Toward Tree Substitution Grammars with Latent Annotations

Francis Ferraro, Benjamin Van Durme and Matt Post

human language technology
center of excellence
JOHNS HOPKINS
UNIVERSITY

Motivation

- Treebanks are coarse in multiple dimensions
- Complementary solutions have been proposed
  - Latent annotations (symbol refinements)
  - Tree substitution grammars
- Can we combine these approaches?

Learning Latent Annotations

Learning Latent Annotations

PRP

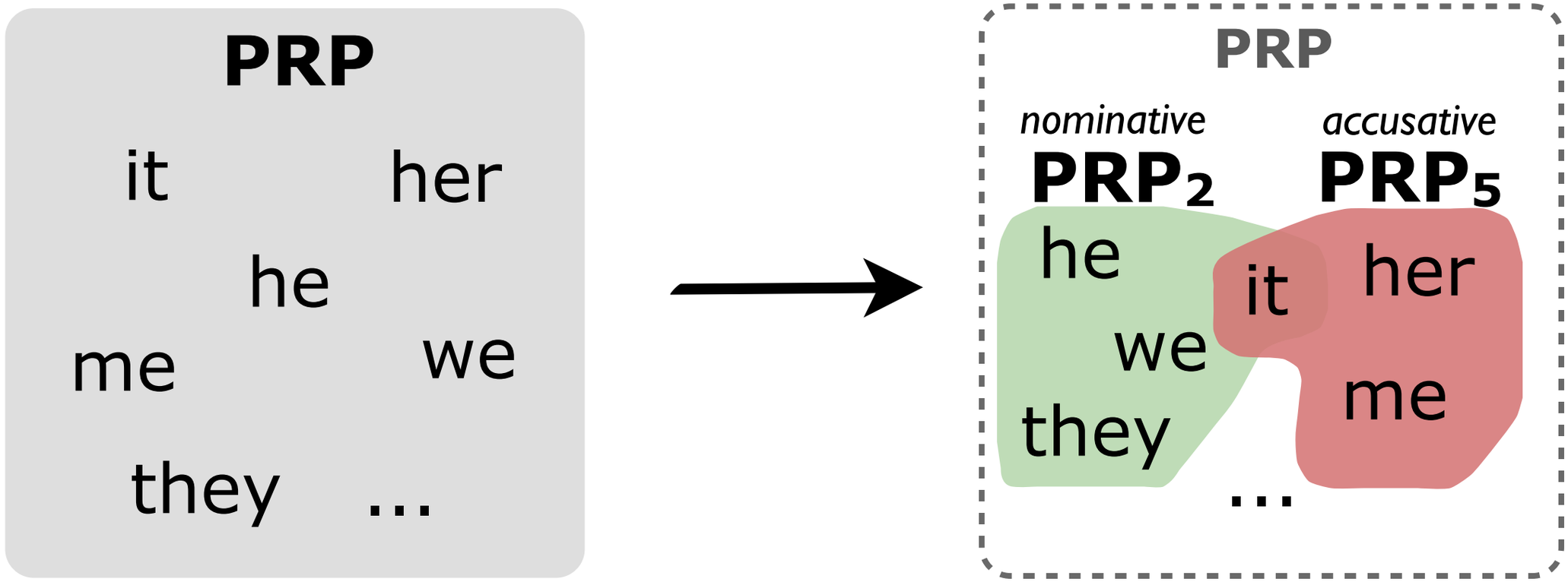
it her

he

me we

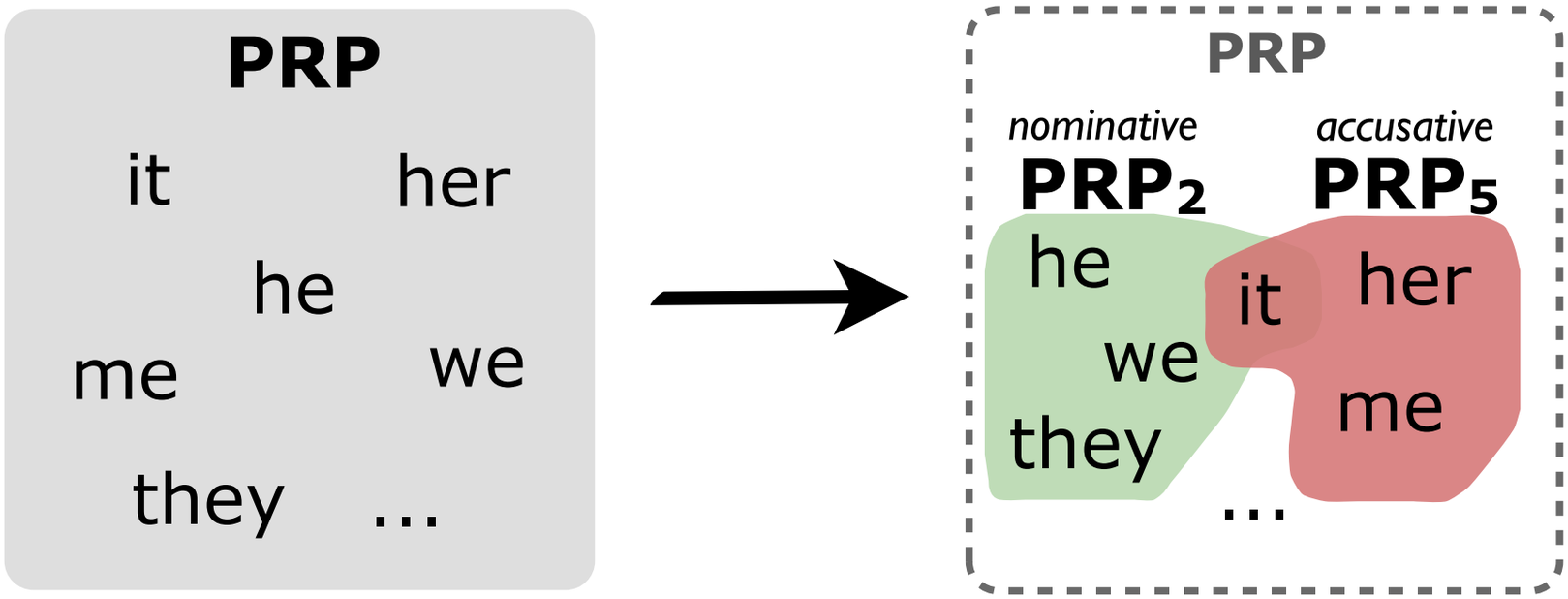
they ...

Learning Latent Annotations



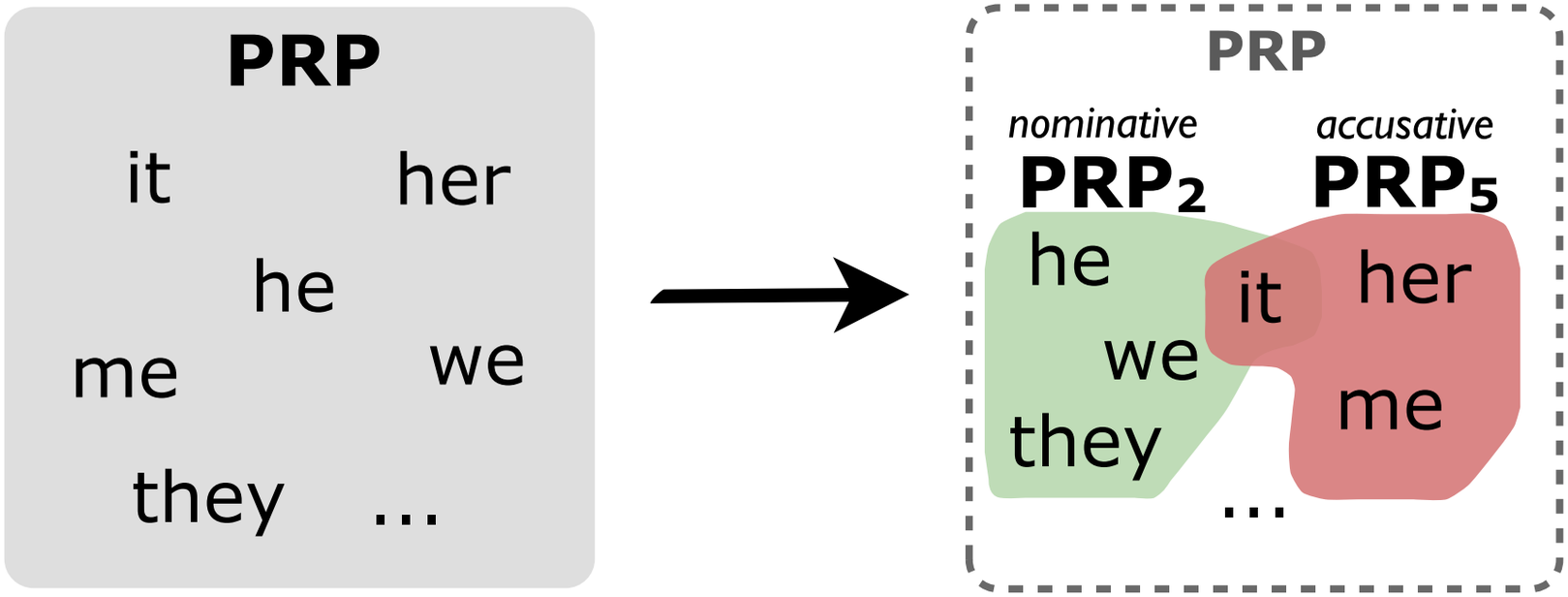
Learning Latent Annotations

- Can we compensate for PCFG permissiveness?
- Johnson (1998): parent annotation
- Klein and Manning (2003): linguistically motivated annotations

