Discontinuous Parsing with an Efficient and Accurate DOP Model

Andreas van Cranenburgh Rens Bod

Huygens ING
Royal Netherlands Academy of Arts and Sciences

Institute for Logic, Language and Computation
University of Amsterdam

November 27, 2013

IWPT 2013, Nara, Japan

This talk

Parsing with . . .

- discontinuous constituents:
  Linear Context-Free Rewriting Systems (LCFRS)
- treebank fragments:
  Data-Oriented Parsing (DOP)
  Tree-Substitution Grammar (TSG)

Discontinuous constituents

Example:

- Why did the chicken cross the road?
- The chicken crossed the road to get to the other side.

Discontinuous trees

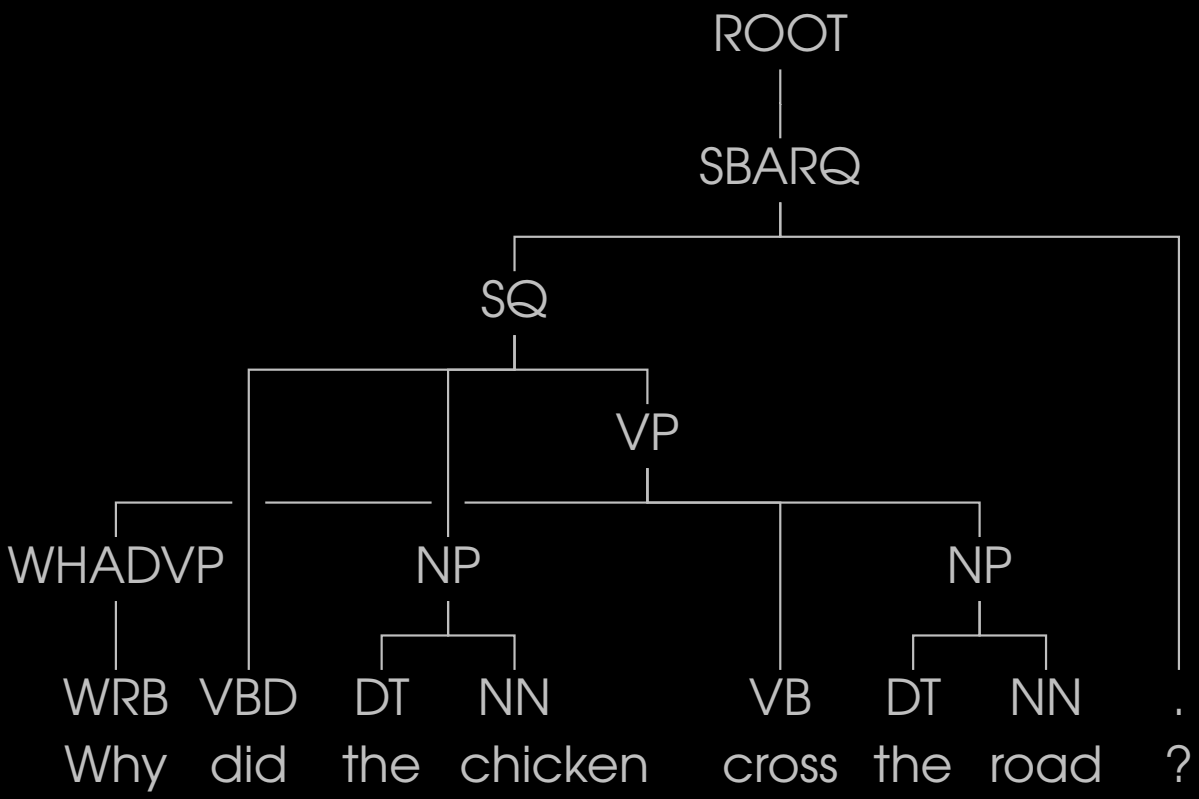


Figure : A discontinuous tree not found in the Penn treebank.

Discontinuous constituents

Motivation:

- Flexible word-order
- Capture argument structure
- Combine information from constituency & dependency structures
- Information is available in treebanks (German, Dutch, English after conversion).

