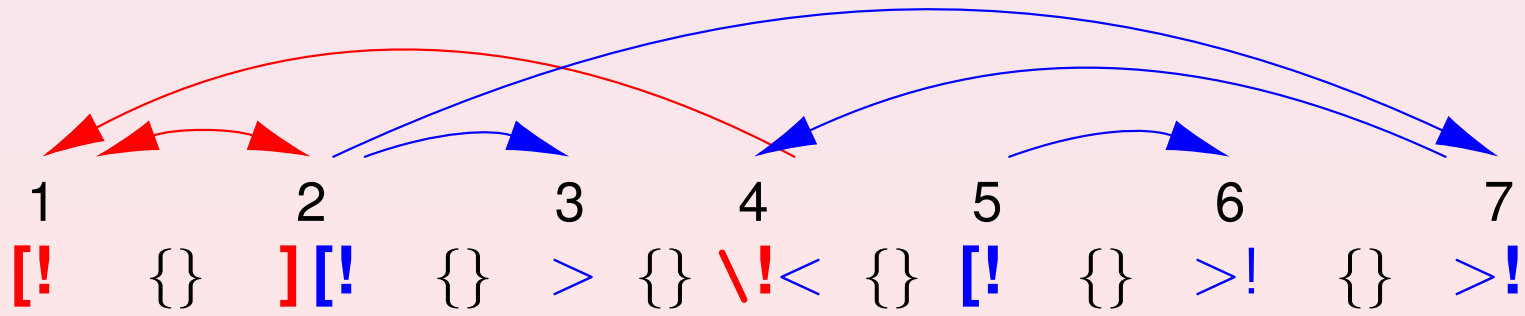
The scripts: `https://github.com/amikael/depconv`

Planarisation

i.e. conversion of (di)graphs into (di)graphs that whose edges do not crossing these are drawn above a line.

Approaches:

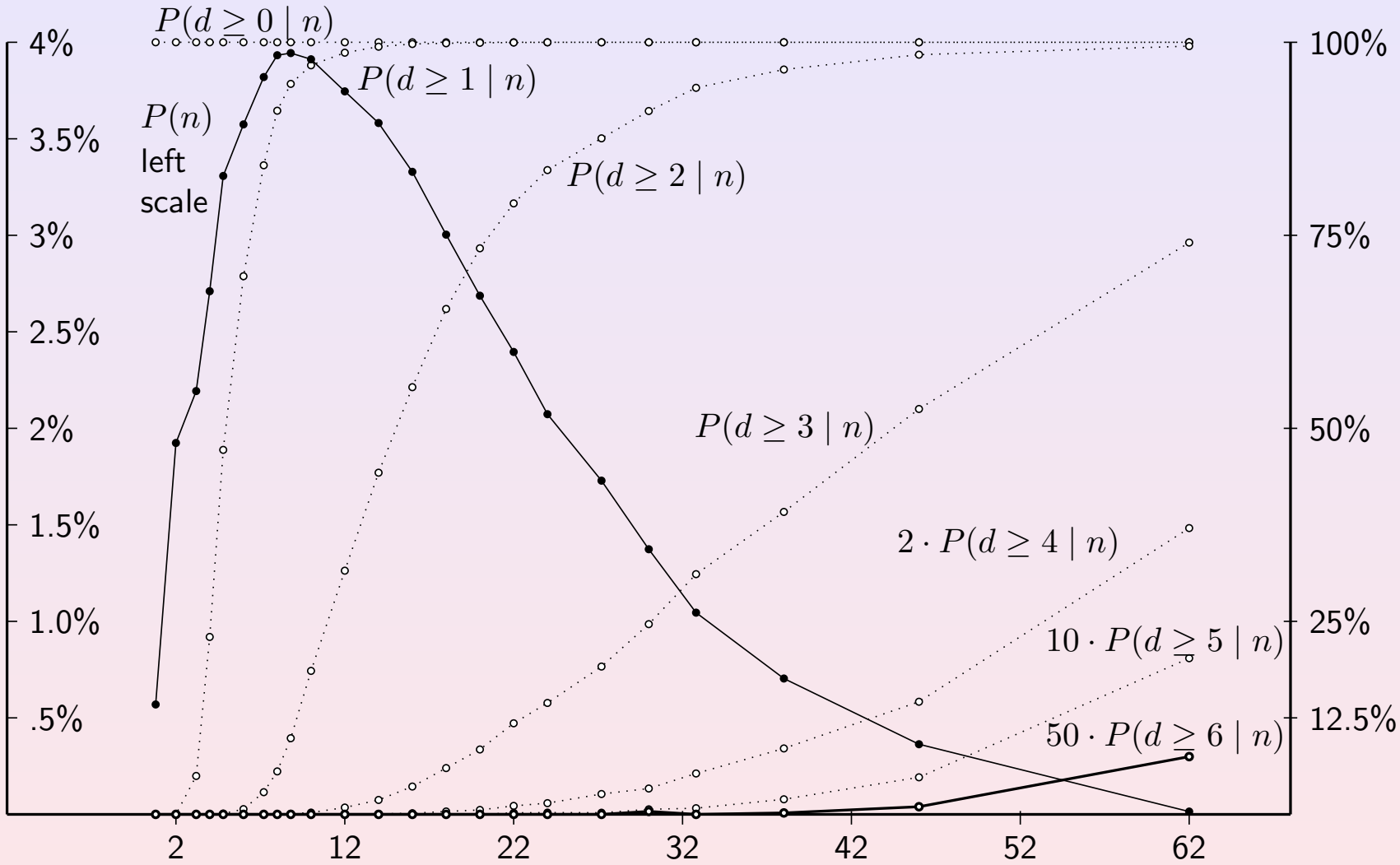
- heuristic lifting
- preserving as many original arcs as possible (Goldberg)
- minimize lifts
- permute words to canonical word order
- bracketing edges and rematching the brackets as edges (here)