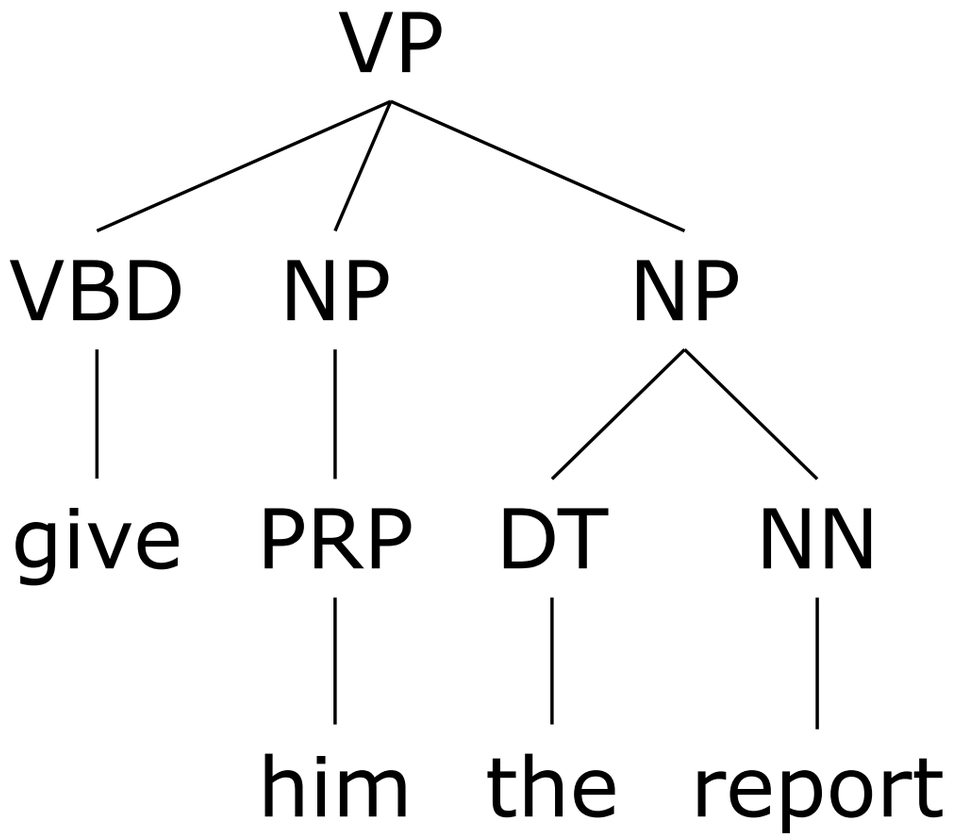


Learning Latent Annotations

- Can we learn them automatically?
- Split all categories equally and learn weights via EM (Matsuzaki et al., 2005)
- Iteratively improve refinements: "split-merge" framework (Petrov et al., 2006)
- Intuitive lexical clusterings