Discontinuous trees

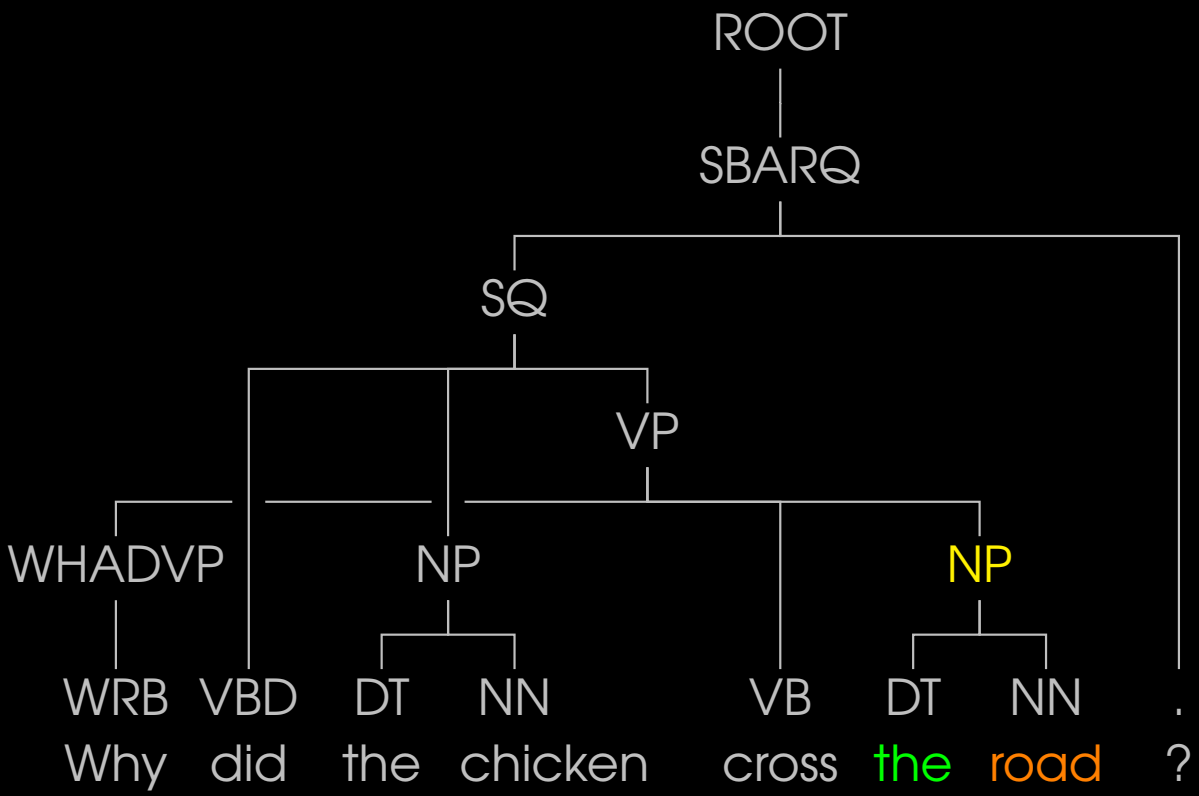


Figure : A discontinuous tree not found in the Penn treebank.

Context-Free Grammar (CFG)

$NP(ab) \rightarrow DT(a)\ NN(b)$

Discontinuous trees

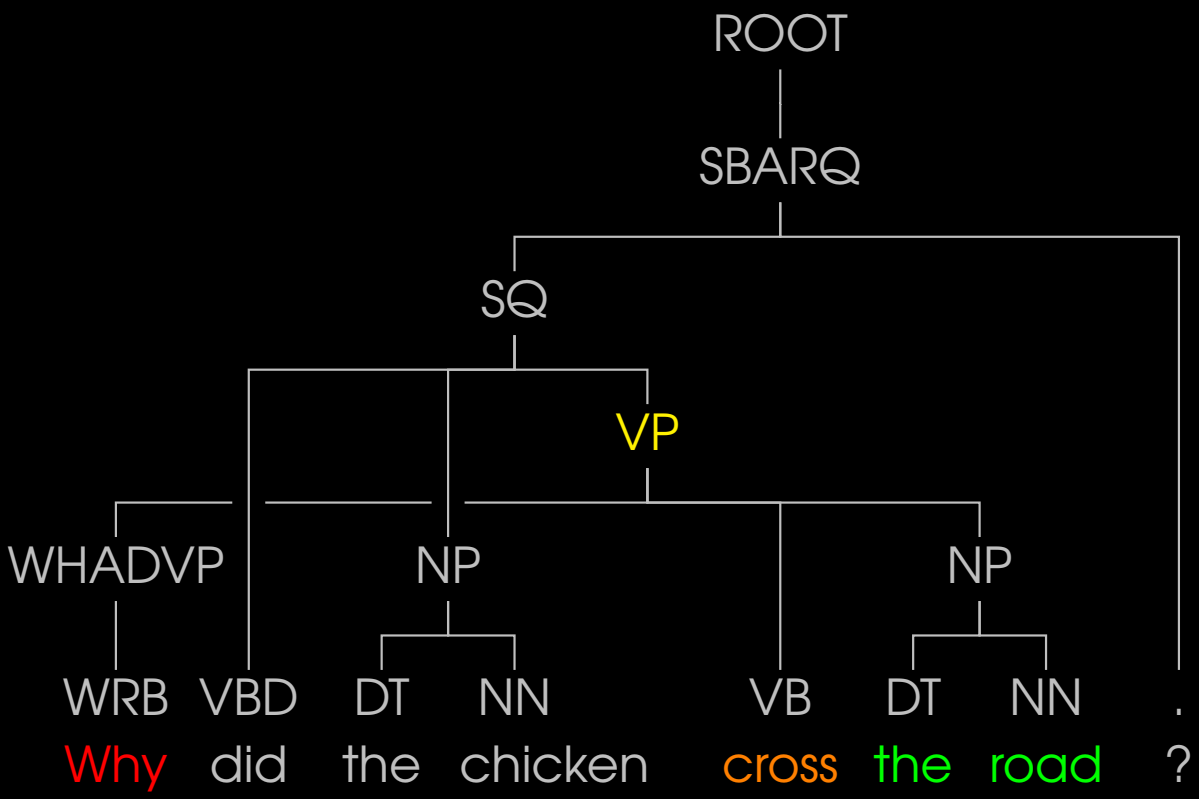


Figure : A discontinuous tree not found in the Penn treebank.

Linear Context-Free Rewriting System (LCFRS)

