Towards Grammar Extraction and Probabilistic Parsing for Tree Wrapping Grammar

David Arps, Tatiana Bladier, Jakub Waszczuk

TreeGraSP Workshop
26 February 2020, Düsseldorf

Heinrich Heine Universität Düsseldorf

European Research Council

Overview

Tree Wrapping Grammar

TWG Grammar Extraction

TWG Parsing with TuLiPa

TWG Probabilistic Parsing

Outline

Tree Wrapping Grammar

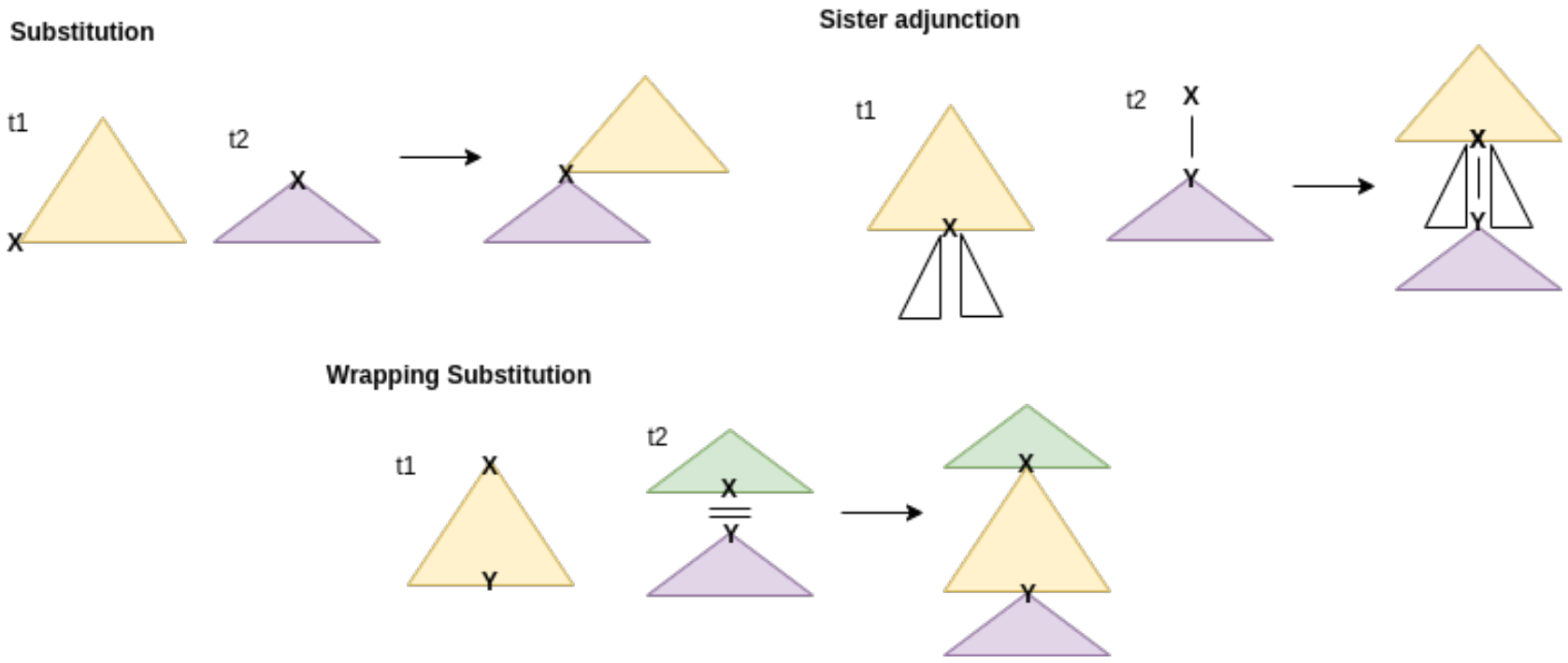
TWG Grammar Extraction

TWG Parsing with TuLiPa

TWG Probabilistic Parsing

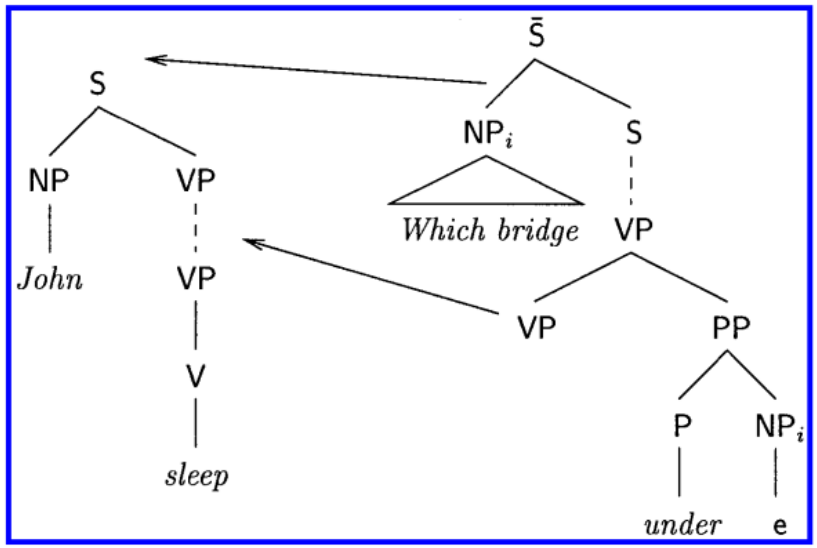
Tree-Wrapping Grammar

- ⋆ Tree-rewriting system developed by Kallmeyer et al. (2013b).
- ⋆ Originally developed with regard to RRG.
- ⋆ Three combination operations: substitution, sister-adjunction, wrapping substitution.



Some characteristics of TWG

- ⋆ TWG (thanks to wrapping) treats some linguistic cases more linguistically plausible than TAG (Rambow et al., 2001).
- ⋆ TWG has more expressive powerful than TAG (Kallmeyer et al., 2013b)
  → all counting languages, copy languages and MIX languages can be generated.
- ⋆ Some long-distance dependencies can be expressed more naturally with TWGs.



A problematic case for TAG which can be expressed more naturally using wrapping substitution, from (Rambow et al., 2001).

Outline

Tree Wrapping Grammar

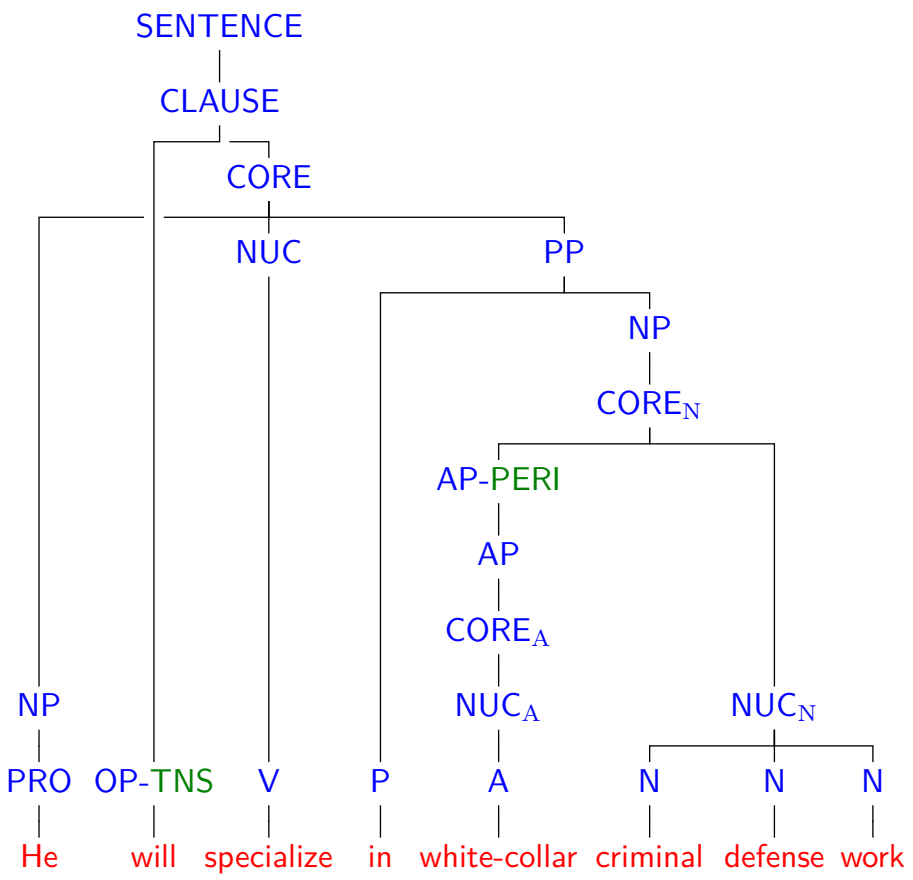
TWG Grammar Extraction

TWG Parsing with TuLiPa

TWG Probabilistic Parsing

RRGbank

- ⋆ Corpus of RRG annotated sentences (Bladier et al., 2018) → automatically converted from Penn Treebank, → manually checked and validated;
- ⋆ 2131 gold sentences, 1212 silver annotated sentences;
- ⋆ RRGbank and RRG annotation tool: `rrgbank.phil.hhu.de`.

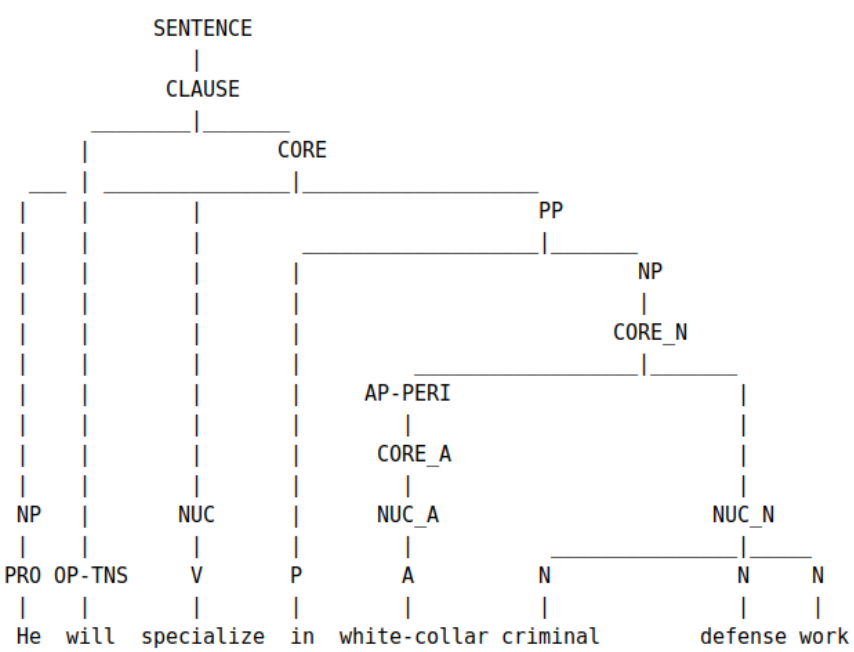


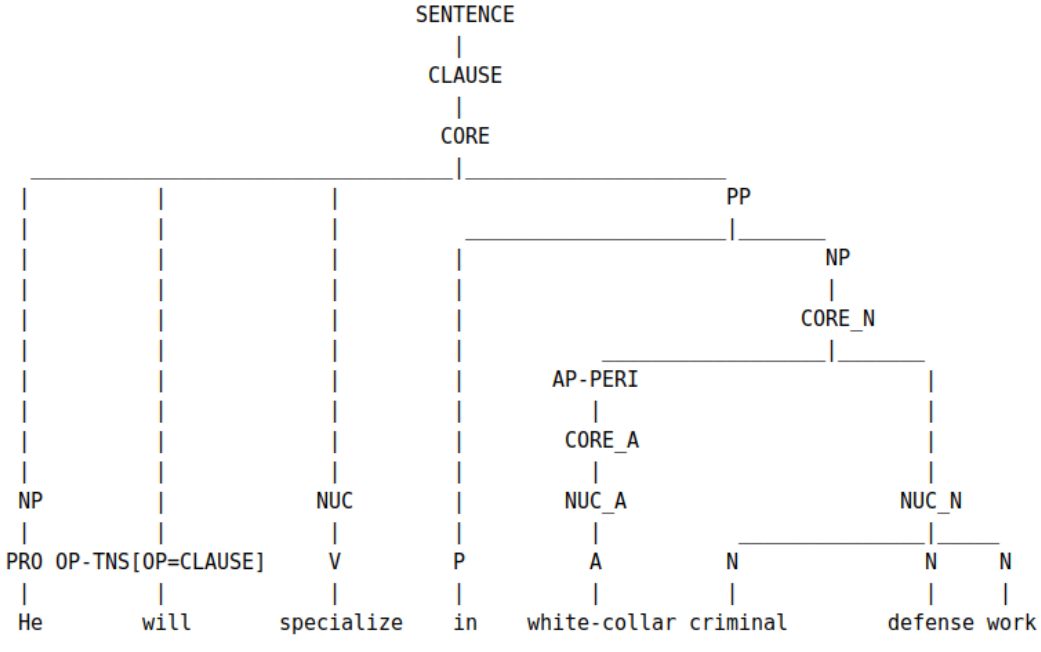
Data: Gold development set from RRGbank

- ⋆ 2000 sentences from gold development set.
- ⋆ Average sentence length: 8,51 tokens (incl. punctuation marks).
- ⋆ The longest sentence: 44 tokens.
- ⋆ # sentences $\geq$ 15 tokens: 129,
  # sentences $\geq$ 25 tokens: 9.

TWG Extraction: Removing crossing branches (1/2)

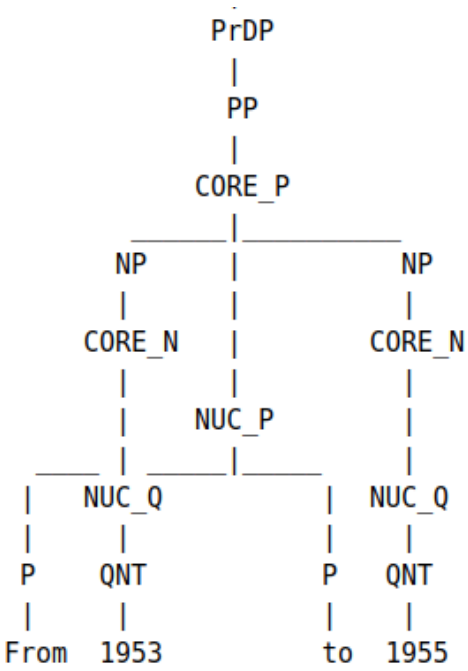
1. First, try to do minimal changes to the tree:
   → reattach the node with the OP- or -PERI to the nearest new parent a level below;
   → mark the original parent of the node;
   → check if this step alone removed crossing branches in the tree.