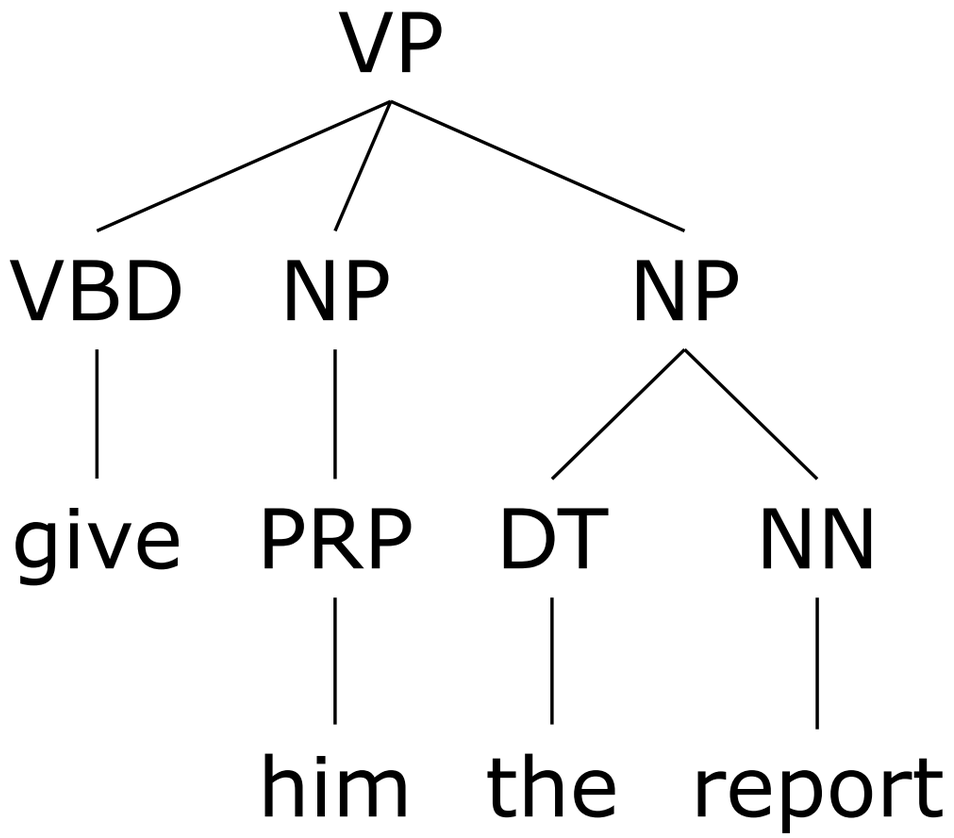


Tree Substitution Grammars



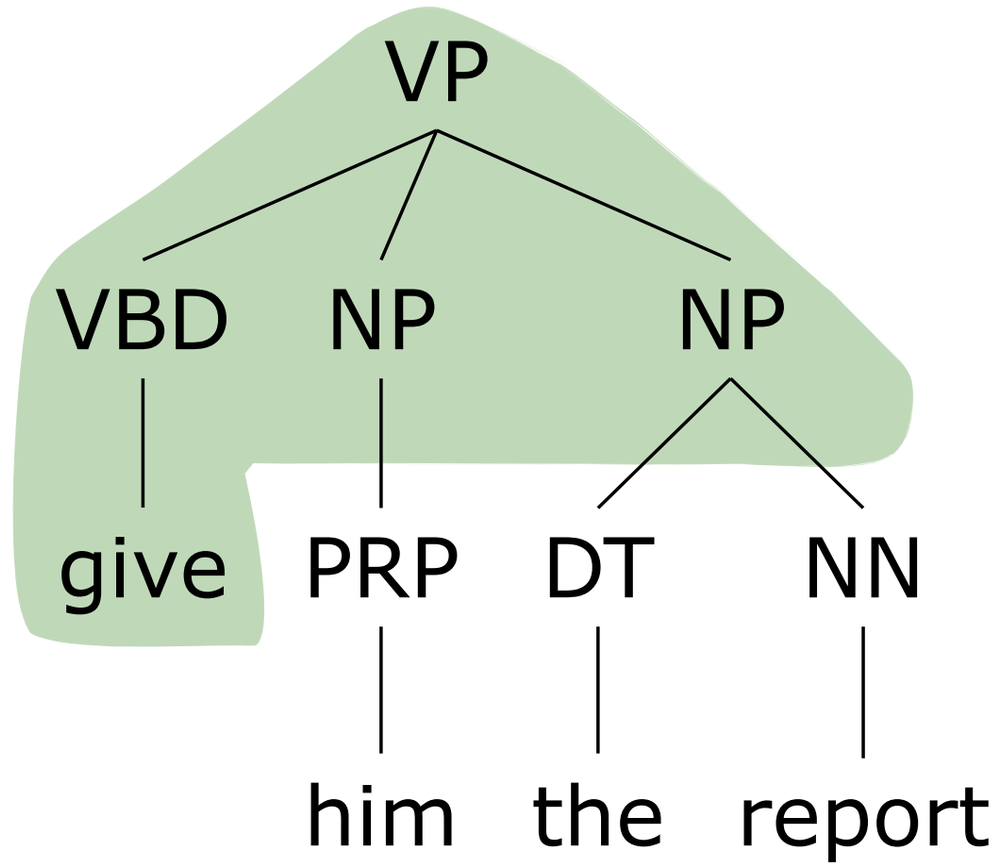
Tree Substitution Grammars

- Non-terminals rewrite as tree fragments



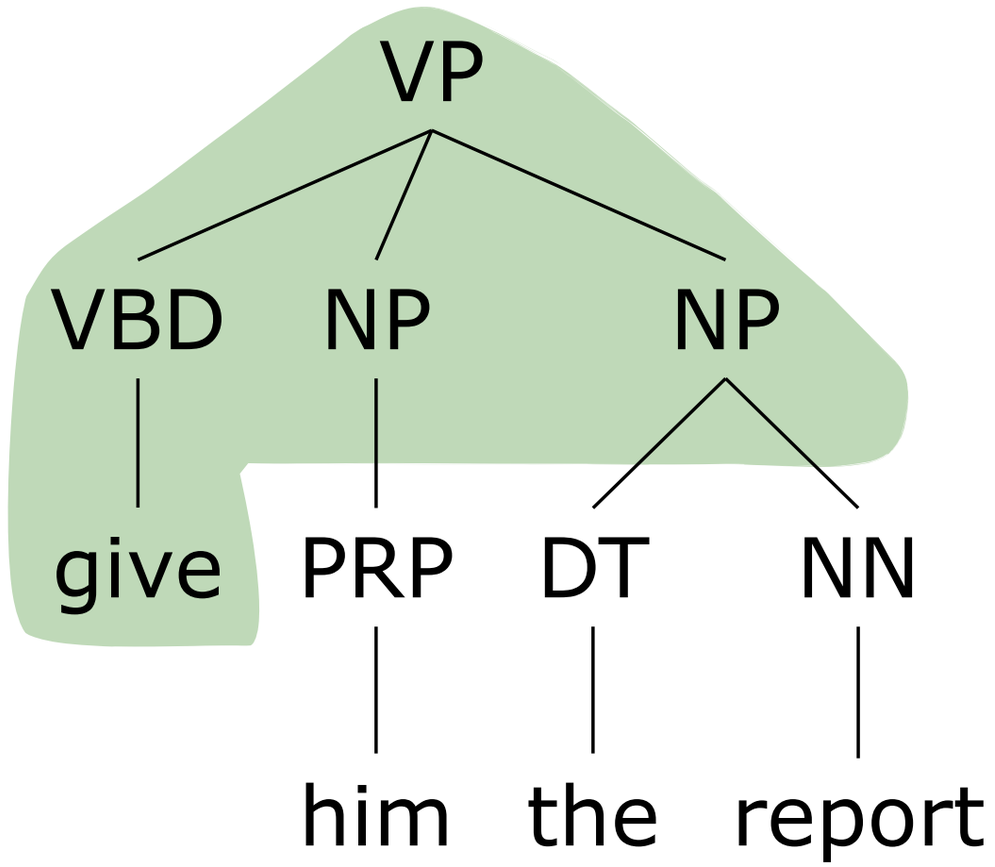
Tree Substitution Grammars