$$VP_2(a, bc) \rightarrow WHADVP(a)\ VB(b)\ NP(c)$$

Discontinuous trees

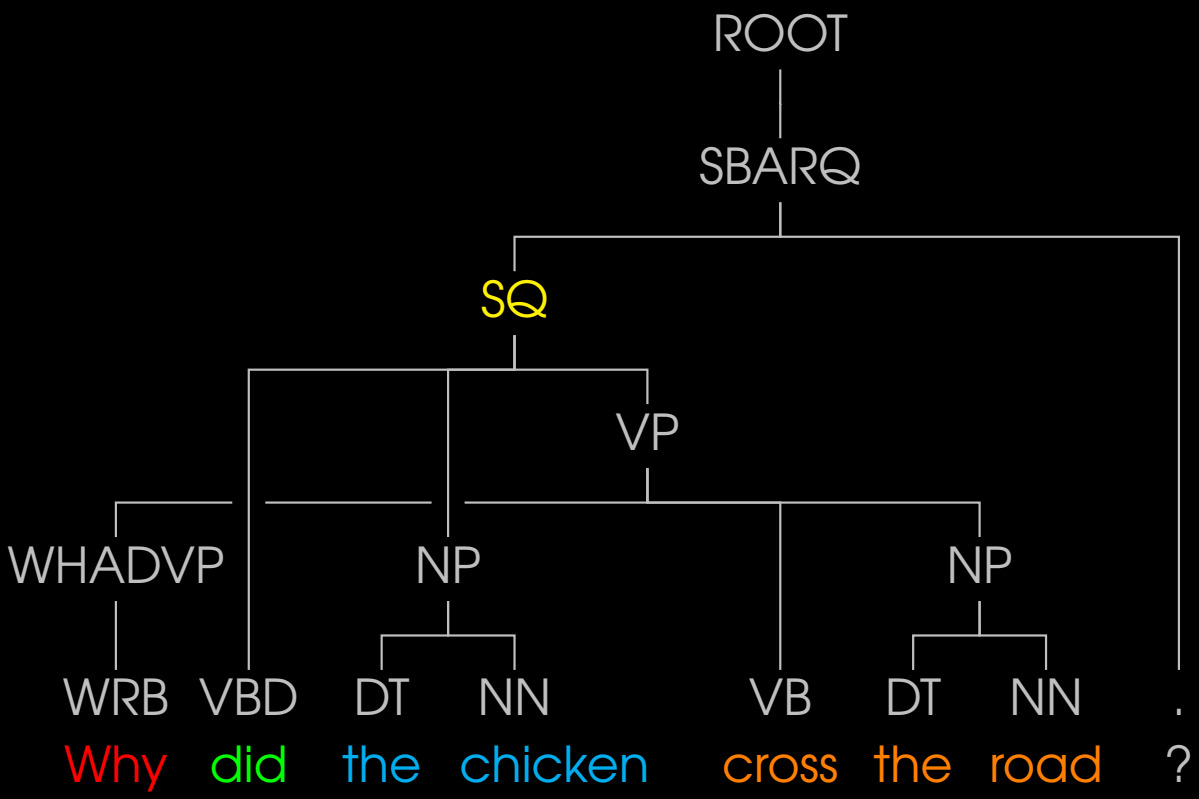


Figure : A discontinuous tree not found in the Penn treebank.

Linear Context-Free Rewriting System (LCFRS)

$$VP_2(a, bc) \rightarrow WHADVP(a)\ VB(b)\ NP(c)$$
$$SQ(abcd) \rightarrow VBD(b)\ NP(c)\ VP_2(a, d)$$

Linear Context-Free Rewriting Systems

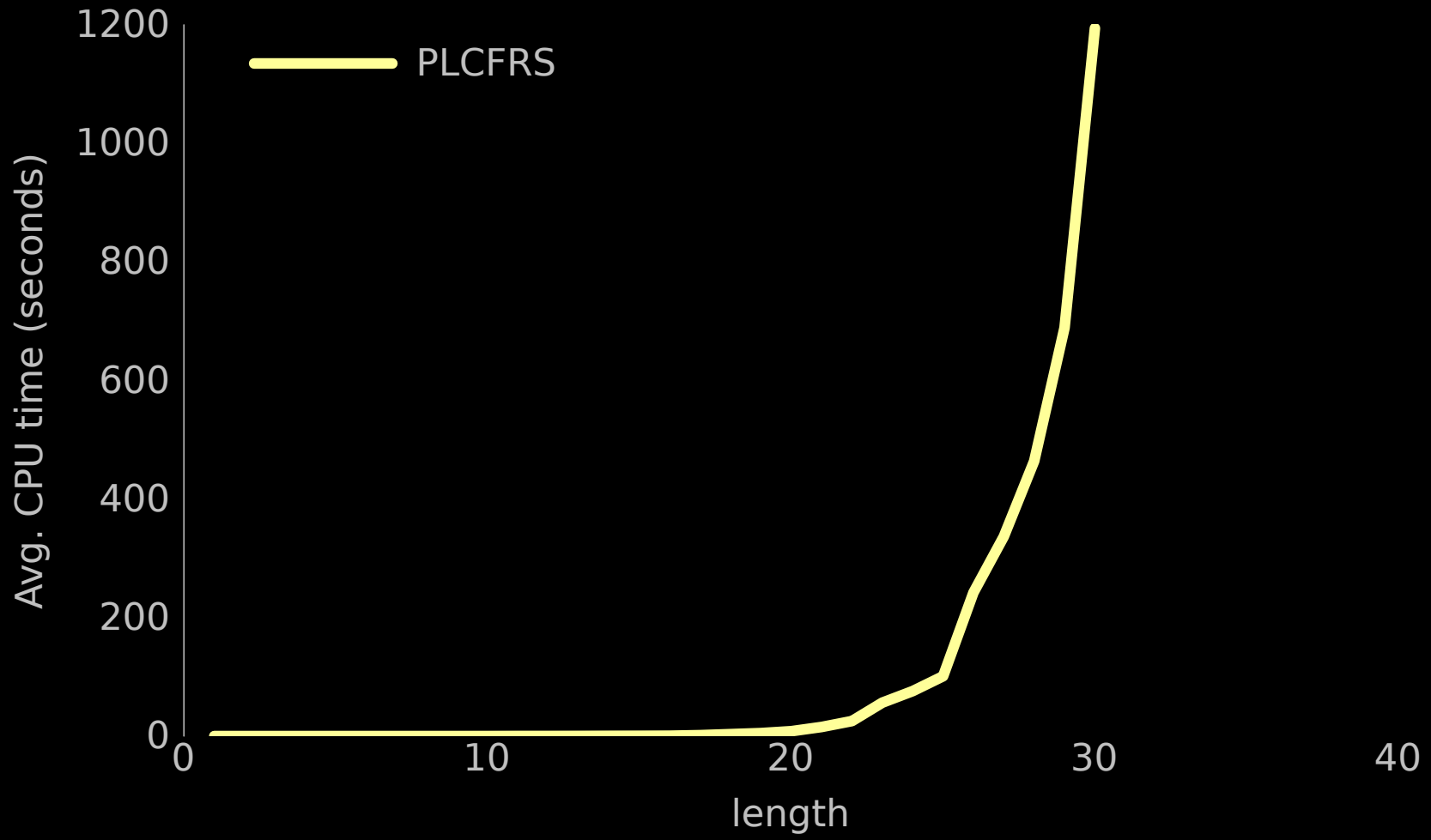
- Mildly context-sensitive grammar formalism
- Can be parsed with tabular parsing algorithm
- Agenda-based probabilistic parser for LCFRS (Kallmeyer & Maier 2010);
  extended to produce k-best derivations
- Parsing a binarized LCFRS has polynomial complexity:

$$\mathcal{O}(n^{3\varphi})$$

where φ is the maximum number of components covered by a non-terminal (fan-out).