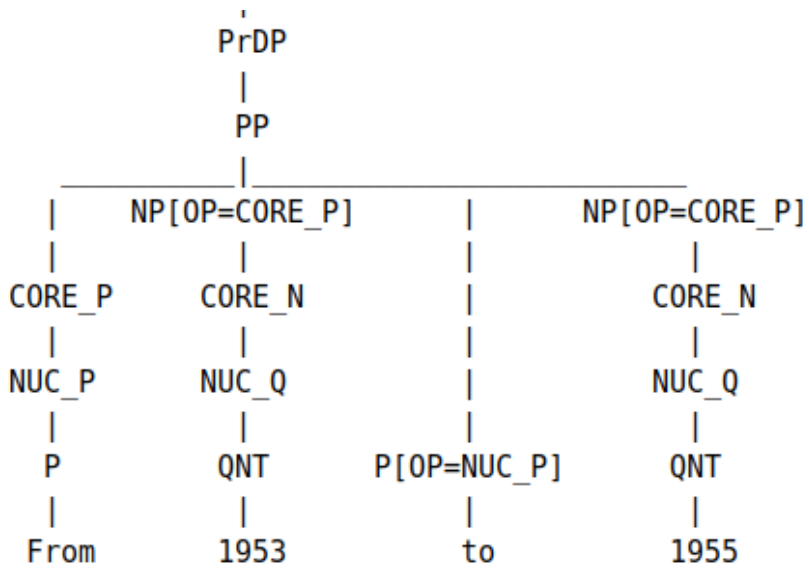




TWG Extraction: Removing crossing branches (2/2)

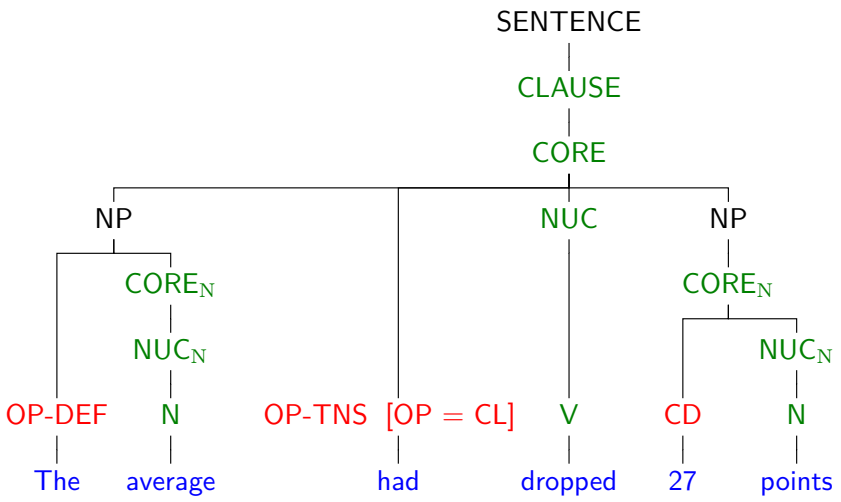
2. if the step above did not help, apply Boyd split (Boyd, 2007), as implemented in treetools by Wolfgang Maier:
   → split the children such that each sequence of children dominates a continuous block of terminals;
   → mark a single unique node as head block if it covers the original head daugther of the unsplit node;
   → reattach children blocks to the head blocks.

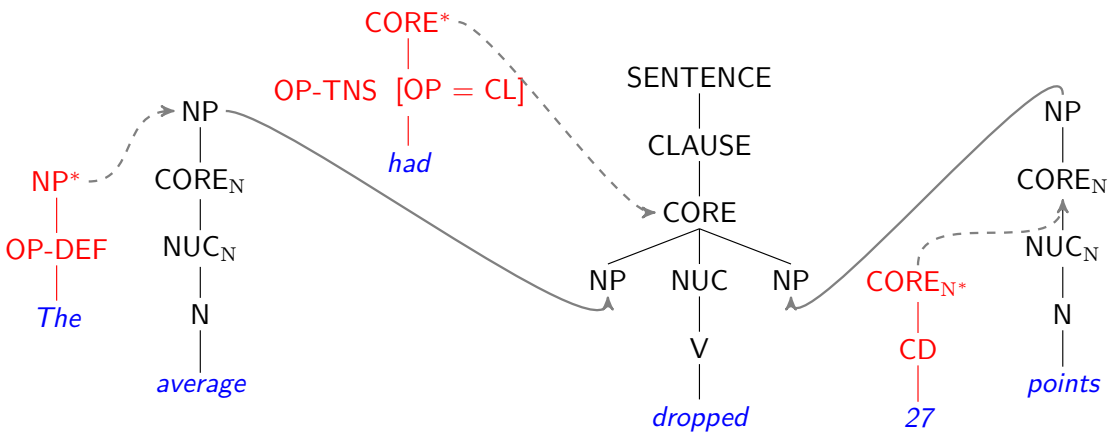




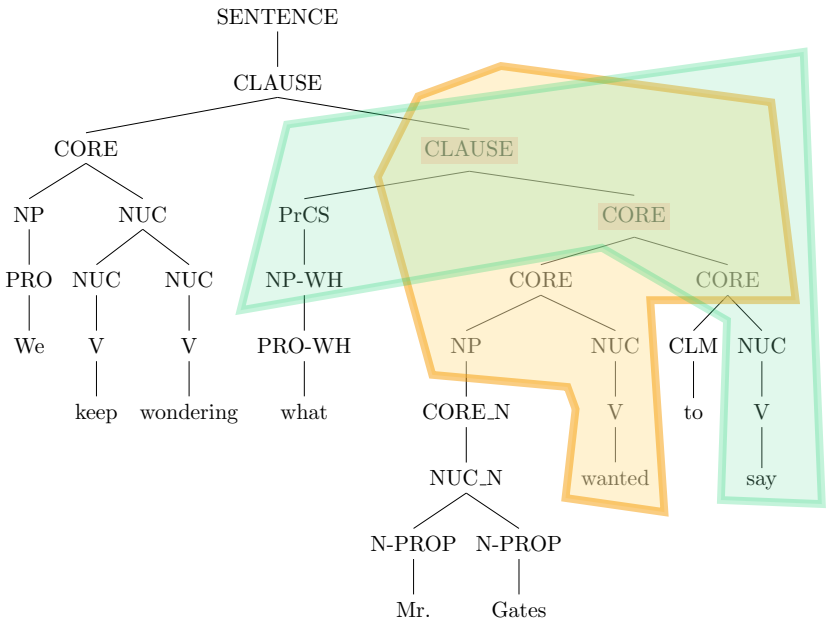
TWG: Substitution and Sister-Adjoining Trees

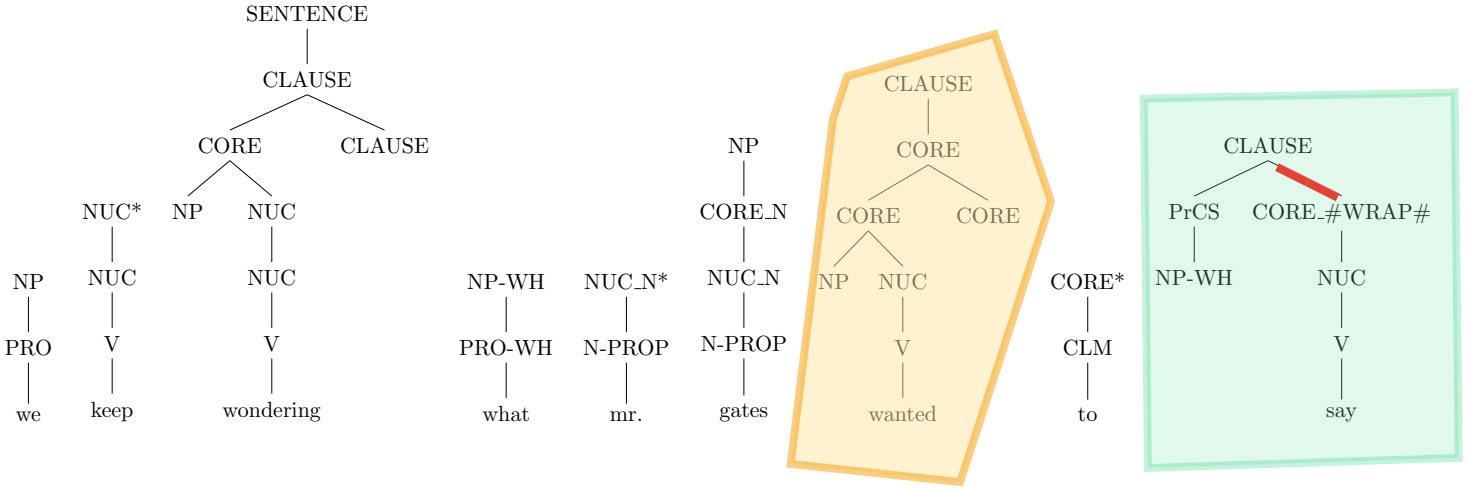
- ⋆ Adaptation of Xia's algorithm for TAG extraction (Xia, 1999):
  → Mark head nodes, modifier nodes and tree-wrapping nodes.
  → Extract elementary trees.



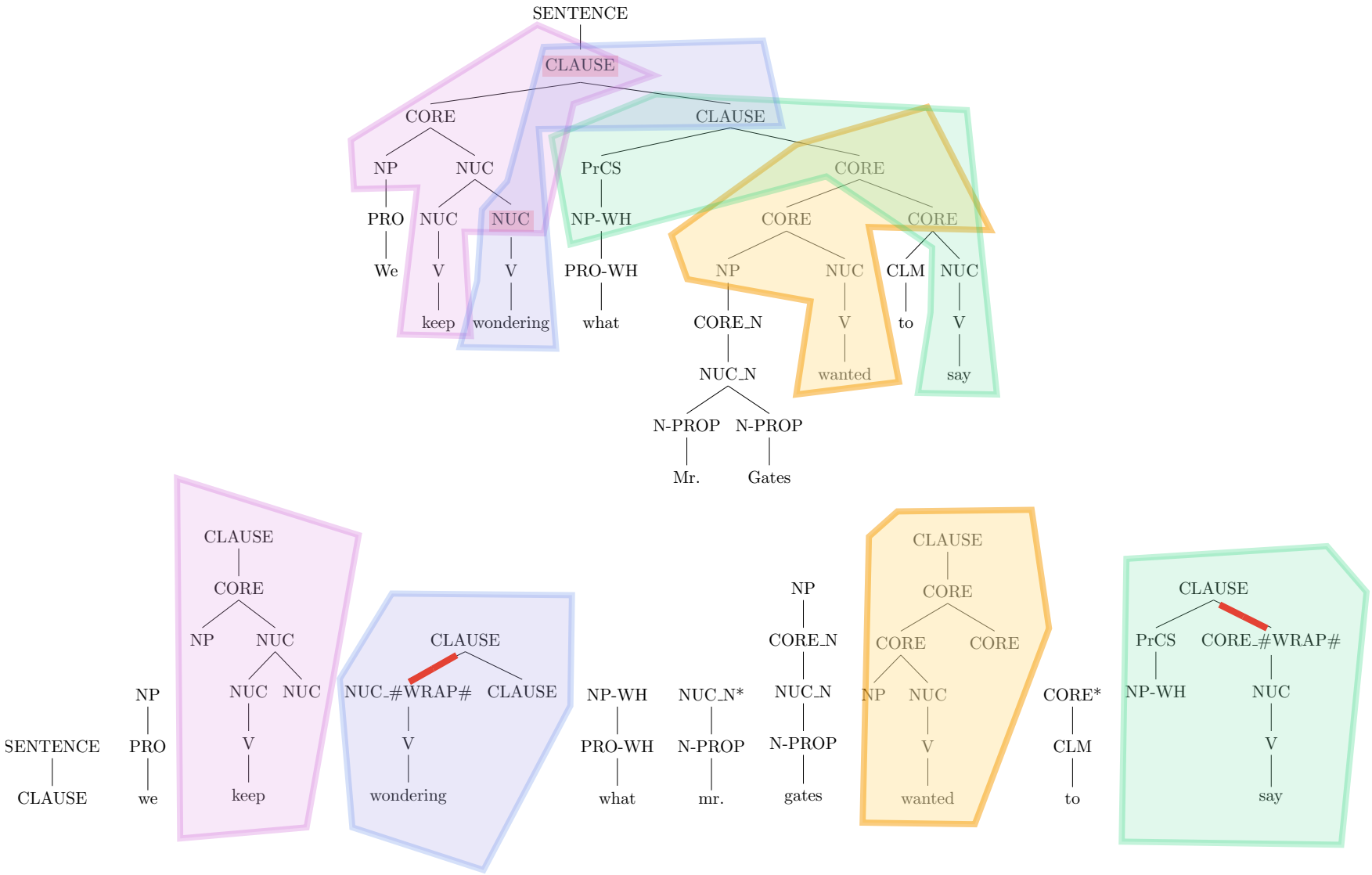


TWG: Tree-Wrapping Elementary Trees





TWG: Tree-Wrapping Elementary Trees (2nd version)



Extraction of Tree-Wrapping Elementary Trees

Two steps:

1. hard step: identify nodes which require tree-wrapping extraction.
2. easy step: extract tree-wrapping elementary trees.

Tree Wrapping: A-bar (Non-Argument) movement

Moving an argument or adjunct to an A-bar position (Radford, 2004).

WH-movement

(1) a. **Who**$_i$ did you tell Mary that she should meet _$_i$?
b. **Who**$_i$ do you think _$_i$ loves Mary?
c. **Whom**$_i$ do you think Fred loves _$_i$?
d. **What**$_i$ did Dana force/expect John to write _$_i$?
e. **Which bridge**$_i$ did John sleep under _$_i$?

Topicalization

(2) a. **This book**$_i$, I asked Bill to get his students to read _$_i$.
b. **Hot dogs**$_i$ he claims Mary seems to adore _$_i$.

Relativization

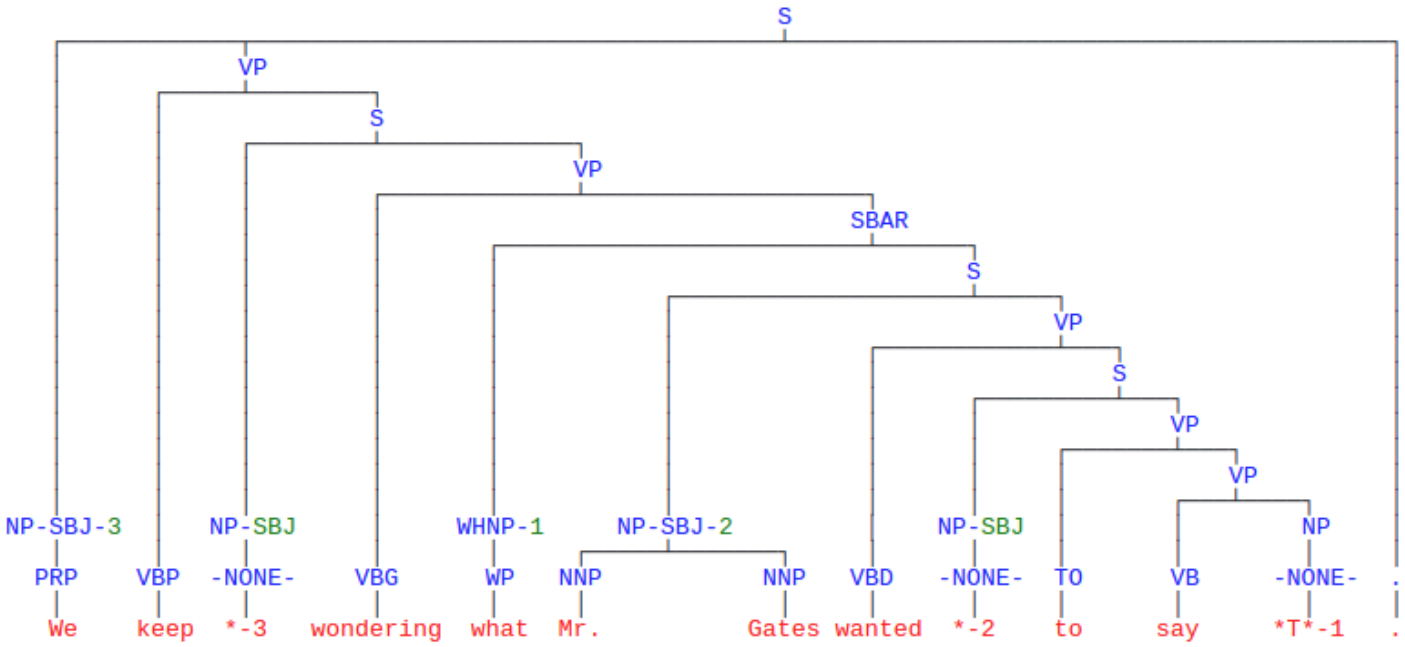
(3) a. **The book**$_i$ which I asked Bill to get his students to read _$_i$.