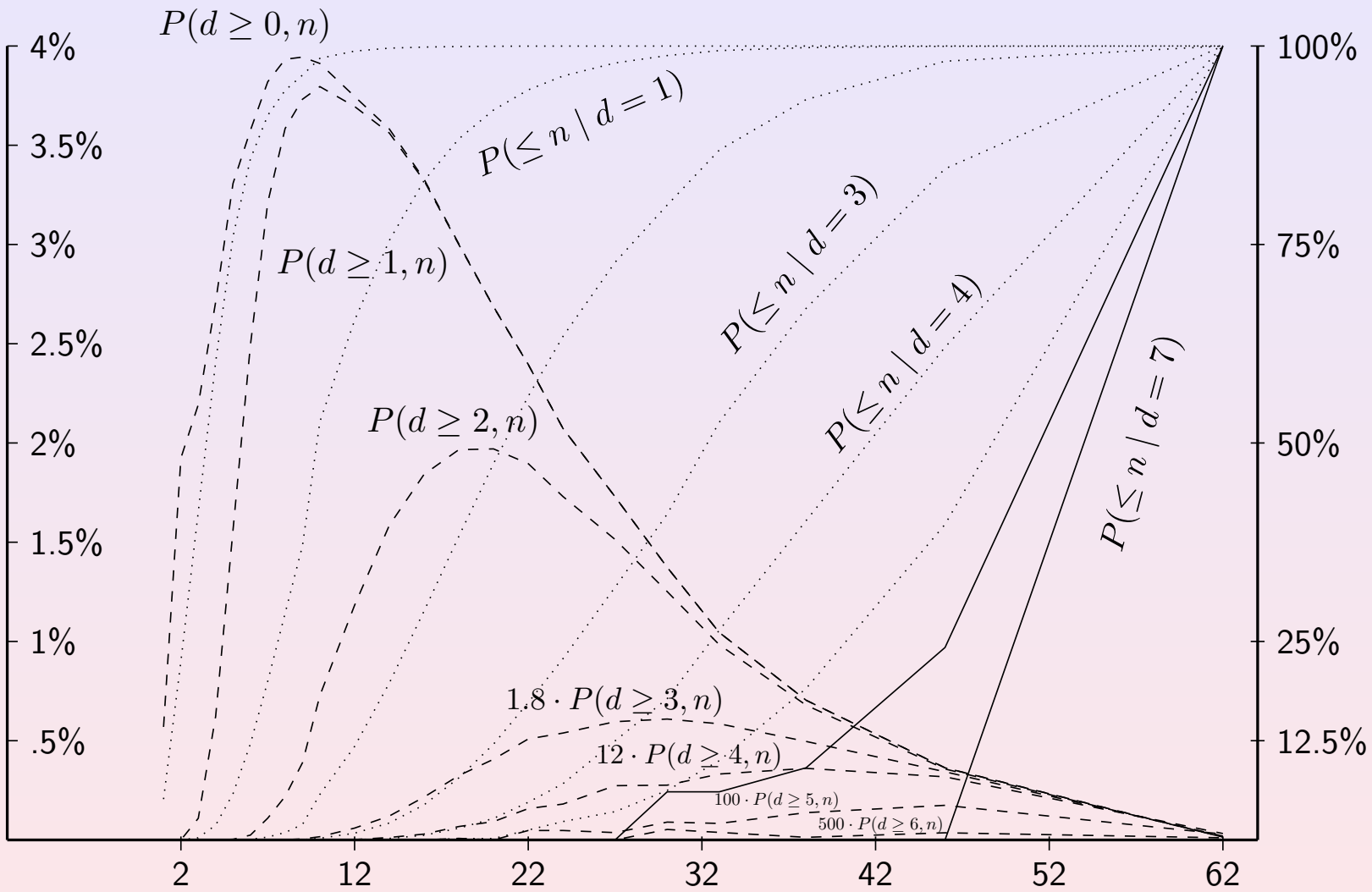
Data Coverage (Universal Dependencies)