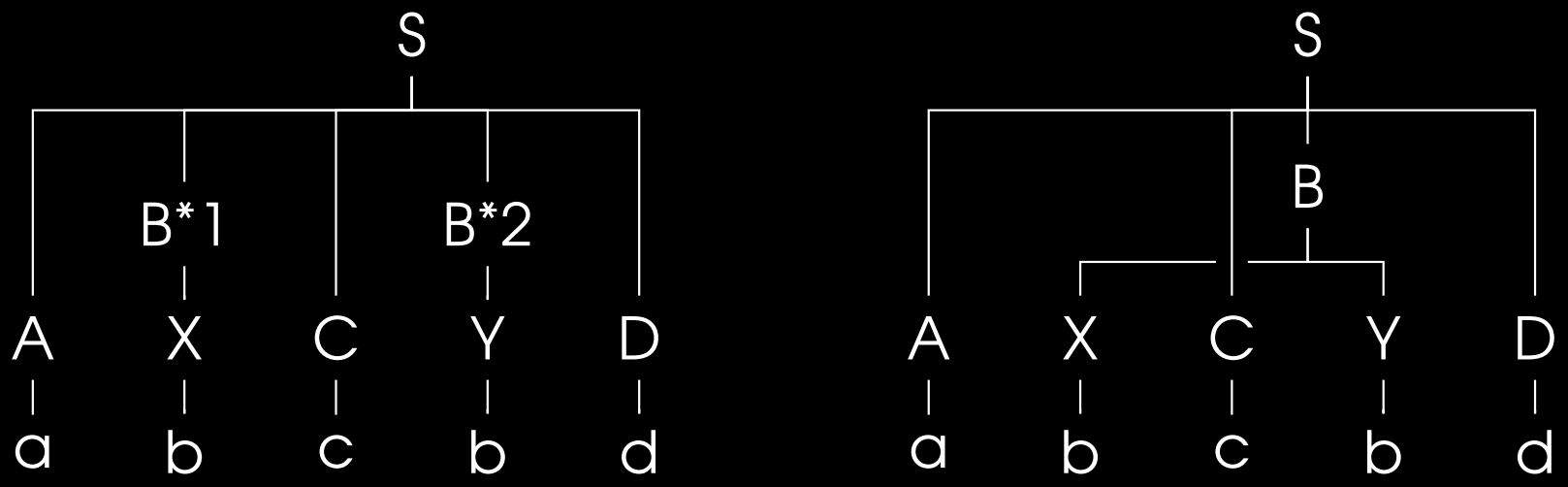
Kallmeyer & Maier (2010). Data-driven parsing with probabilistic linear context-free rewriting systems.

But . . .



Negra dev. set, gold tags

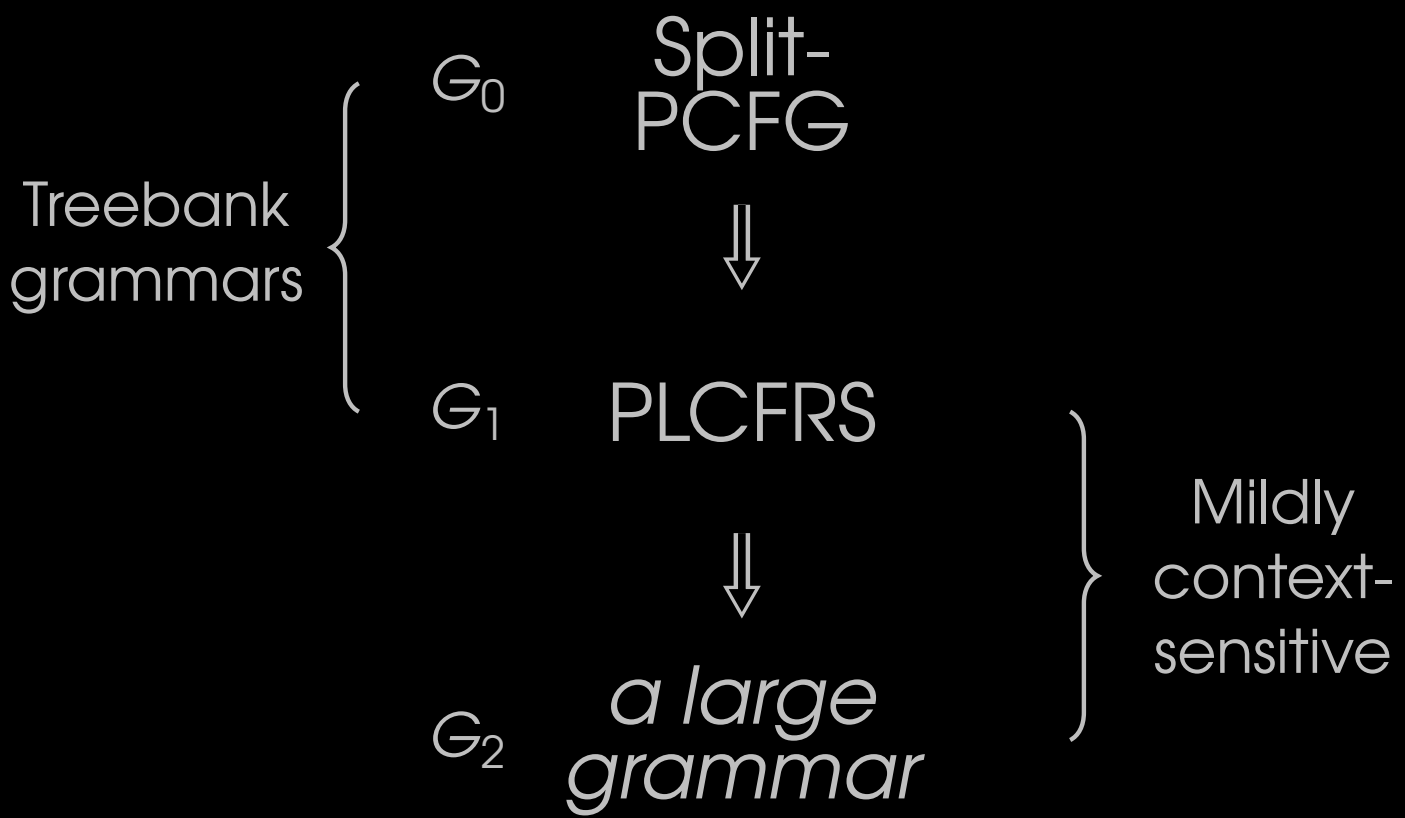
PCFG approximation of PLCFRS



- Transformation is reversible
- Increased independence assumption:
  $\Rightarrow$ every component is a new node
- Language is a superset of original PLCFRS
  $\Rightarrow$ coarser, overgenerating PCFG (`split-PCFG')