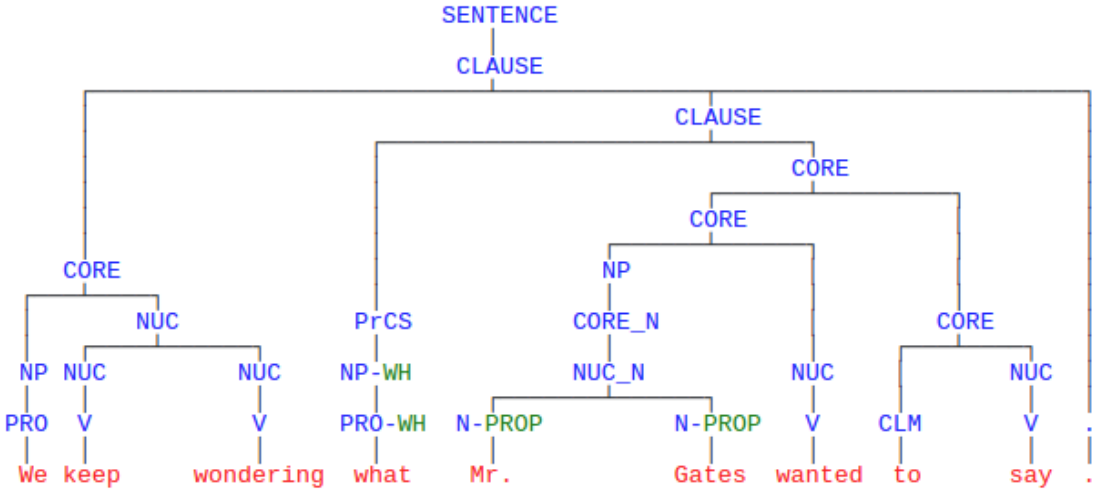
Tree Wrapping in PTB

- A-bar movement traces are indicated with *T*-markers in PTB (Bies et al., 1995)
- Not every case identified with *T*-marker is suitable for wrapping.

```
(SBARQ (WHNP-1 (WHNP Which story)
               (PP about
                   (NP tribbles)))
       (SQ did
           (NP-SBJ you)
           (VP read
               (NP *T*-1)))
       ?)
```

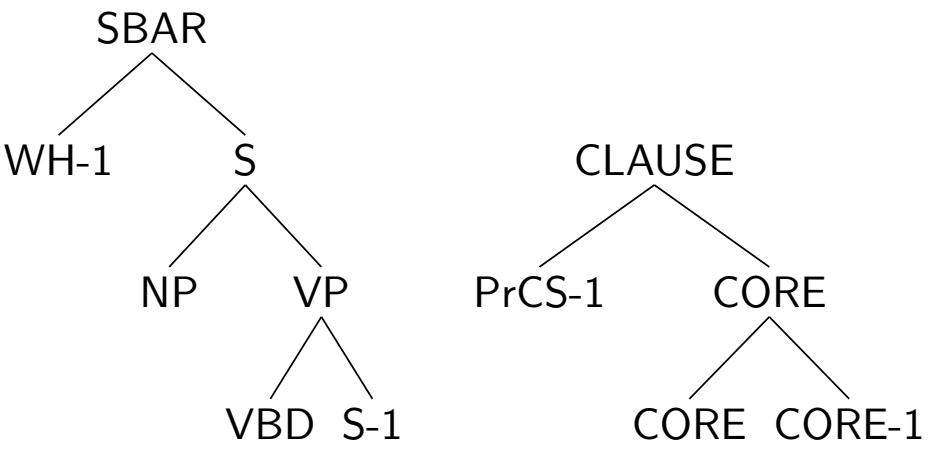
Tree-Wrapping Constructions in RRGbank





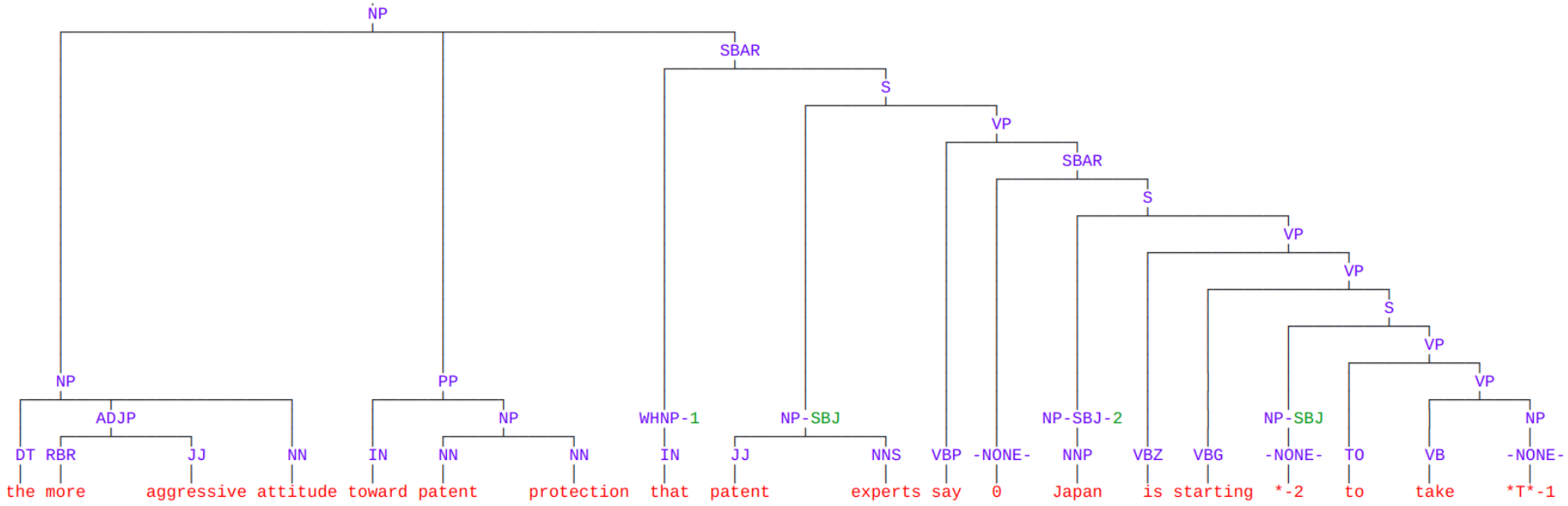
Tree-Wrapping Constructions in RRGbank

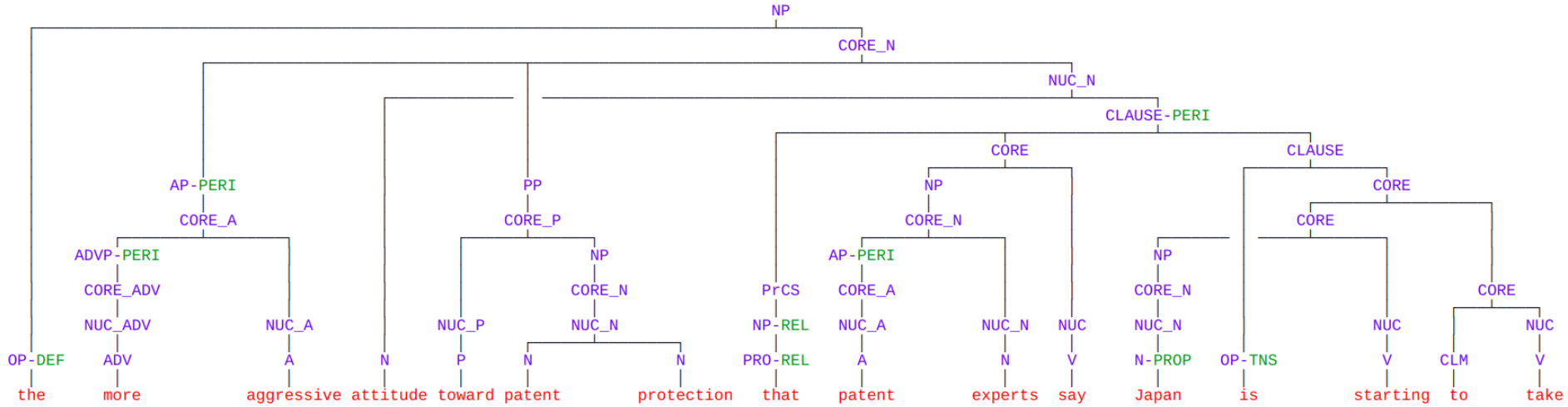
. . . *what Mr. Gates wanted to say*



Tree-Wrapping Patterns in PTB and RRGbank

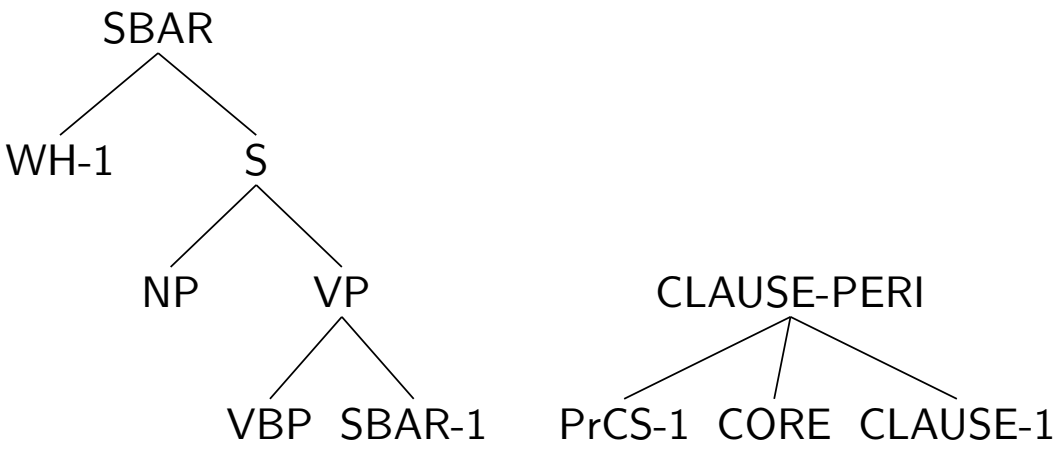
Tree-Wrapping Constructions in RRGbank





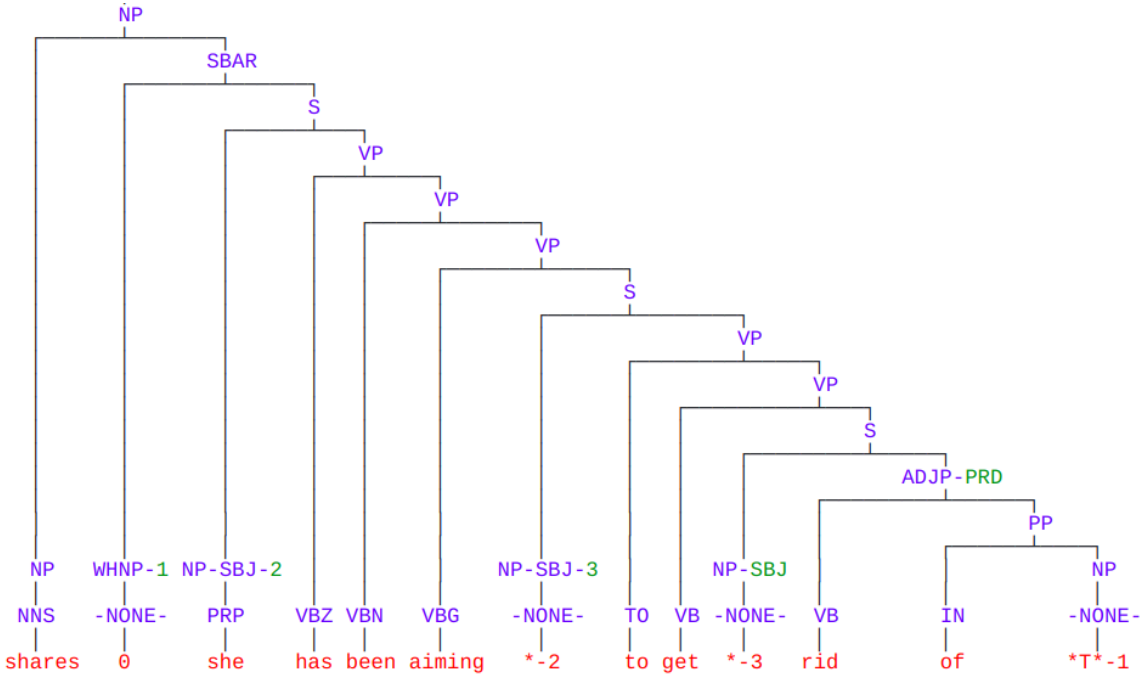
Tree-Wrapping Constructions in RRGbank

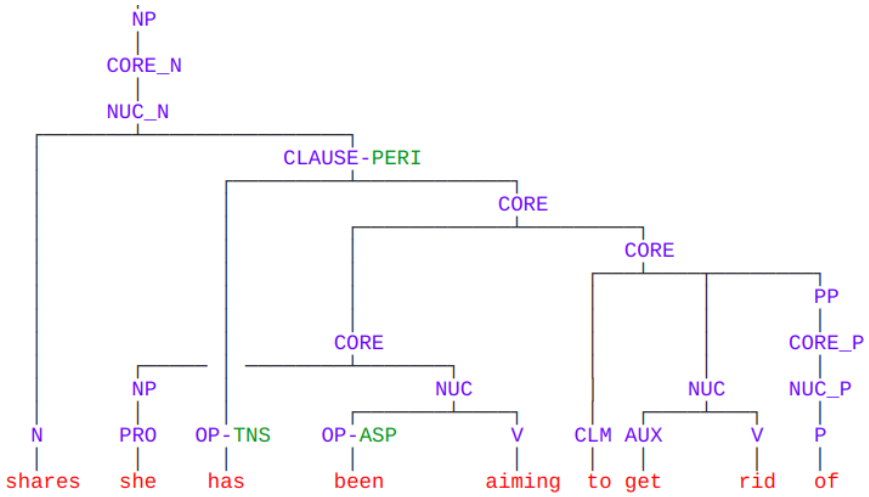
. . . that patent experts say Japan is starting to take