- Non-terminals rewrite as tree fragments



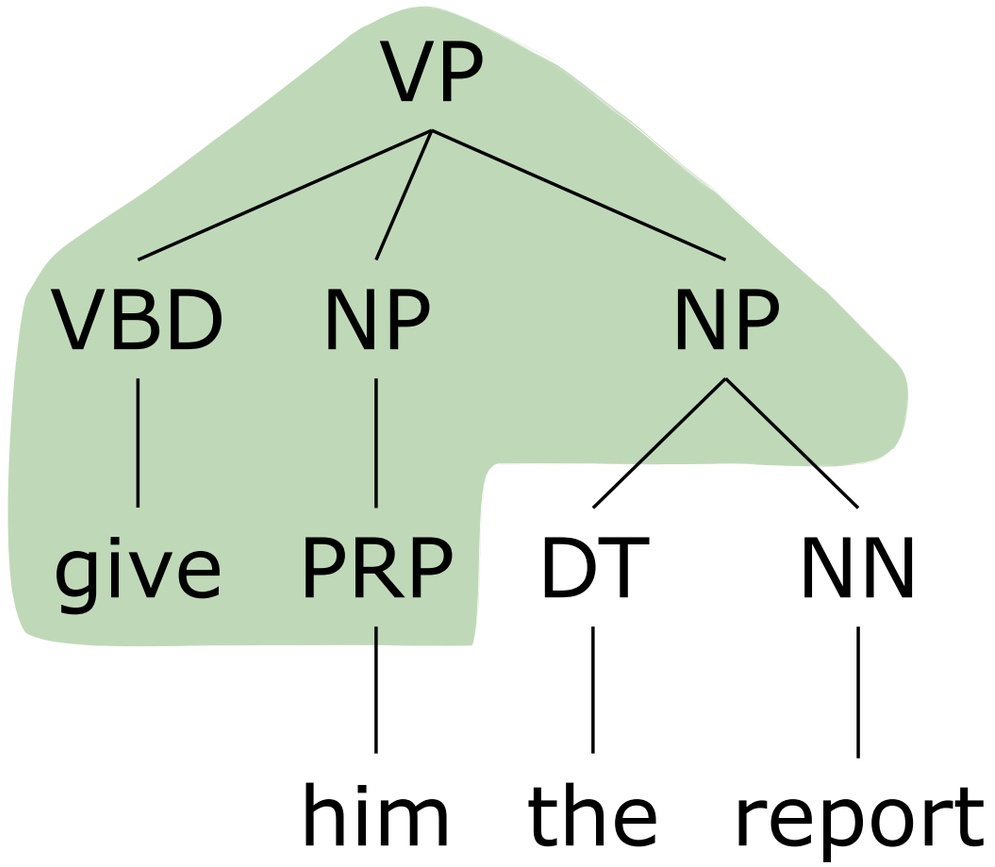
Tree Substitution Grammars

- Non-terminals rewrite as tree fragments
- Extended domain of locality
- Capture long-range grammatical dependencies