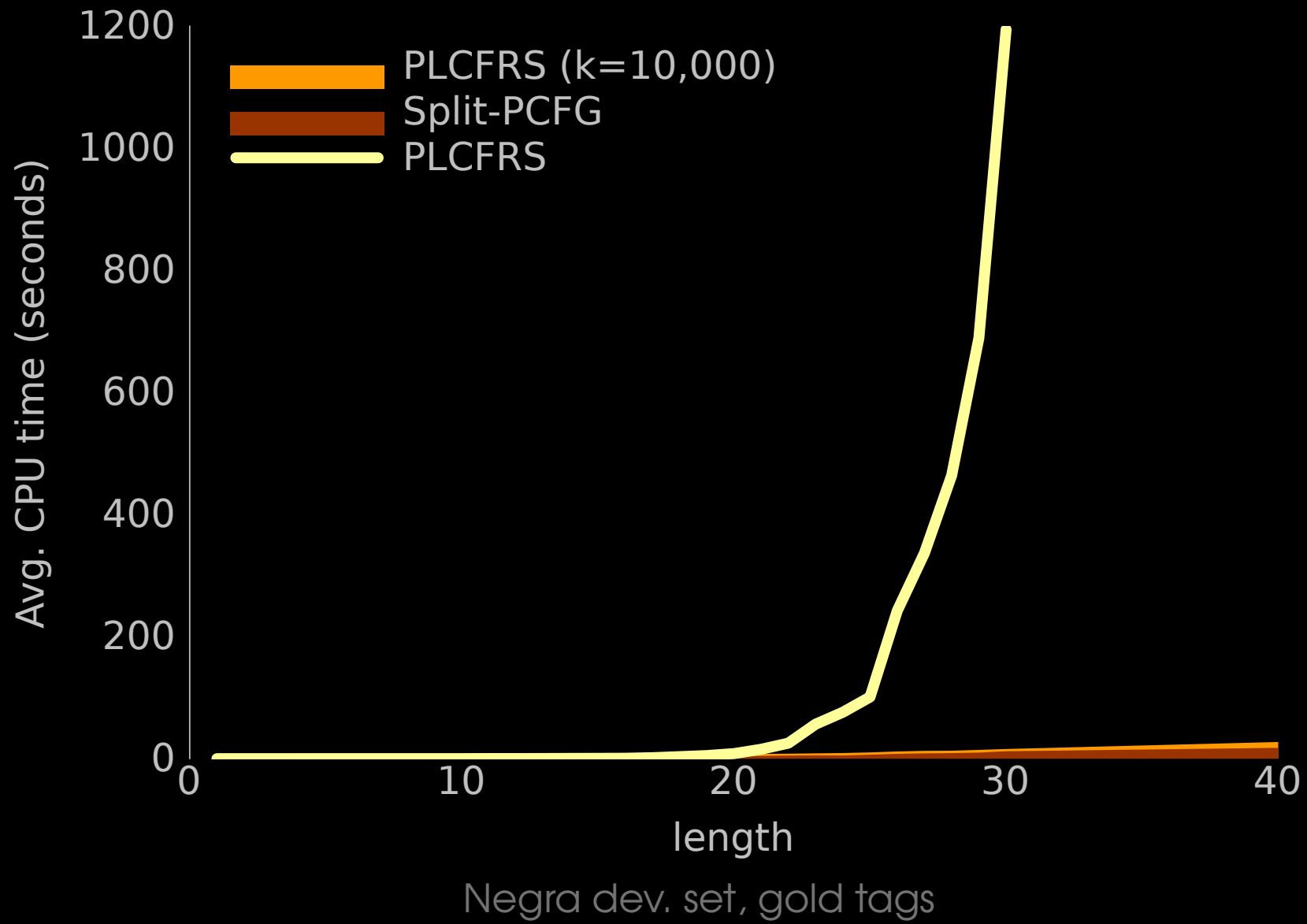
Boyd (2007). Discontinuity revisited.

Coarse-to-fine pipeline



prune parsing with G_{m+1} by only considering items in k-best G_m derivations.

With coarse-to-fine



Negra dev. set, gold tags

Data-Oriented Parsing

Treebank grammar

trees $\Rightarrow$ productions + rel. frequencies

$\Rightarrow$ problematic independence assumptions

Data-Oriented Parsing (DOP)