Tree-Wrapping Patterns in RRG

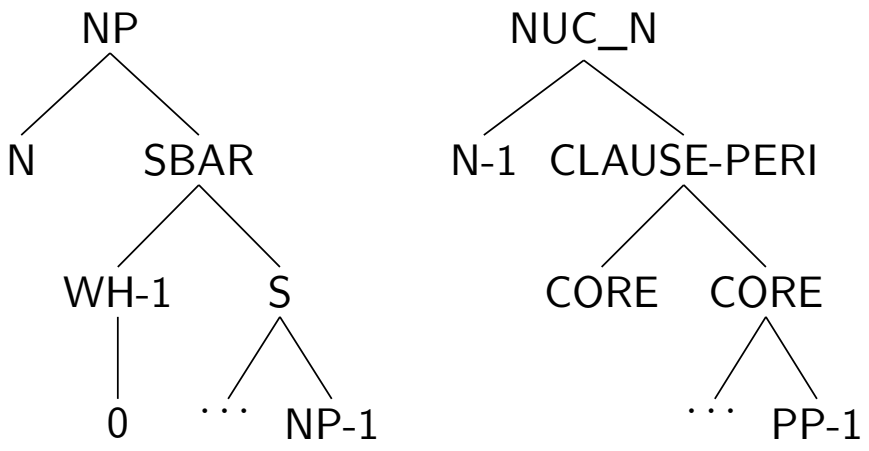
Tree-Wrapping Constructions in RRGbank





Tree-Wrapping Constructions in RRGbank

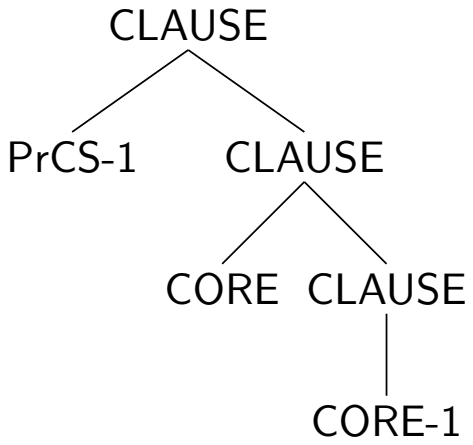
. . . shares she has been ready to get rid of



Tree-Wrapping Patterns in PTB and RRGbank

Tree-Wrapping Constructions in RRGbank

What does John claim Mary loves?



Tree-Wrapping Patterns in RRG (Kallmeyer et al., 2013a)

Statistics on extracted TWG

Parameters	**Extracted TWG**
Supertags	753
Supertags occuring once	409
POS tags	40
Sentences	2000
Avg. sentence length	8,51
# initial trees	540
# auxiliary trees	213
# d_edge trees	?

Most frequent supertags

	Supertag	**#**
1	(NP (CORE_N (NUC_N (N ⋄))))	2051
2	(NP* (OP-DEF ⋄))	1331
3	(CORE_N* (AP-PERI (CORE_A (NUC_A (A ⋄)))))	798
4	(NUC_N* (N-PROP ⋄))	768
5	(PP (CORE_P (NUC_P (P ⋄)) (NP)))	603
6	(NP (CORE_N (NUC_N (N-PROP ⋄))))	563
7	(NP (PRO ⋄))	495
8	(SENTENCE (CLAUSE (CORE (NP) (NUC (V ⋄)) (NP))))	441
9	(NUC_N* (N ⋄))	431
10	(CORE* (ADVP-PERI (CORE_ADV (NUC_ADV (ADV ⋄)))))	263
11	(CORE* (OP-TNS[OP=CLAUSE] ⋄))	263
12	(QP (CORE_Q (NUC_Q (QNT ⋄))))	233
13	(NUC* (AP (CORE_A (NUC_A (A ⋄)))))	221
14	(SENTENCE (CLAUSE (CORE (NP) (NUC (V ⋄)))))	211
15	(NP (CORE_N (NUC_N (N ⋄)) (PP)))	196

Outline

Tree Wrapping Grammar

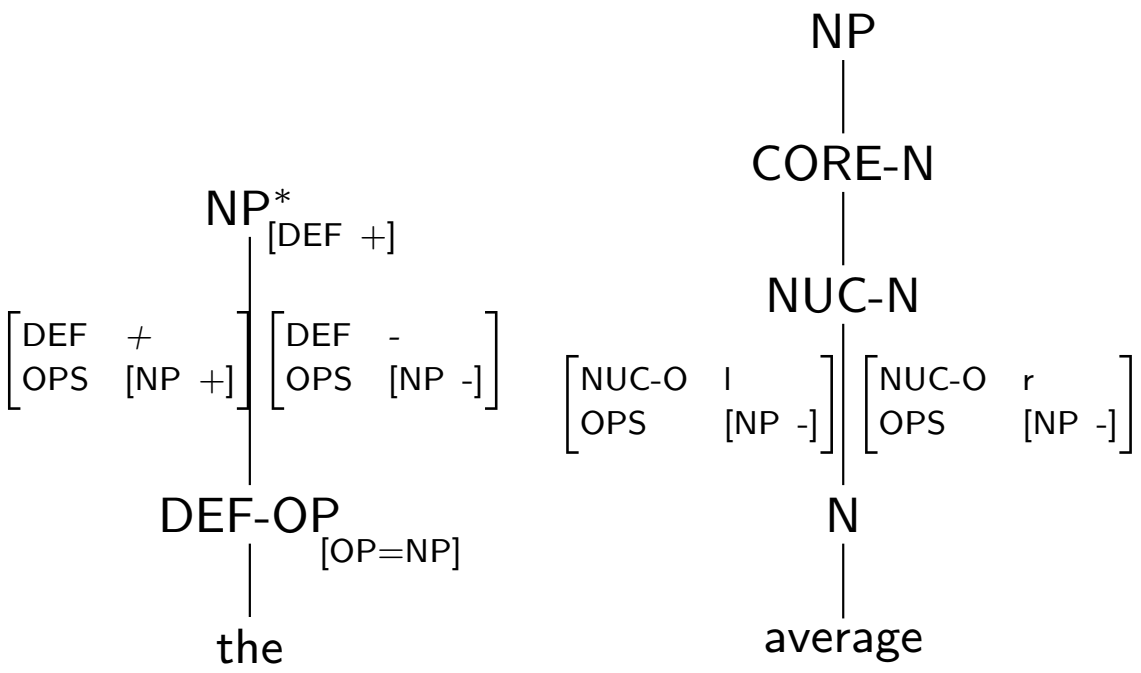
TWG Grammar Extraction

TWG Parsing with TuLiPa

TWG Probabilistic Parsing

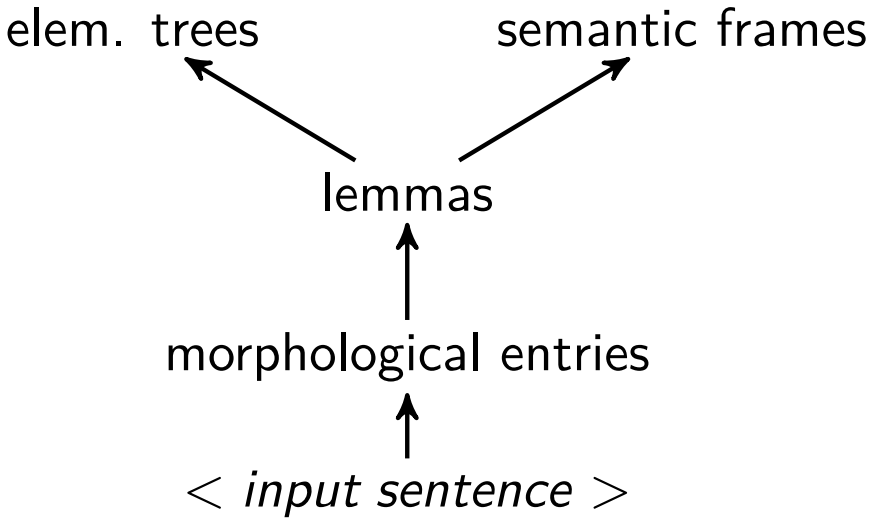
TuLiPA parsing

- application: precision grammars
- most resources created with XMG
- elementary trees can have feature structures on nodes and edges
- separated lexicon



TuLiPA parsing

- application: precision grammars
- most resources created with XMG
- elementary trees can have feature structures on nodes and edges
- separated lexicon



TuLiPA parser

- Chart-based left-to-right bottom-up parser
- **Start:** scan input
- **Goal:** reach root of derived tree

2 kinds of rules:

- move within one elementary tree
- move between elementary trees

Parsing Wrapping substitution in TuLiPA

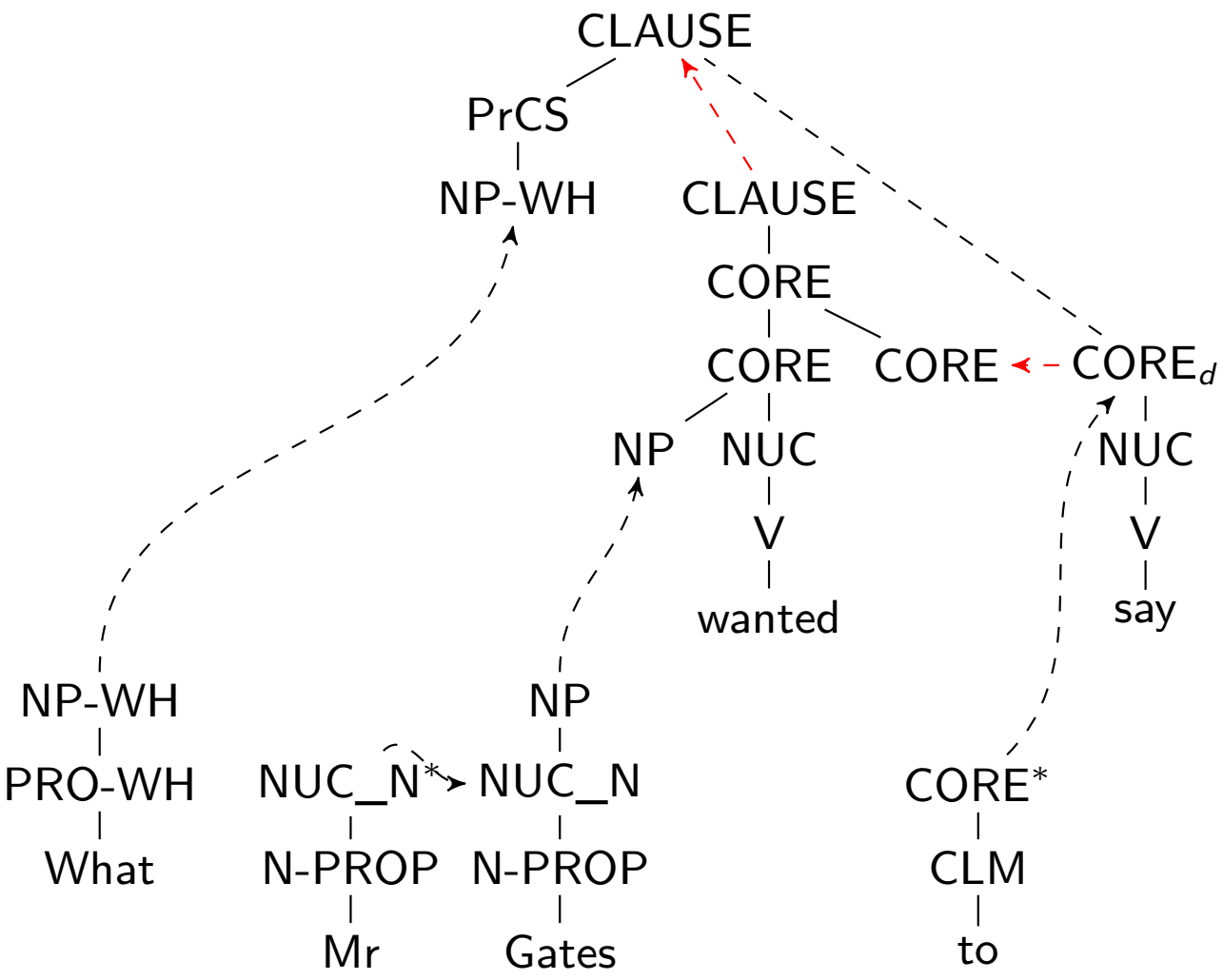
PREDICT-WRAPPING

- When in a d-daughter, what substitution nodes can we jump to?
- parse item stores a gap with span and label of the d-daughter

COMPLETE-WRAPPING

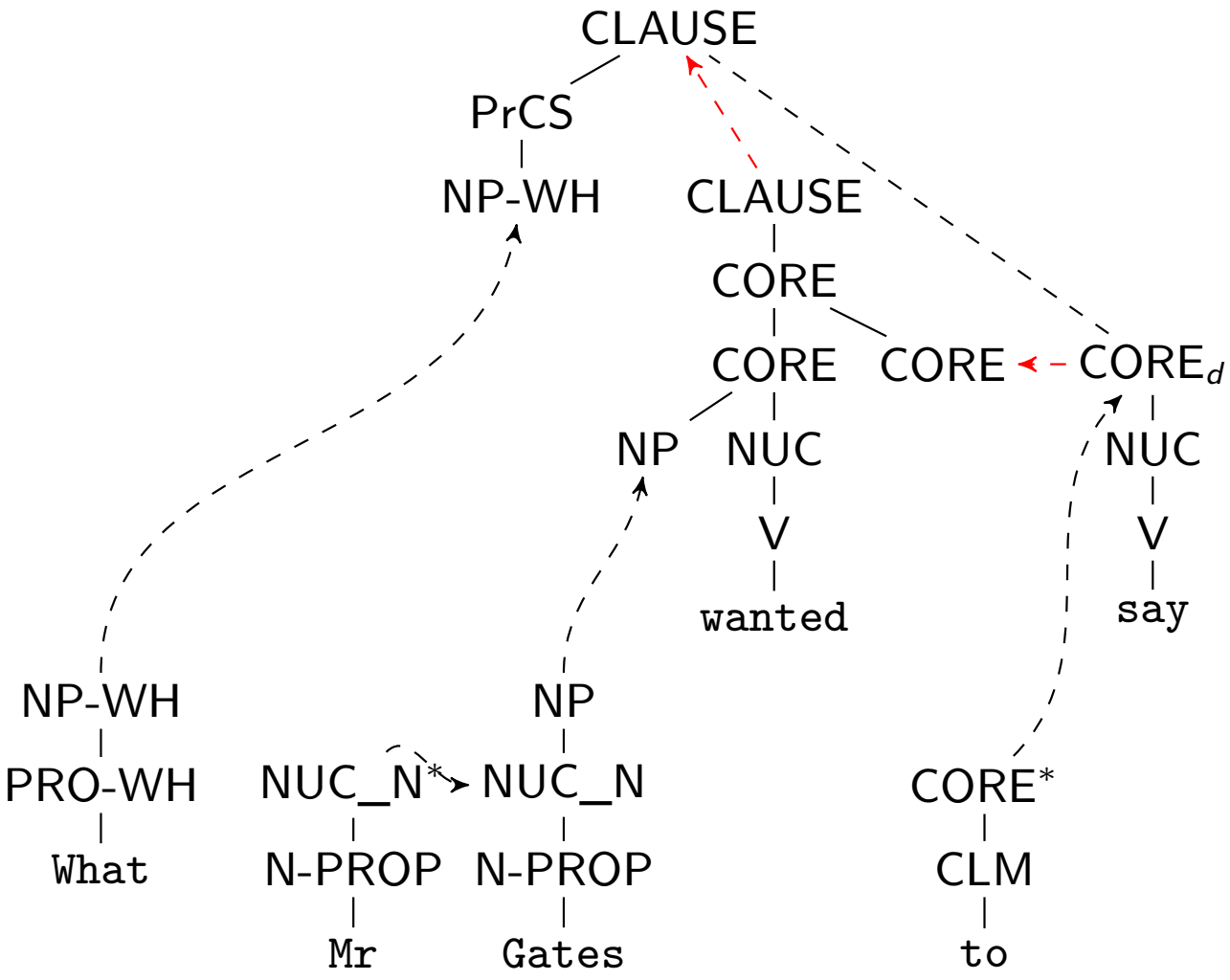
- at the root of the inner tree of the wrapping substitution
- close the gap by attaching the upper part of the wrapping tree