lang	N	0	1	2	3	depth 4	depth 5	depth 6	depth 7
Arabic	26722	4%	21%	65%	94.4%	99.64%	99.99%	+0.011% (3)	
Czech	102660	11%	43%	87%	98.5%	99.89%	99.99%	+0.010% (10)	
German	14917	2%	43%	88%	98.6%	99.91%	99.97%	+0.027% (4)	
English	19785	17%	54%	91%	99.2%	99.96%	100.00%		
Spanish	31546	2%	25%	77%	97.2%	99.79%	100.00%	+0.003% (1)	
Finnish	30437	30%	77%	97%	99.6%	99.96%	99.99%	+0.007% (2)	
French	19294	3%	38%	88%	98.9%	99.94%	99.99%	+0.005% (1)	
Hindi	14963	0%	37%	83%	98.0%	99.90%	99.98%	+0.020% (3)	
Hungarian	1351	1%	23%	69%	94.9%	98.96%	99.78%	+0.148% (2)	+0.074% (1)
Portuguese	19765	5%	32%	81%	98.0%	99.83%	99.99%	+0.005% (1)	
Romanian	8795	1%	26%	85%	98.5%	99.87%	99.99%	+0.011% (1)	
Russian	59827	13%	74%	97%	99.8%	99.98%	100.00%	+0.002% (1)	
Turkish	4660	22%	71%	93%	98.9%	99.89%	99.98%	+ 0.021% (1)	
Chinese	4497	0%	20%	70%	94.5%	99.51%	99.96%	+ 0.044% (2)	
others	...	...	...	...	...	...			
	630518	11%	50%	88%	98.7%	99.91%	99.994%	+ 0.005% (33)	+ 0.0003% (2)

Distributions of Depth



Distributions of Length



Conclusions

- Streamlined search space representation
  - weak bracketing for high degree nodes
  - uniform [for all edge types
- Approximated search space for syntactic graphs
  - small FSA approximation (442 states)
  - covers 99.994% (of planarised UD trees)
- The exponential parse ”forest” with only $O(n)$ weights
- Further work
  - towards complete parsing in $O(n)$:
    - we can almost encode crossing digraphs
    - use word embeddings and neural contexts
  - a finite-state subset of the ontology of Y&G (2017)