trees $\Rightarrow$ fragments + rel. frequencies

fragments are arbitrarily sized chunks from the corpus

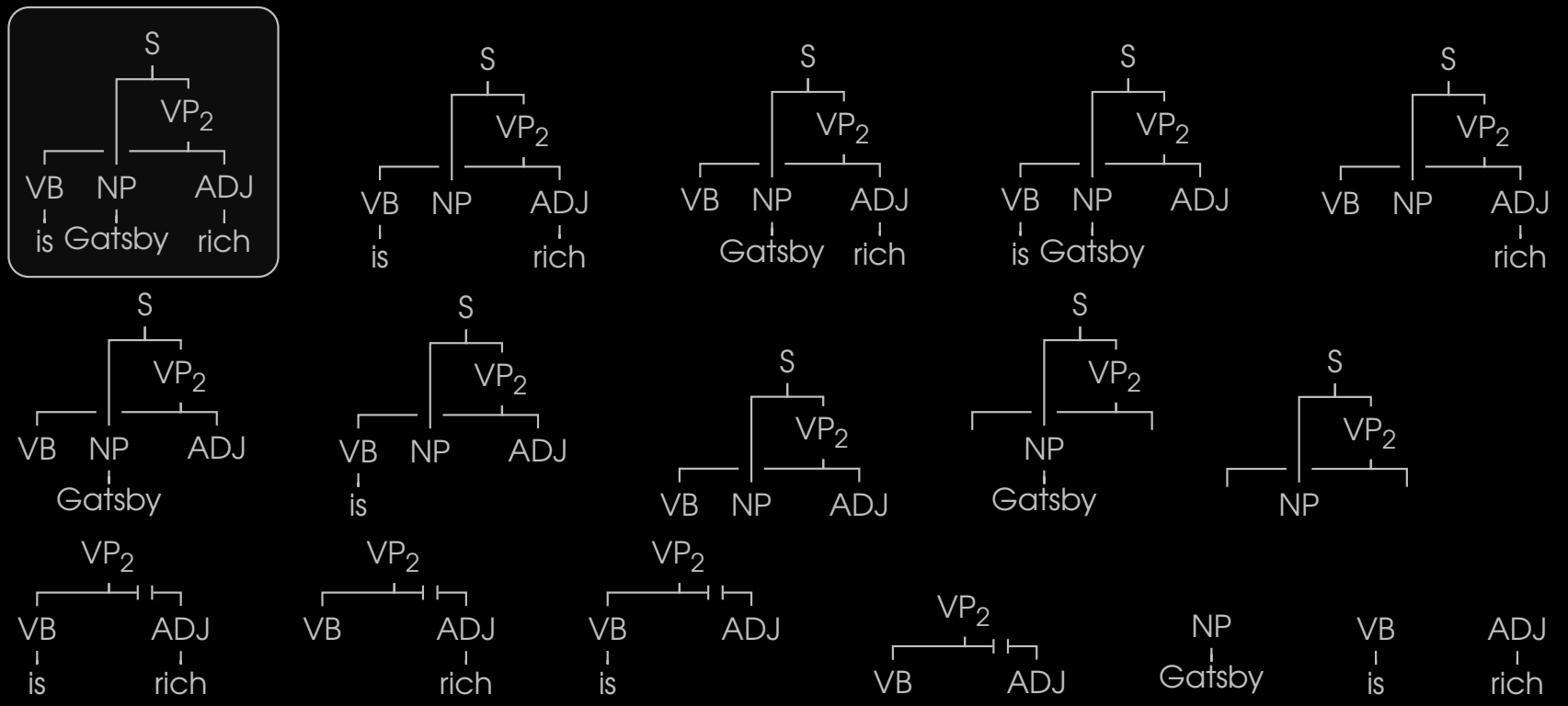
consider all possible fragments from treebank

. . . and "let the statistics decide"

Scha (1990): Lang. theory and lang. tech.; competence and performance

Bod (1992): A computational model of language performance

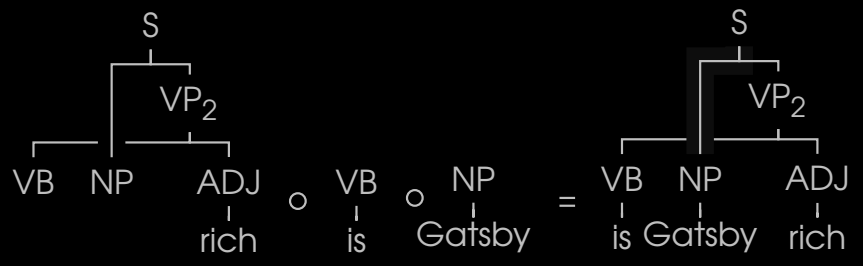
DOP fragments



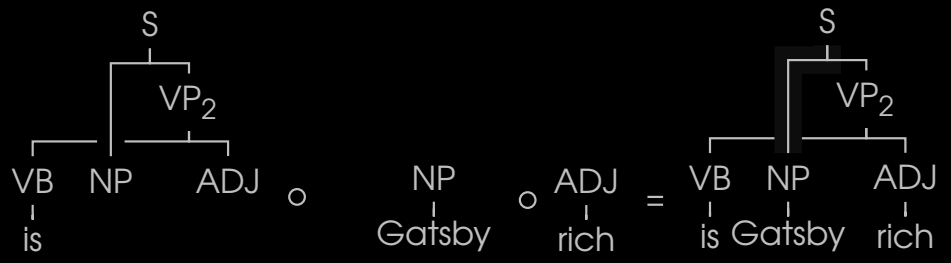
$$P(f) = \frac{\text{count}(f)}{\sum_{f' \in F} \text{count}(f')} \text{ where } F = \{ f' \mid root(f') = root(f) \}$$

Note: discontinuous frontier non-terminals mark destination of components

DOP derivation



$P(d) = 0.2$



$P(d) = 0.3$

Derivations for this tree $P(t) = 0.5$

$$P(d) = P(f_1 \circ \cdots \circ f_n) = \prod_{f \in d} p(f)$$

$$P(t) = P(d_1) + \cdots + P(d_n) = \sum_{d \in D(t)} \prod_{f \in d} p(f)$$

DOP implementation issues

Exponential number of fragments
due to all-fragments assumption

- Can use DOP reduction (Goodman 2003);
  weight of fragments spread over many productions
- Can restrict number of fragments
  by depth or frontier nodes &c.,
  $\Rightarrow$ but: not data-oriented!

Goodman (2003): Efficient parsing of DOP with PCFG-reductions

Double-DOP

- Extract fragments that occur at least twice in treebank
- For every pair of trees, extract maximal overlapping fragments
- Can be extracted in linear average time
- Number of fragments is small enough to parse with directly

Sangati & Zuidema (2011). Accurate parsing w/compact TSGs: Double-DOP

From fragments to grammar

- Fragments mapped to unique rules, relative frequencies as probabilities
  - Remove internal nodes, leaves root node, substitution sites & terminals $X \rightarrow X_1 \ldots X_n$
- Reconstruct derivations after parsing

Sangati & Zuidema (2011). Accurate parsing w/compact TSGs: Double-DOP

Preprocessing

- Remove function labels
- Binarize w/markovization (h=1, v=1)
- Simple unknown word model
  - Rare words replaced by features (model 4 from Stanford parser)
  - Reserve probability mass for unseen (tag, word) pairs

Results w/Double-DOP

	F1 %
DOP reduction	74.3
Double-DOP	

(Negra dev set $\leq$ 40 words, gold tags)

Results w/Double-DOP

	F1 %
DOP reduction	74.3
Double-DOP	76.3

(Negra dev set $\leq$ 40 words, gold tags)

Also: parsing 3$\times$ faster, grammar 3$\times$ smaller

Results w/Double-DOP

	k=50 F1 %	k=5000 F1 %
DOP reduction	74.3	73.5
Double-DOP	76.3	

(Negra dev set $\leq$ 40 words, gold tags)

What if we reduce pruning?