Tree Substitution Grammars

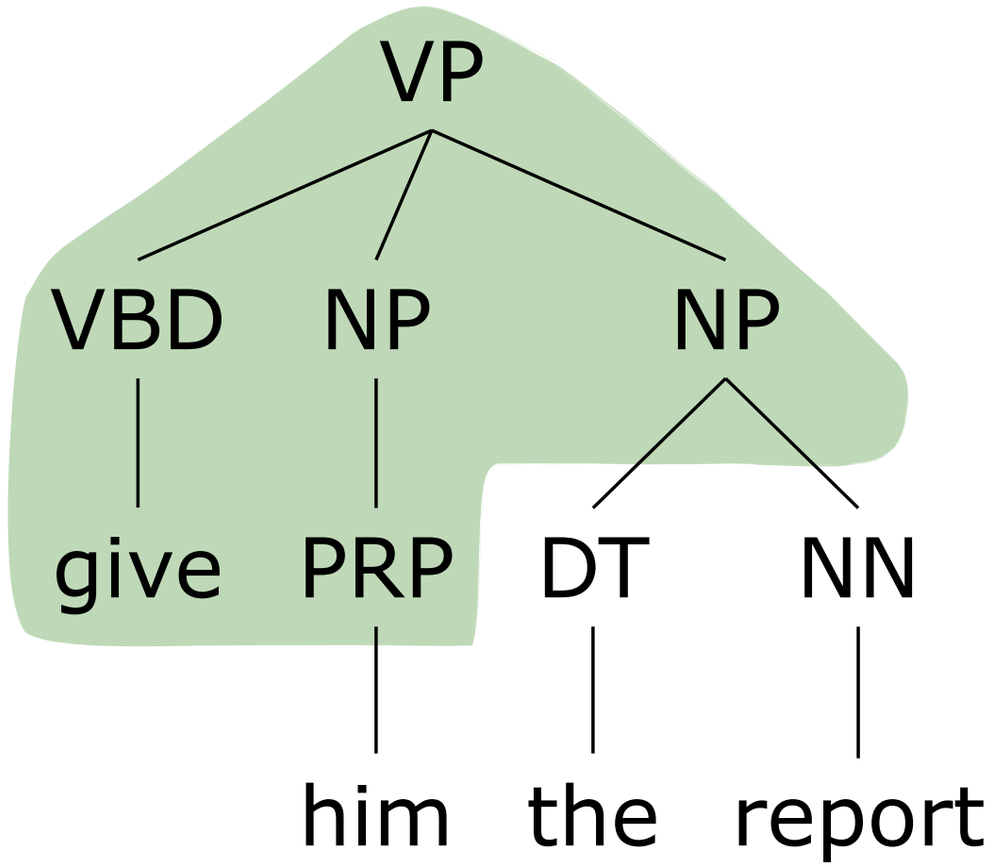
- Non-terminals rewrite as tree fragments
- Extended domain of locality
- Capture long-range grammatical dependencies