Parsing Example



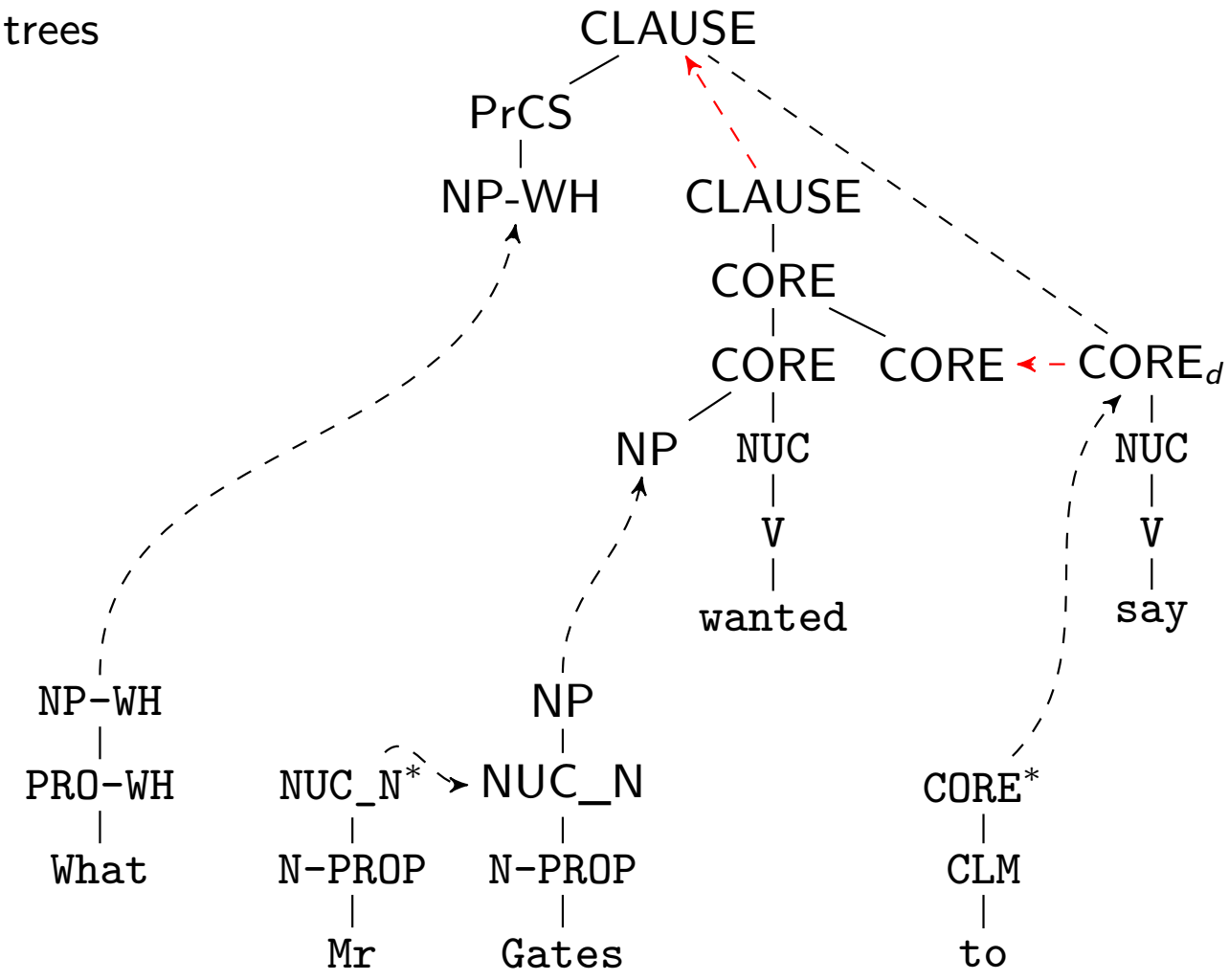
Parsing Example

* Scan



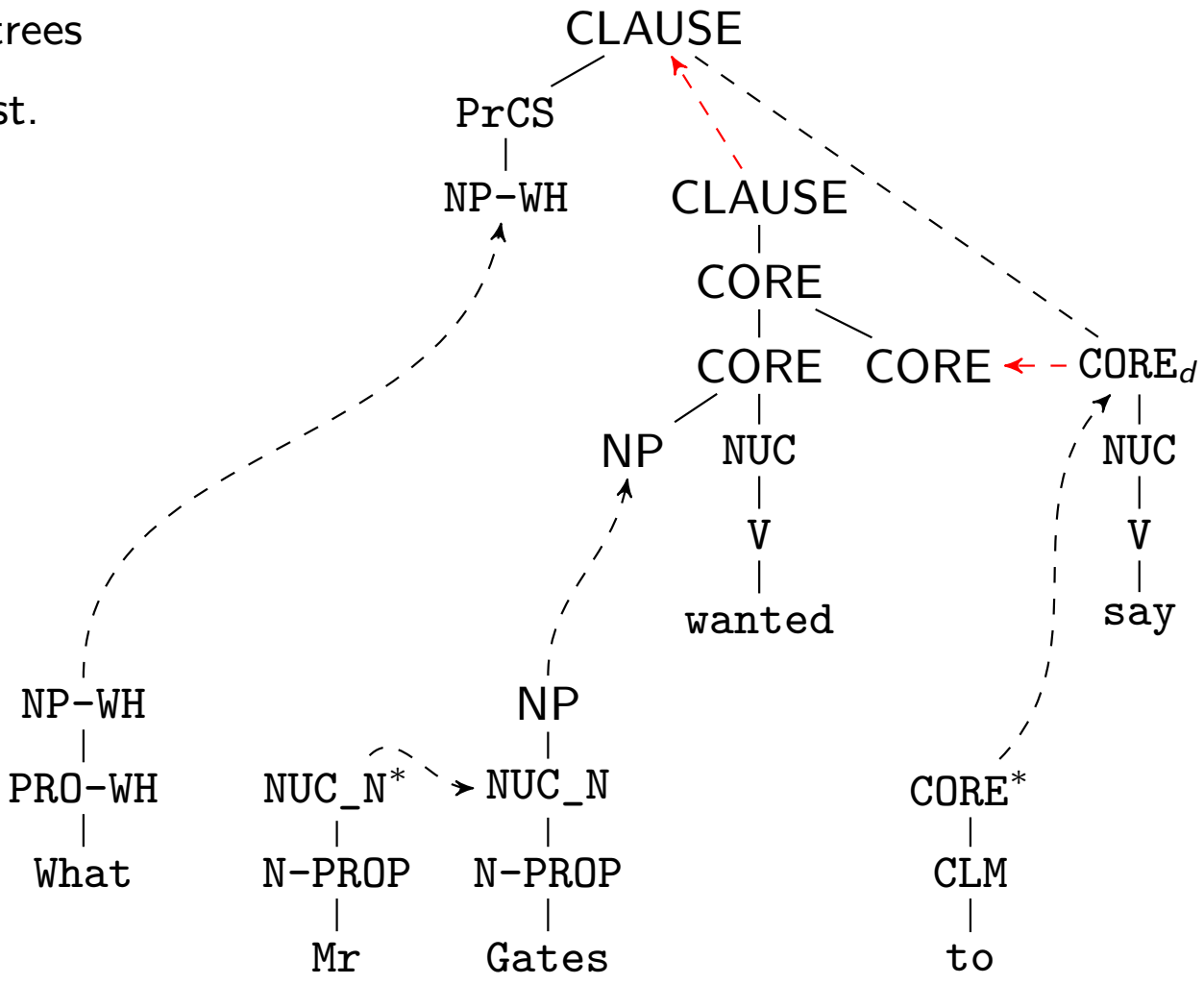
Parsing Example

* Scan
* Move within el.trees



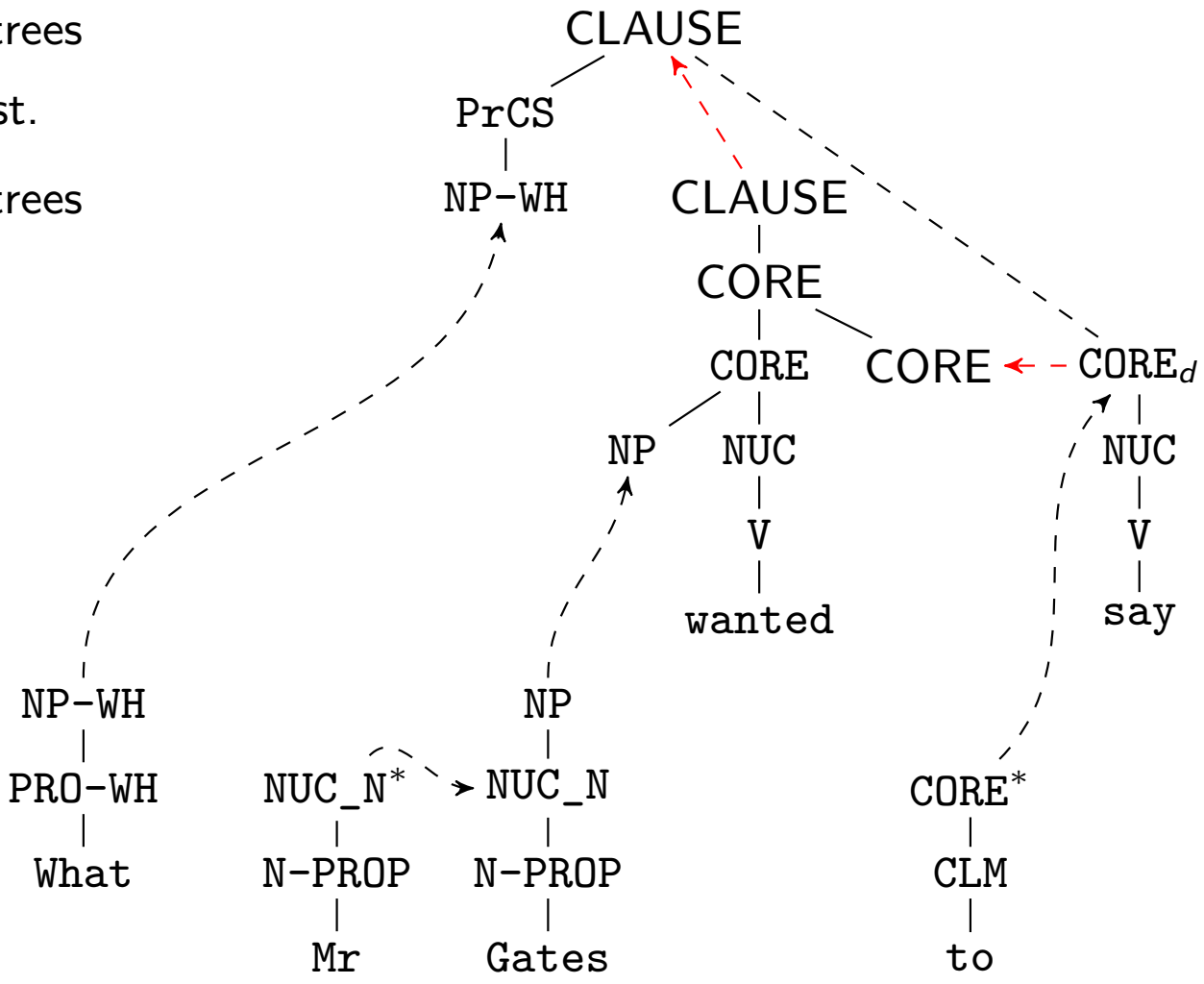
Parsing Example

* Scan
* Move within el.trees
* sis.adj. and subst.