Results w/Double-DOP

	k=50 F1 %	k=5000 F1 %
DOP reduction	74.3	73.5
Double-DOP	76.3	77.7

(Negra dev set $\leq$ 40 words, gold tags)

What if we reduce pruning?
$\Rightarrow$ For Double-DOP, performance does not deterioriate with expanded search space.

Main Results: test sets

Parser, treebank	$\|w\|$	POS	F1	EX
GERMAN				
vanCra2012, Negra	≤ 40	100	72.3	33.2
#KaMa2013, Negra	≤ 30	100	75.8	
this paper, Negra	≤ 40	100	**76.8**	**40.5**
this paper, Negra	≤ 40	96.3	74.8	38.7
HaNi2008, Tiger	≤ 40	97.0	75.3	32.6
this paper, Tiger	≤ 40	97.6	**78.8**	**40.8**

KaMa: Kallmeyer & Maier (2013) (different test set);
vanCra: van Cranenburgh (2012); HaNi: Hall & Nivre (2008).

Main Results: test sets

ENGLISH				
#EvKa2011, disc. WSJ	< 25	100	79.0	
this paper, disc. WSJ	≤ 40	96.6	**85.6**	31.3
SaZu2011, WSJ	≤ 40		87.9	33.7

EvKa: Evang & Kallmeyer (2011) (different test set);
SaZu: Sangati & Zuidema (2011).

Main Results: test sets

ENGLISH				
#EvKa2011, disc. WSJ	< 25	100	79.0	
this paper, disc. WSJ	≤ 40	96.6	**85.6**	31.3
SaZu2011, WSJ	≤ 40		87.9	33.7
DUTCH				
this paper, Alpino	≤ 40	85.2	65.9	23.1
this paper, Lassy	≤ 40	94.6	77.0	35.2

EvKa: Evang & Kallmeyer (2011) (different test set);
SaZu: Sangati & Zuidema (2011).

Can DOP handle discontuinity without LCFRS?

Split-PCFG

⇓

PLCFRS

⇓

PLCFRS Double-DOP

77.7 % F1

41.5 % EX

Split-PCFG

⇓

Split-Double-DOP

Can DOP handle discontuinity without LCFRS?

Split-PCFG

⇓

PLCFRS

⇓

PLCFRS Double-DOP

77.7 % F1

41.5 % EX

Split-PCFG

⇓

Split-Double-DOP

78.1 % F1

42.0 % EX

Answer: Yes!

Fragments can capture discontinuous contexts

Conclusions

- Multilingual results for discontinuous parsing, w/automatic assignment of tags

Conclusions

- Multilingual results for discontinuous parsing, w/automatic assignment of tags
- All fragments vs. selected fragments
  - Explicit representation of recurring fragments with Double-DOP leads to better sample of derivations than parsing with all fragments

Conclusions

- Multilingual results for discontinuous parsing, w/automatic assignment of tags
- All fragments vs. selected fragments
  - Explicit representation of recurring fragments with Double-DOP leads to better sample of derivations than parsing with all fragments
- Not necessary to parse beyond CFG!
  $\Rightarrow$ Increase amount of context
  through fragments / labels

Conclusions

- Multilingual results for discontinuous parsing, w/automatic assignment of tags
- All fragments vs. selected fragments
  - Explicit representation of recurring fragments with Double-DOP leads to better sample of derivations than parsing with all fragments
- Not necessary to parse beyond CFG!
  $\Rightarrow$ Increase amount of context through fragments / labels
  - LCFRS could be exploited for other things than discontinuity: adjunction, synchronous parsing, ...