Tree Substitution Grammars

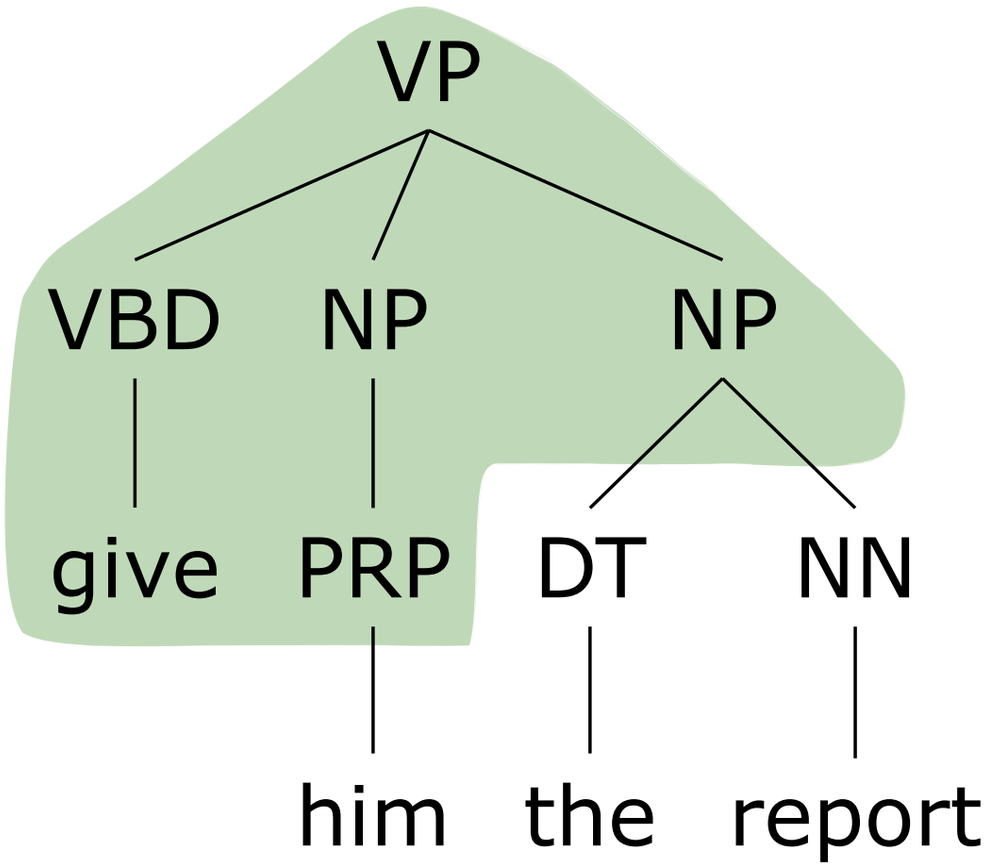
- Learning TSGs is not straight-forward
- DOP: All fragments (Bod, 1993)

  ✗ Even with heuristic selection, get large, over-fit grammars (Bod, 2001)



Tree Substitution Grammars

- Learning TSGs is not straight-forward
- DOP: All fragments (Bod, 1993)
  - ✗ Even with heuristic selection, get large, over-fit grammars (Bod, 2001)
- Non-parametric: DP Prior (Cohn et al., 2009; Post et al., 2009)
  - ✓ Compact fragments
  - ✗ Non-deterministic
  - ✗ Complex to implement.

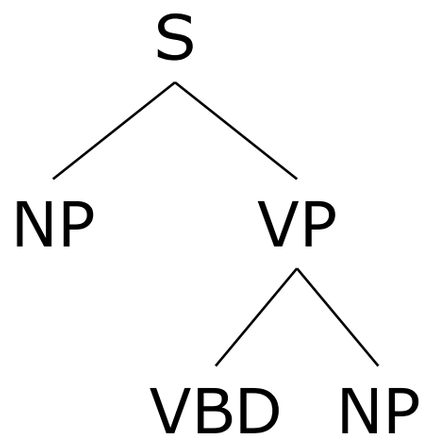


Framework

- Build within the Berkeley parser codebase
- Make the root of every internal depth-one subtree unique
  - Place the entirety of the TSG weight on the root depth-one rule.

Framework

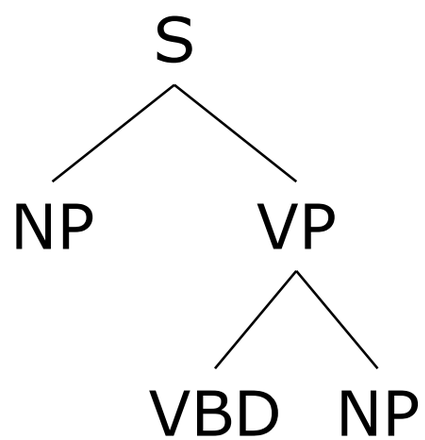
- Build within the Berkeley parser codebase
- Make the root of every internal depth-one subtree unique
  - Place the entirety of the TSG weight on the root depth-one rule.



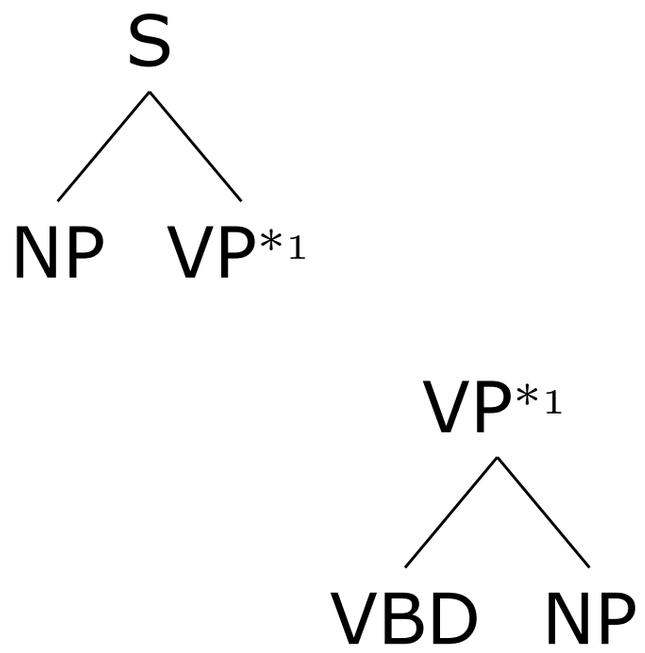
0.5

Framework

- Build within the Berkeley parser codebase
- Make the root of every internal depth-one subtree unique
  - Place the entirety of the TSG weight on the root depth-one rule.