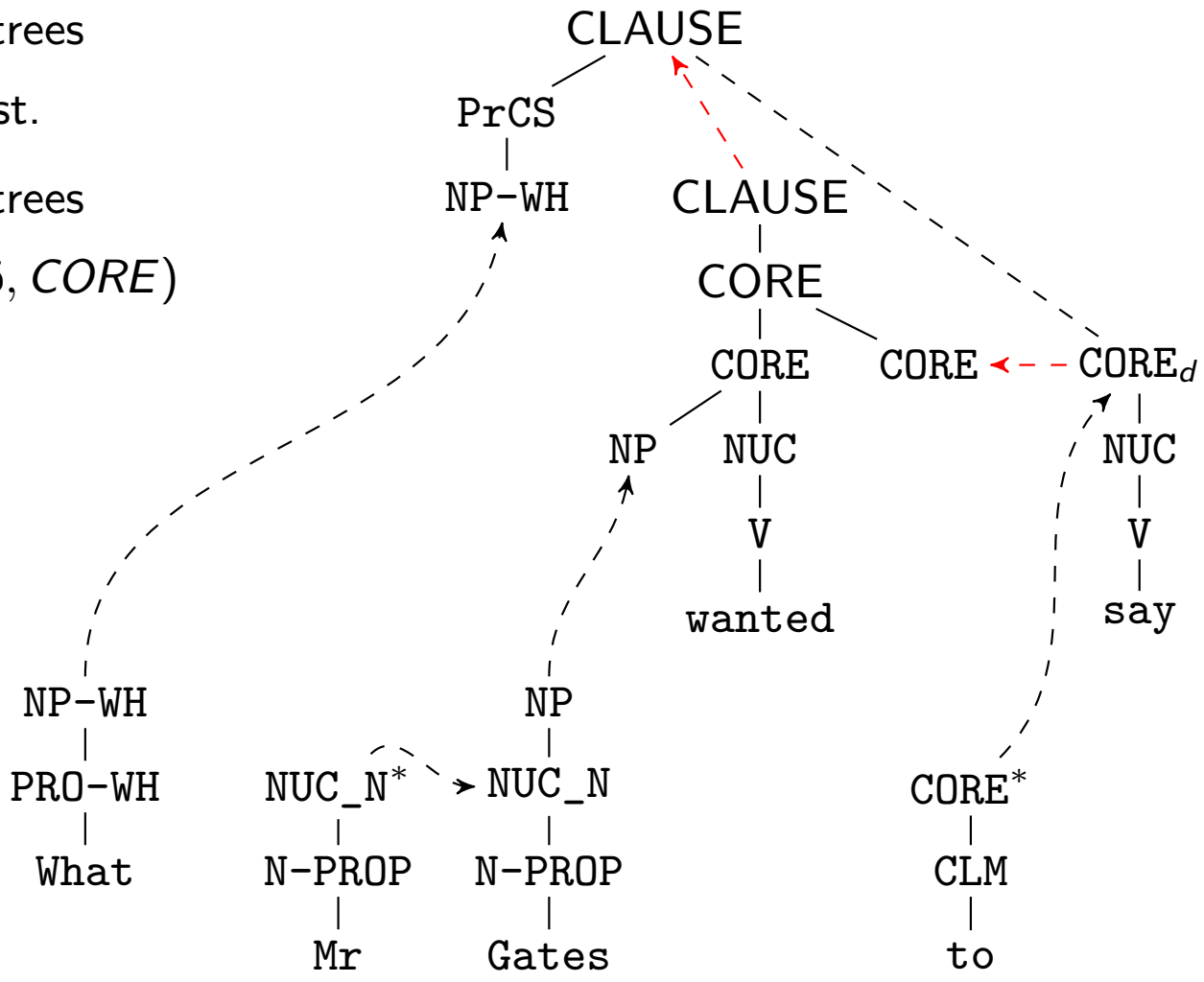
Parsing Example

* Scan
* Move within el.trees
* sis.adj. and subst.
* Move within el.trees



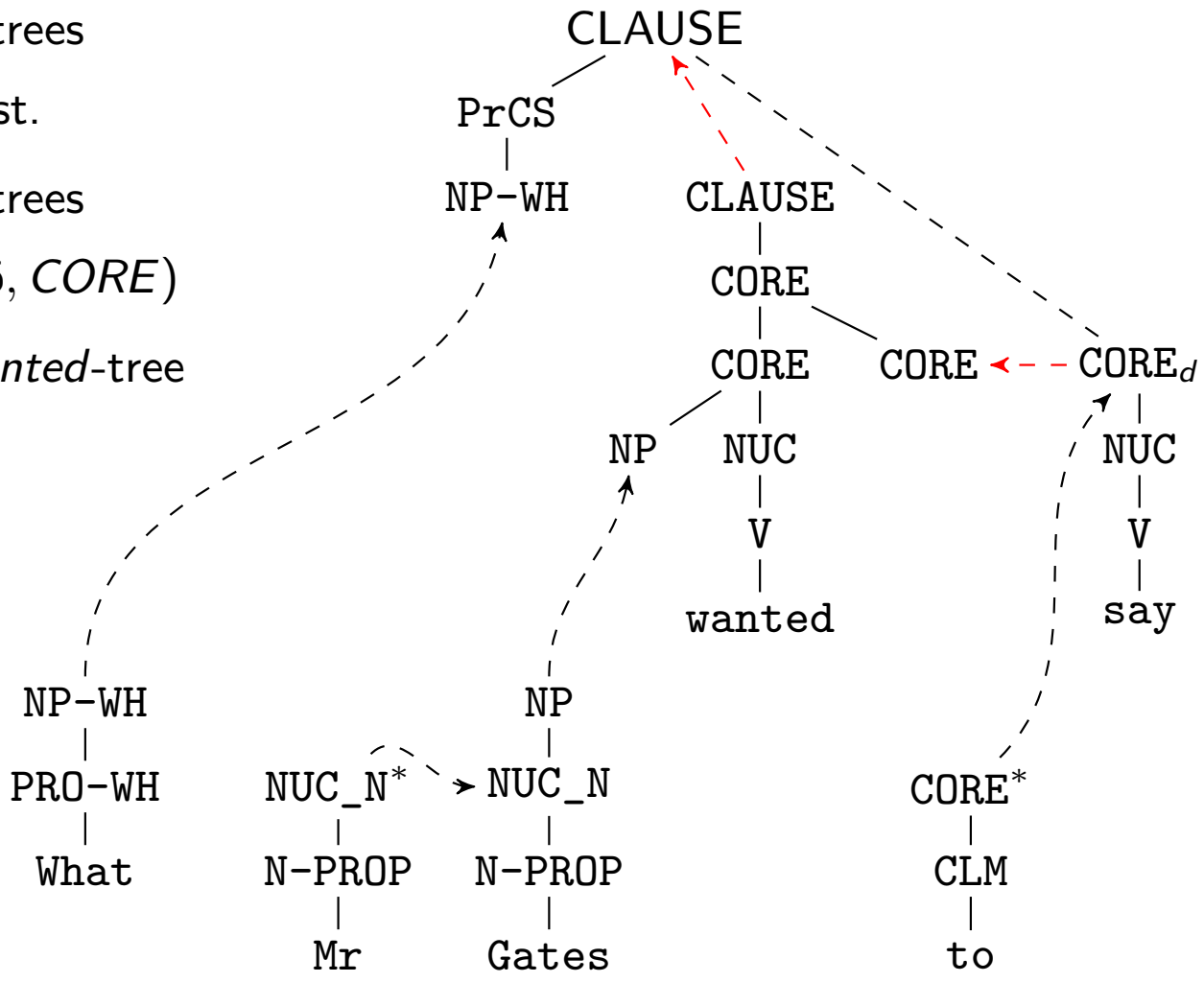
Parsing Example

* Scan
* Move within el.trees
* sis.adj. and subst.
* Move within el.trees
* Open Gap: $(4, 6, \mathit{CORE})$



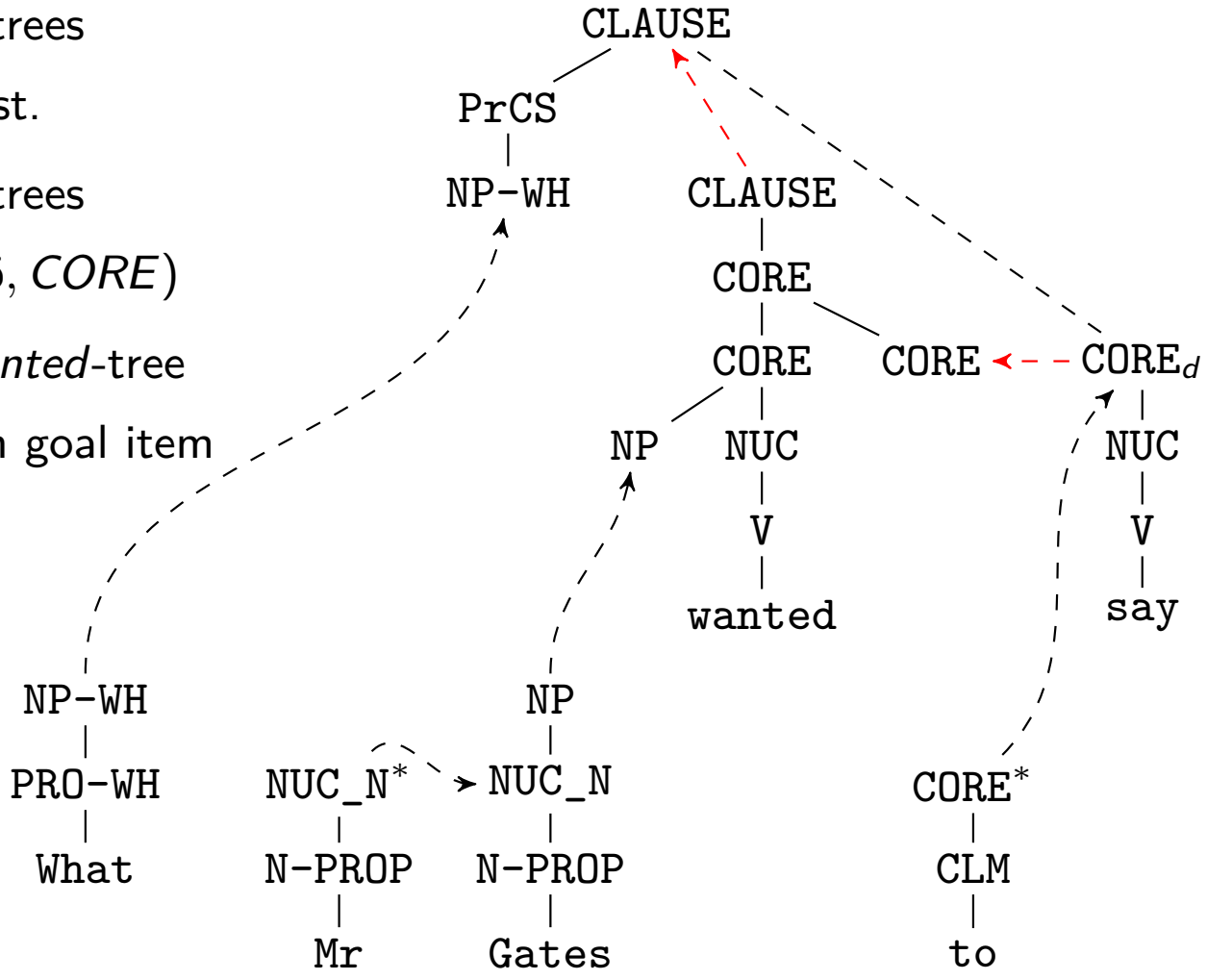
Parsing Example

* Scan
* Move within el.trees
* sis.adj. and subst.
* Move within el.trees
* Open Gap: $(4, 6, \mathit{CORE})$
* Move within *wanted*-tree



Parsing Example

* Scan
* Move within el.trees
* sis.adj. and subst.
* Move within el.trees
* Open Gap: $(4, 6, \mathit{CORE})$
* Move within *wanted*-tree
* Close gap, reach goal item



Complexity of parsing wrapping substitution

2 potential wrt. computational complexity:

1. During **PREDICT-WRAPPING:** too many possible substitution nodes
   - **Solution:**
2. During **COMPLETE-WRAPPING:** expressive power of TWG increases with the number of gaps in one parse item (gap degree)
   - **Solution:**

Complexity of parsing wrapping substitution

2 potential wrt. computational complexity:

1. During **PREDICT-WRAPPING:** too many possible substitution nodes
   - **Solution:** only include trees that can lead to a successful derivation (unlexicalized trees, trees that are already part of the derivation)
2. During **COMPLETE-WRAPPING:** expressive power of TWG increases with the number of gaps in one parse item (gap degree)
   - **Solution:** we are happy if treebank examples have one extracted argument. Not a problem for parsing extracted grammars

Outline

Tree Wrapping Grammar

TWG Grammar Extraction

TWG Parsing with TuLiPa

TWG Probabilistic Parsing

ParTAGe

- A* TAG parser $\rightarrow$ now being adapted to TWG
- symbolic core: chart-based, left-to-right, bottom-up
- relies on a supertagger and a dependency parser

$$P(\text{derivation}) \propto \prod_{i=1}^{n} \left(P(\text{supertag}_i) \times P(\text{dep-head}_i)\right)$$