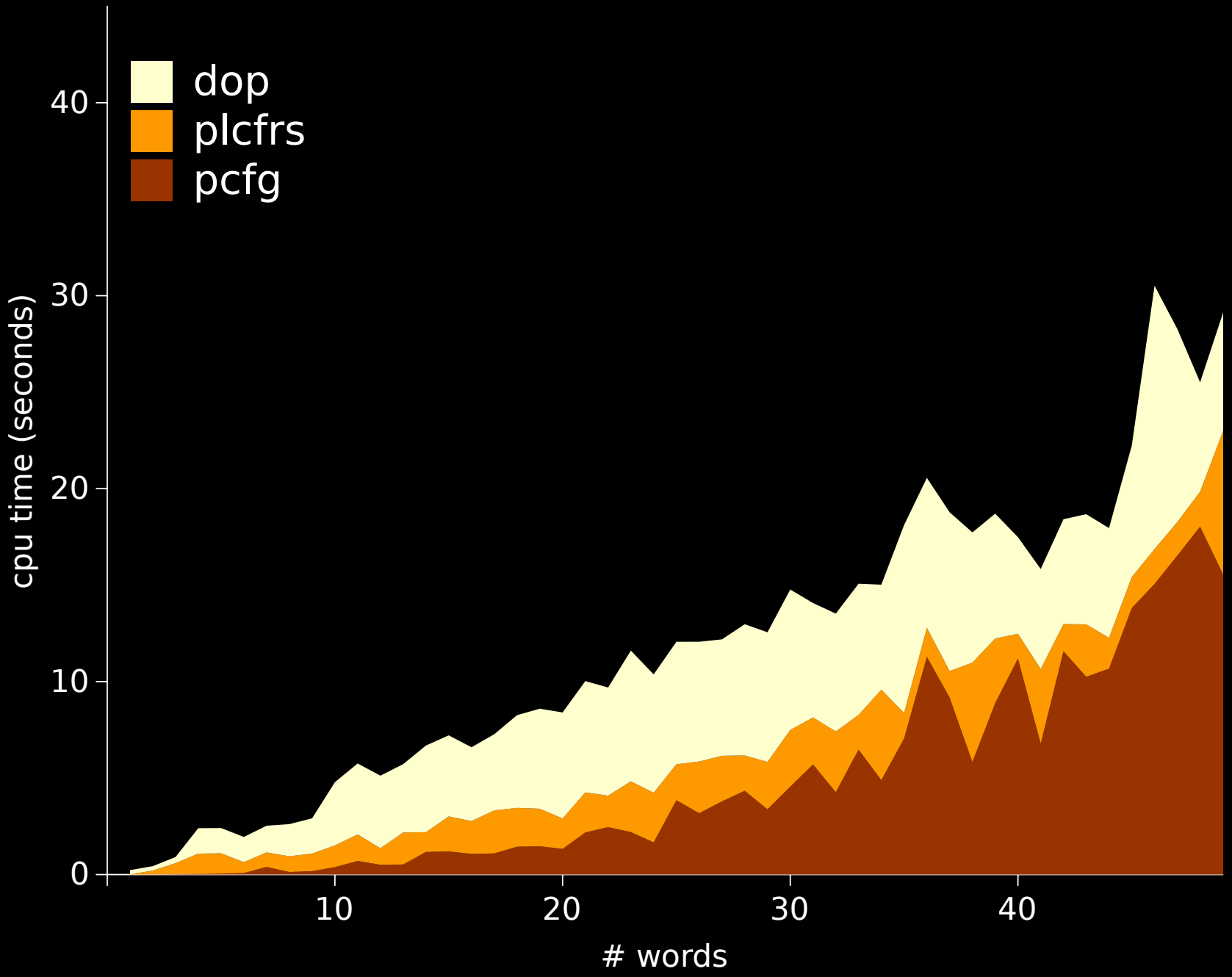
THE END

Codes: `http://github.com/andreasvc/disco-dop`

Wait . . . there's more

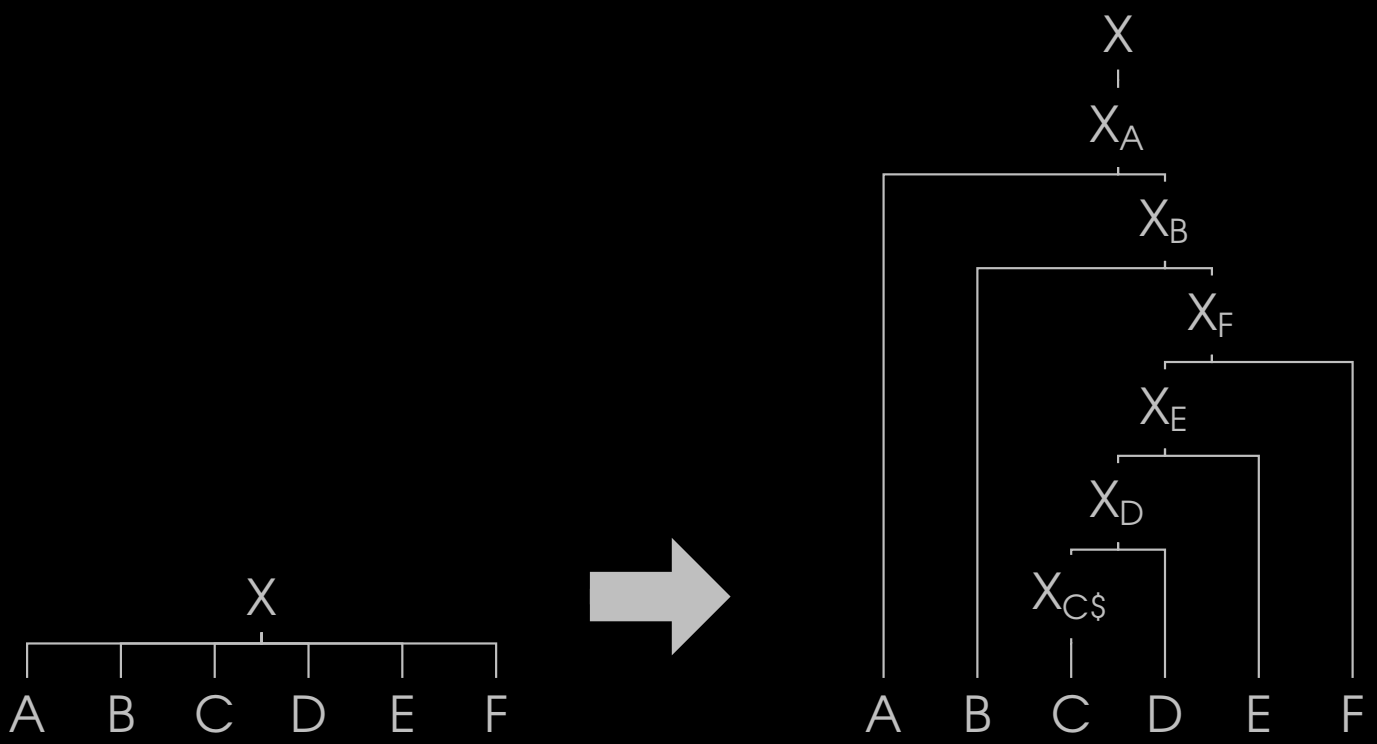
BACKUP SLIDES

Efficiency (Negra dev set)



Binarization

- mark heads of constituents
- head-outward binarization (parse head first)
- no parent annotation: $v = 1$
- horizontal Markovization: $h = 1$



Klein & Manning (2003): Accurate unlexicalized parsing.

Parser setup

```
traincorpus='wsj02-21.export',
testcorpus='wsj24.export',
corpusdir='../../dptb',
stages=[
    dict(
        name='pcfg', mode='pcfg',
        split=True, markorigin=True,
    ),
    dict(
        name='plcfrs', mode='plcfrs',
        prune=True, splitprune=True, k=10000,
    ),
    dict(
        name='dop', mode='plcfrs',
        prune=True, k=5000,
        dop=True, usedoubledop=True, m=10000,
        estimator='dop1', objective='mpp',
    ),
],
[...]
```

Web-based interface