where $n =$ sentence length

Input

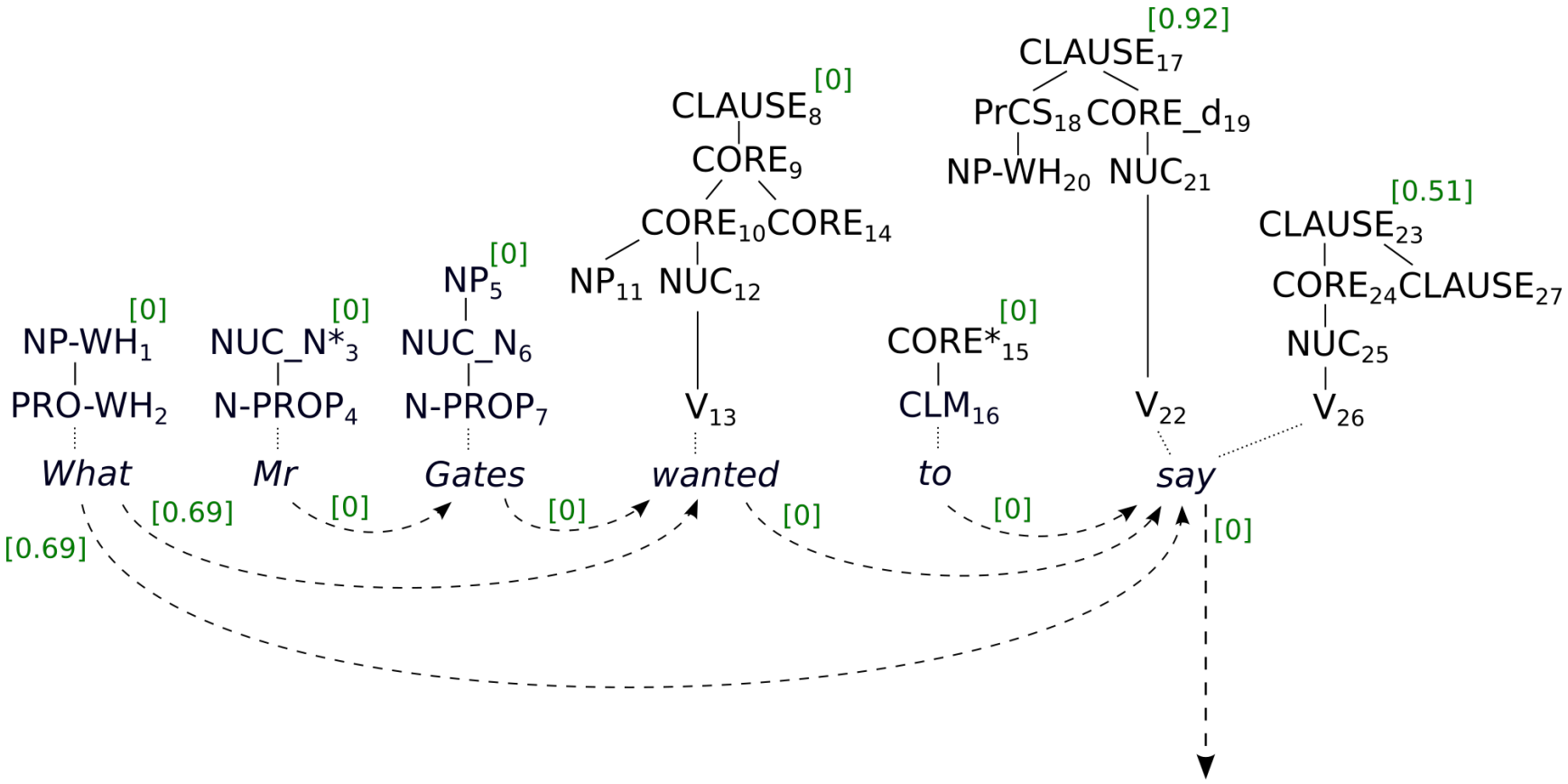


Figure: Weighted input grammar example

Output

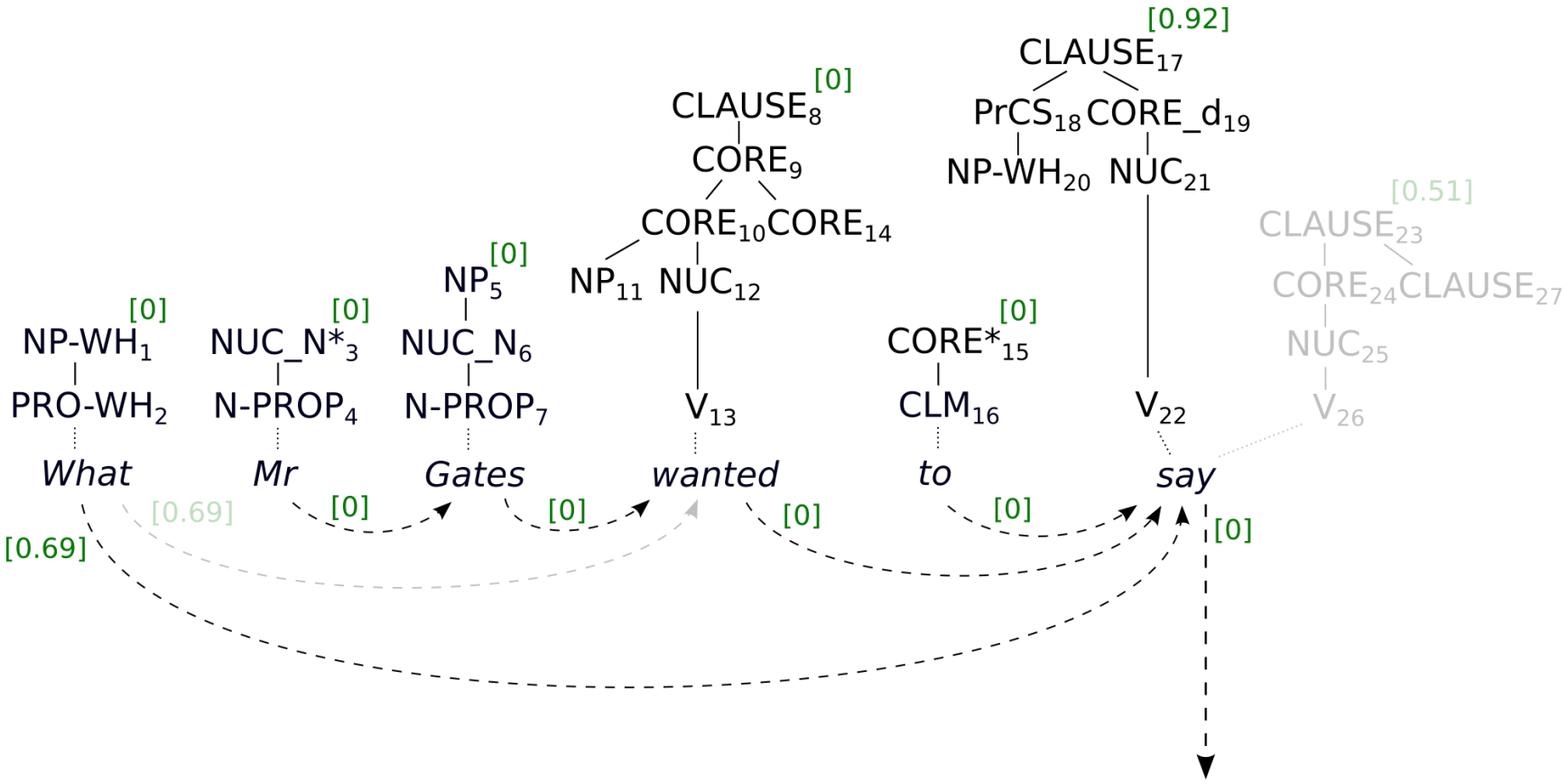


Figure: Weighted input grammar example

A* algorithm (idea)

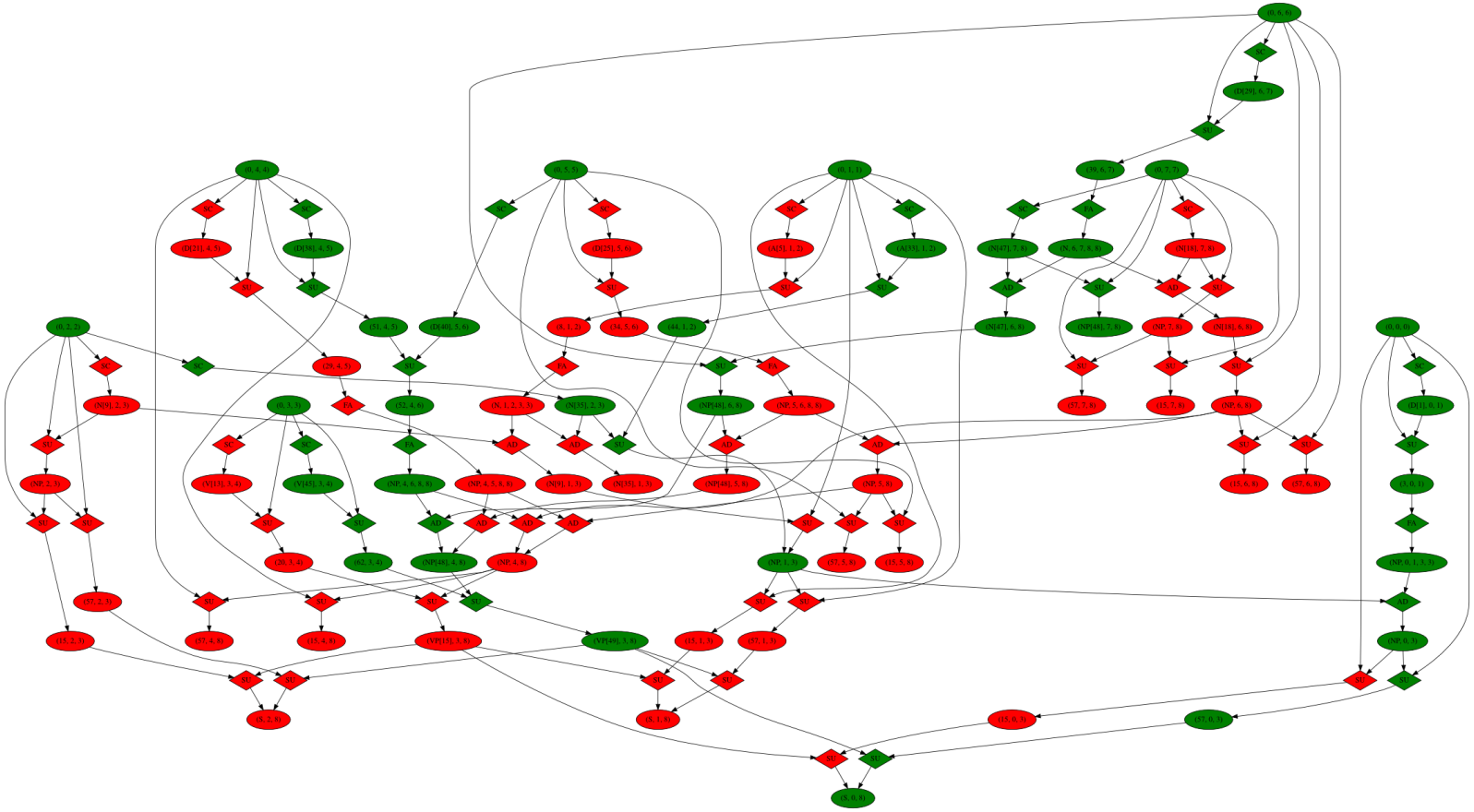
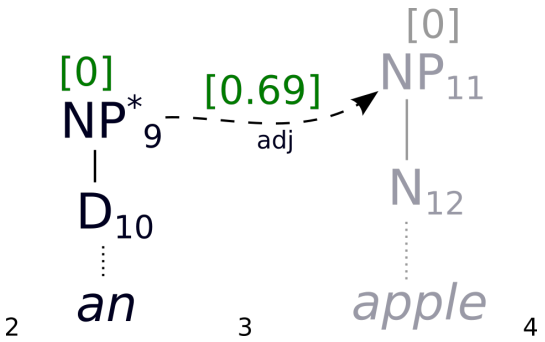


Figure: Goal: visit only the part of the chart (marked in green) corresponding to the best-score derivation

Calculating inside weights: Sister adjunction

Deduction rule

$$\frac{w_1 \ : \ (N \rightarrow \alpha \bullet \beta, i, j, \Gamma_1) \quad w_2 \ : \ (M, j, k, \Gamma_2, no)}{w_1 + w_2 + \omega(M, N) \ : \ (N \rightarrow \alpha \bullet \beta, i, k, \Gamma_1 \oplus \Gamma_2)} \ : \ \begin{matrix} label(M) = label(N) \\ sister(M) \end{matrix}$$



Rule instance

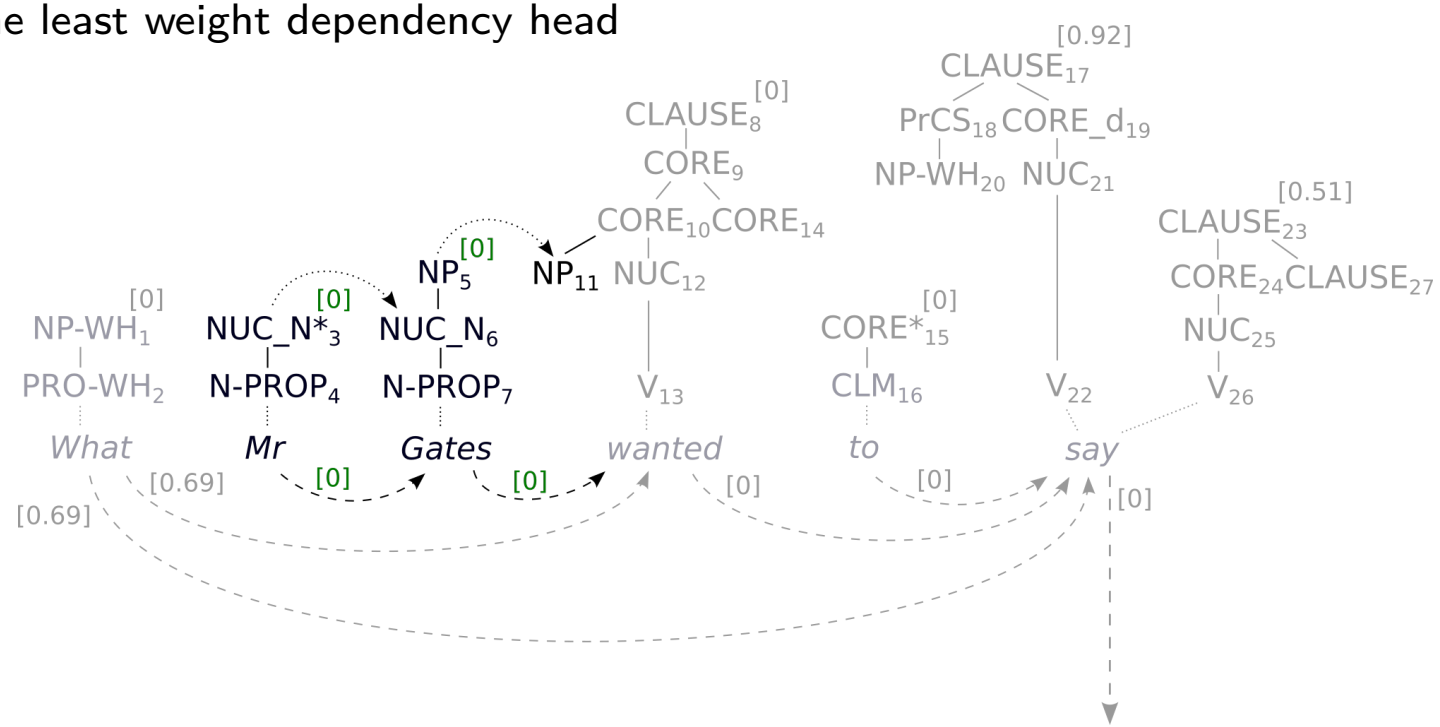
$$\frac{0 \ : \ (NP_{11} \rightarrow \bullet N_{12}, 2, 2, \emptyset) \quad 0 \ : \ (NP^*_9, 2, 3, \emptyset, no)}{0.69 \ : \ (NP_{11} \rightarrow N_{12} \bullet, 2, 3, \emptyset)}$$

A* heuristic

Idea (CCG, Lewis et al., 2016)

For each not-yet-scanned word assume:

- ⋆ the least weight supertag
- ⋆ the least weight dependency head



$$h(CORE_{10} \rightarrow NP_{11} \bullet NUC_{12}, 1, 3, \emptyset) =$$

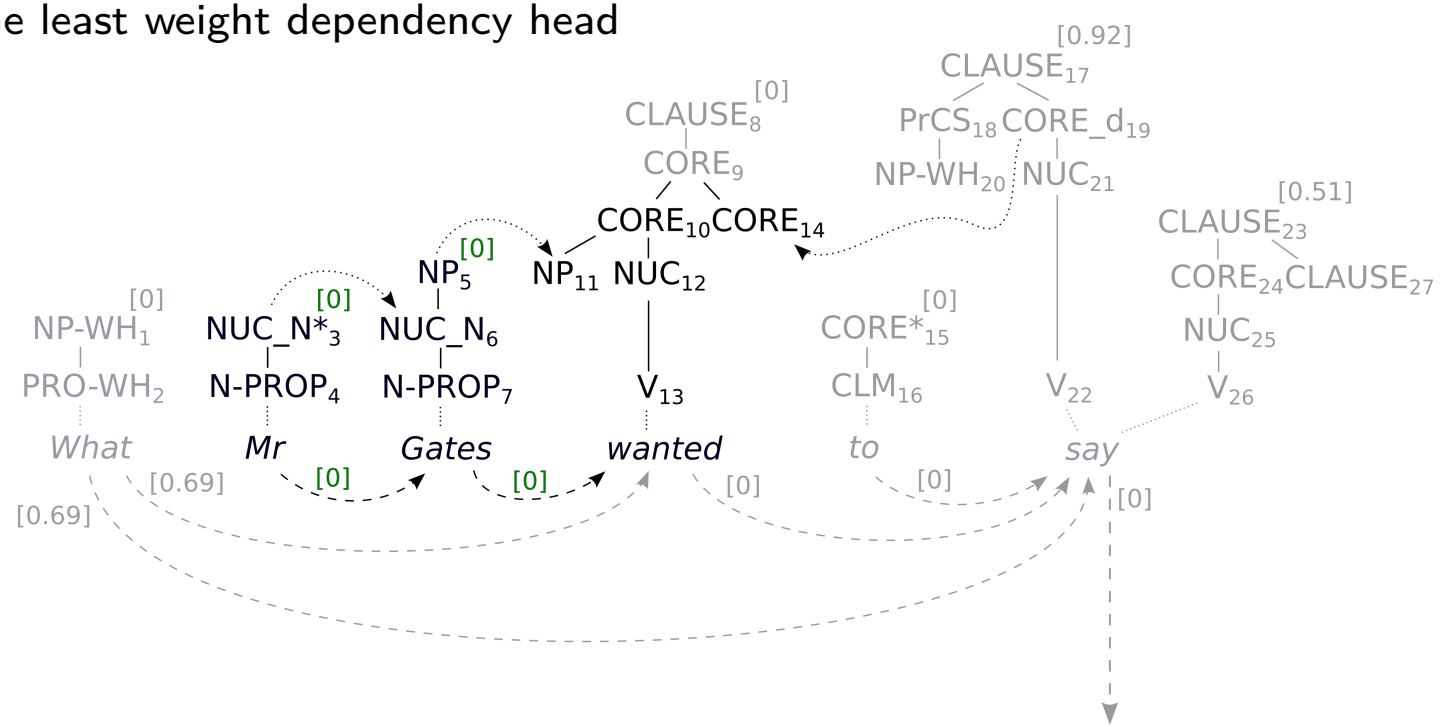
$$0.69 \quad + \quad 0 \quad + \quad 0 \quad + \quad 0.51$$

A* heuristic

Idea (CCG, Lewis et al., 2016)

For each not-yet-scanned word assume:

- ⋆ the least weight supertag
- ⋆ the least weight dependency head



$$h(CORE_9 \rightarrow CORE_{10}CORE_{14}\bullet, 1, 4, \{(4, 6, CORE)\}) =$$

$$0.69 \quad + \quad 0 \quad + \quad 0 \quad + \quad 0.92$$

A* heuristic: TWG

- To each chart item $\eta = (x, i, j, \Gamma)$ a pair (w, m) is assigned:
  - w is the corresponding inside weight
  - m assigns (predicted) weights to gaps in Γ
- The heuristic h (estimate of outside weight) is defined as:

$$h(\eta) = \sum_{0<k\leq i} \mathcal{C}(k) + \sum_{j<k\leq n} \mathcal{C}(k) + A(x) + sum(m)$$

  - $\mathcal{C}(k)$ is the minimal cost of scanning k
  - $A(x)$ is the „amortized weight" of x (e.g. dotted rule)
  - $sum(m)$ is the sum of weights in m ($\sum_{(f_1,f_2,y)\in\Gamma} m(f_1)$)

A* heuristic: TWG

Gap-related book-keeping in deduction rules:

- Predict wrapping:

$$\frac{(w_1,m_1) : (N\rightarrow\alpha\bullet M\beta,i,j,\Gamma_1) \qquad (w_2,m_2) : (D,j,k,\Gamma_2,yes)}{(w_1,m_1[j\Rightarrow w_2+A(D)+sum(m_2)]) : (N\rightarrow\alpha M\bullet\beta,i,k,\Gamma_1\oplus[(j,k,\ell(D))])} \quad \begin{array}{c} leaf(M) \\ \ell(M)=\ell(D) \end{array}$$

- Complete wrapping:

$$\frac{(w_1,m_1) : (R,i,j,\Gamma_1\oplus[(f_1,f_2,y)]\oplus\Gamma_2,ws?) \qquad (w_2,m_2) : (D,f_1,f_2,\Gamma_3,yes)}{(w_1+w_2+\omega(R,D),m_1[f_1\Rightarrow\bot]\oplus m_2) : (D,i,j,\Gamma_1\oplus\Gamma_3\oplus\Gamma_2,no)} \quad \begin{array}{c} root(R) \wedge y=\ell(D) \\ \ell(parent(D))=\ell(R) \end{array}$$

Next Steps

- Finalize the implementation of the $A^\star$ parser for TWG (I/O, chart indexing, formal monotonicity proof)
- Identify more patterns in RRGbank which mark tree-wrapping nodes.
- Transfer some of the trace markings from PTB to RRGbank $\rightarrow$ better identification of tree-wrapping cases.
- Consider extracting a TWG from PTB and doing parsing experiments.

Thank you!

THANK YOU VERY MUCH FOR YOUR ATTENTION!

References I

Bies, A., Ferguson, M., Katz, K., MacIntyre, R., Tredinnick, V., Kim, G., Marcinkiewicz, M. A., and Schasberger, B. (1995). Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97:100.

Bladier, T., van Cranenburgh, A., Evang, K., Kallmeyer, L., Möllemann, R., and Osswald, R. (2018). RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank. In *Proceedings of the 17th International Workshop on Treebanks and Linguistic Theories (TLT 2018), December 13–14, 2018, Oslo University, Norway*, number 155, pages 5–16. Linköping University Electronic Press.

Boyd, A. (2007). Discontinuity revisited: An improved conversion to context-free representations. In *Proceedings of The Linguistic Annotation Workshop (LAW) at ACL 2007*, pages 41–44, Prague, Czech Republic.

Johnson, M. (2002). A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 136–143. Association for Computational Linguistics.

Kallmeyer, L., Osswald, R., and Van Valin, R. D. (2013a). Tree wrapping for role and reference grammar. In *Formal Grammar*, pages 175–190. Springer.

Kallmeyer, L., Osswald, R., and Van Valin, Jr., R. D. (2013b). Tree Wrapping for Role and Reference Grammar. In Morrill, G. and Nederhof, M.-J., editors, *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Lewis, M., Lee, K., and Zettlemoyer, L. (2016). LSTM CCG parsing. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 221–231.

Radford, A. (2004). *Minimalist syntax: Exploring the structure of English*. Cambridge University Press.

Rambow, O., Vijay-Shanker, K., and Weir, D. (2001). D-tree substitution grammars. *Computational Linguistics*, 27(1):87–121.

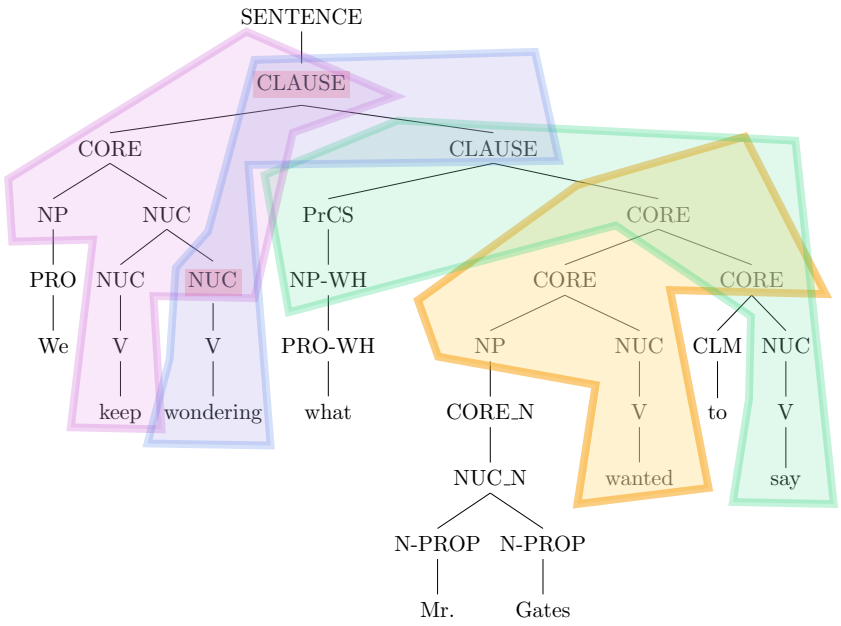
Xia, F. (1999). Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, pages 398–403.

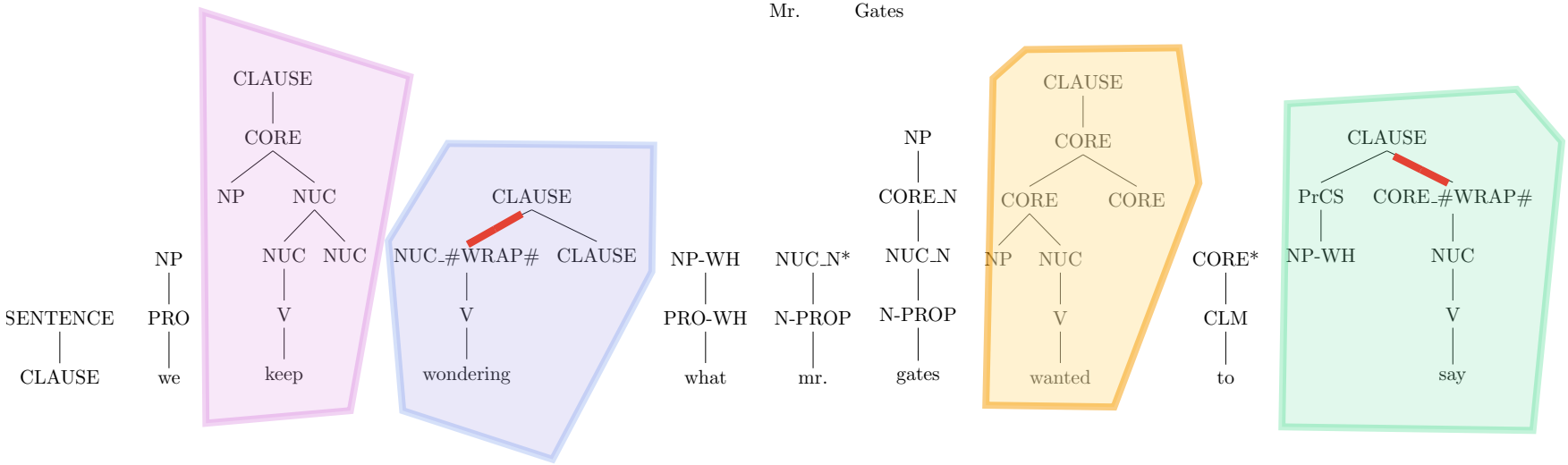
POS tags

- The set of POS tags (currently 40):

  “, ”, „ -LRB-, -RRB-, ., :, A, ADV, ADV-REL, ADV-WH, AUX, AUX-ASP, AUX-TNS, CD, CLM, DEIX, FW, N, N-PROP, N-TNS, OP-ASP, OP-DEF, OP-MOD, OP-NEG, OP-TNS, P, PDT, POS, PRO, PRO-DEF, PRO-DEM, PRO-REL, PRO-WH, PRT, QNT, UH, V, V-GER, V-PART

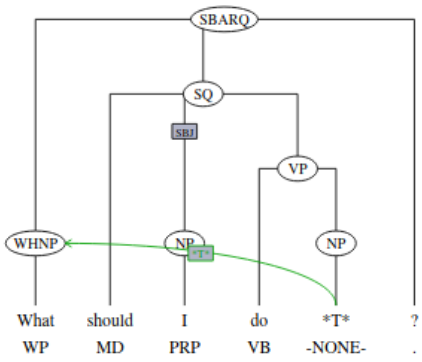
TWG: Tree-Wrapping Elementary Trees (2nd version)

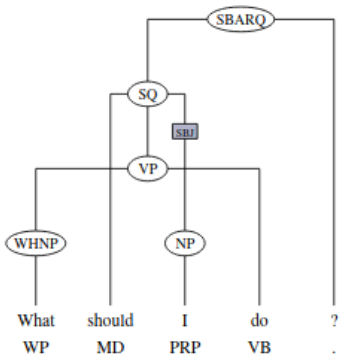


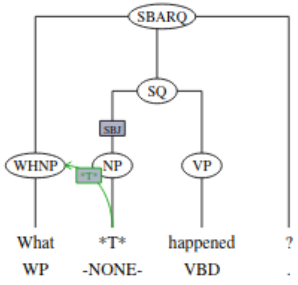


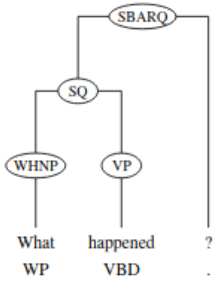
Tree Wrapping in PTB

foreach *sentence s* **do**
  foreach *node c in s coindexed with a *T*, *ICH* or *EXP* null element n* **do**
    select the grandparent of n as the destination node d;
    remove c from its current parent and reattach it to d;
  end
end
delete all null elements;
delete non-terminal nodes without children until none exist;
remove all remaining indices from node labels;









Tree Wrapping in PTB

Most common empty node patterns in PTB training corpus with A-bar movement traces marked with *T* (Johnson, 2002).

	Count	**Pattern**
(1)	5312	(SBAR WHNP-1 (S (NP (-NONE- *T*-1)) VP))
(2)	662	(SBAR (WHNP-1 (-NONE- 0)) (S (NP (-NONE- *T*-1)) VP))
(3)	618	(SBAR (S-1 , NP (VP VBD (SBAR (-NONE- 0) (S (-NONE- *T*-1)))) .)
(4)	499	(SINV “ S-1 , ” (VP VBZ (S (-NONE- *T*-1))) NP .)
(5)	361	(SINV “ S-1 , ” (VP VBD (S (-NONE- *T*-1))) NP .)
(6)	269	(S “ S-1 , ” NP (VP VBD (S (-NONE- *T*-1))) .)

Examples from PTB:

	Example
(1)	Kalipharma is a New Jersey-based pharmaceuticals concern that sells products under the Purepac label .
(2)	There is evidence to support that view .
(3)	
(4)	“ We ’re willing to negotiate , ” says Dennis Gillespie , executive vice president of marketing .
(5)	“ We ’re willing to negotiate , ” said Dennis Gillespie .
(6)	“ You ca n’t give it up that easily , ” he says .

TWG Extraction: Head rules

```
% Heuristic head rules for extraction of elementary trees from the RRGbank
QP CD
ROOT right-to-left CLAUSE
CLAUSE right-to-left NUC CORE
CORE left-to-right NUC NP CLAUSE
CORE right-to-left CORE
NP left-to-right CLM CORE_N NP PRO
CORE_P left-to-right NUC_P
CORE_N right-to-left NUC_N NP
NUC_N right-to-left N
NUC left-to-right V
LDP left-to-right SENTENCE
SENTENCE left-to-right CLAUSE
ADVP-PERI left-to-right CORE_ADV
AP-PERI left-to-right CORE_A
CORE_A left-to-right NUC_A
NUC_A left-to-right A
CLAUSE_#WRAP# right-to-left CORE_#WRAP#
CORE_#WRAP# right-to-left CORE
```

TWG Extraction: Modifier rules

```
% Heuristic rules for distinguishing modifiers from complements

% extended POS-tag set (C&C) includes the following POS tags:
% PONCT, CLS, DET, I, A, PRO, PREF, VPP, NC, ADV, ADVWH,
% C, CC, ADJWH, P, ADJ, NPP, CLR, ET, PROWH, CS, DETWH,
% V, CLO, PROREL, D

* QNT-OP QNT AP LDP *-OP , . CLM AUX `` '' ADVP-PERI *-TNS PP-PERI *-PERI POS TNS-OP MOD-OP DEF-OP PrDP

CORE_N CD

NP NP POS QNT-OP QNT

NUC_N N

ROOT PP-PERI

SENTENCE PP-PERI

CORE AP CORE CLM

NPIP POS

CLAUSE CLAUSE

CORE_#WRAP# CORE
```