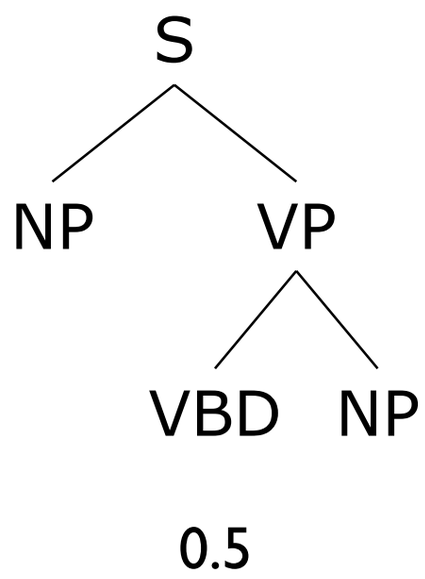


0.5

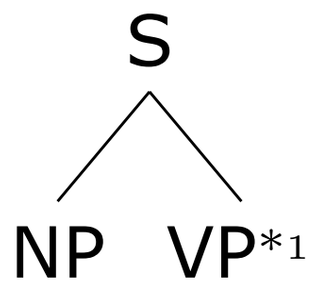


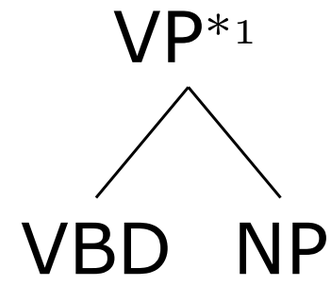
Framework

- Build within the Berkeley parser codebase
- Make the root of every internal depth-one subtree unique
  - Place the entirety of the TSG weight on the root depth-one rule.



0.5

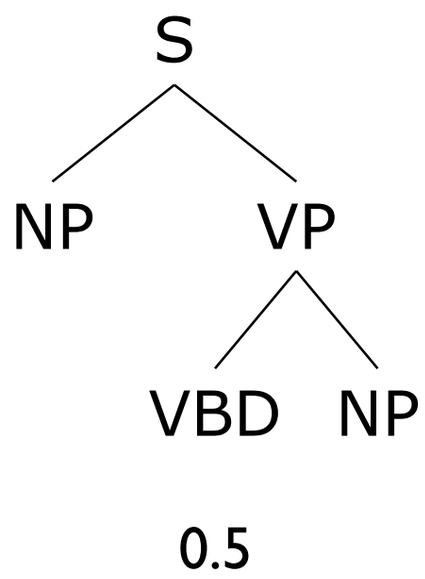


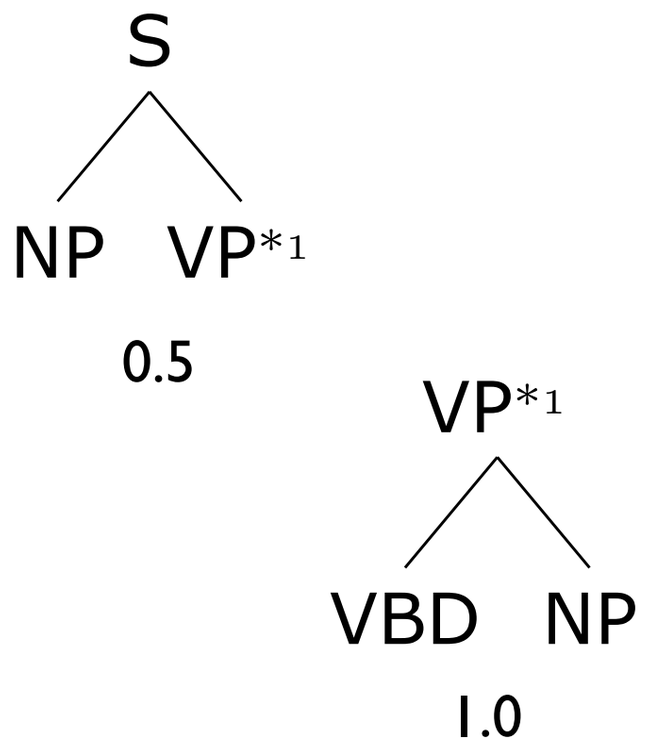


1.0

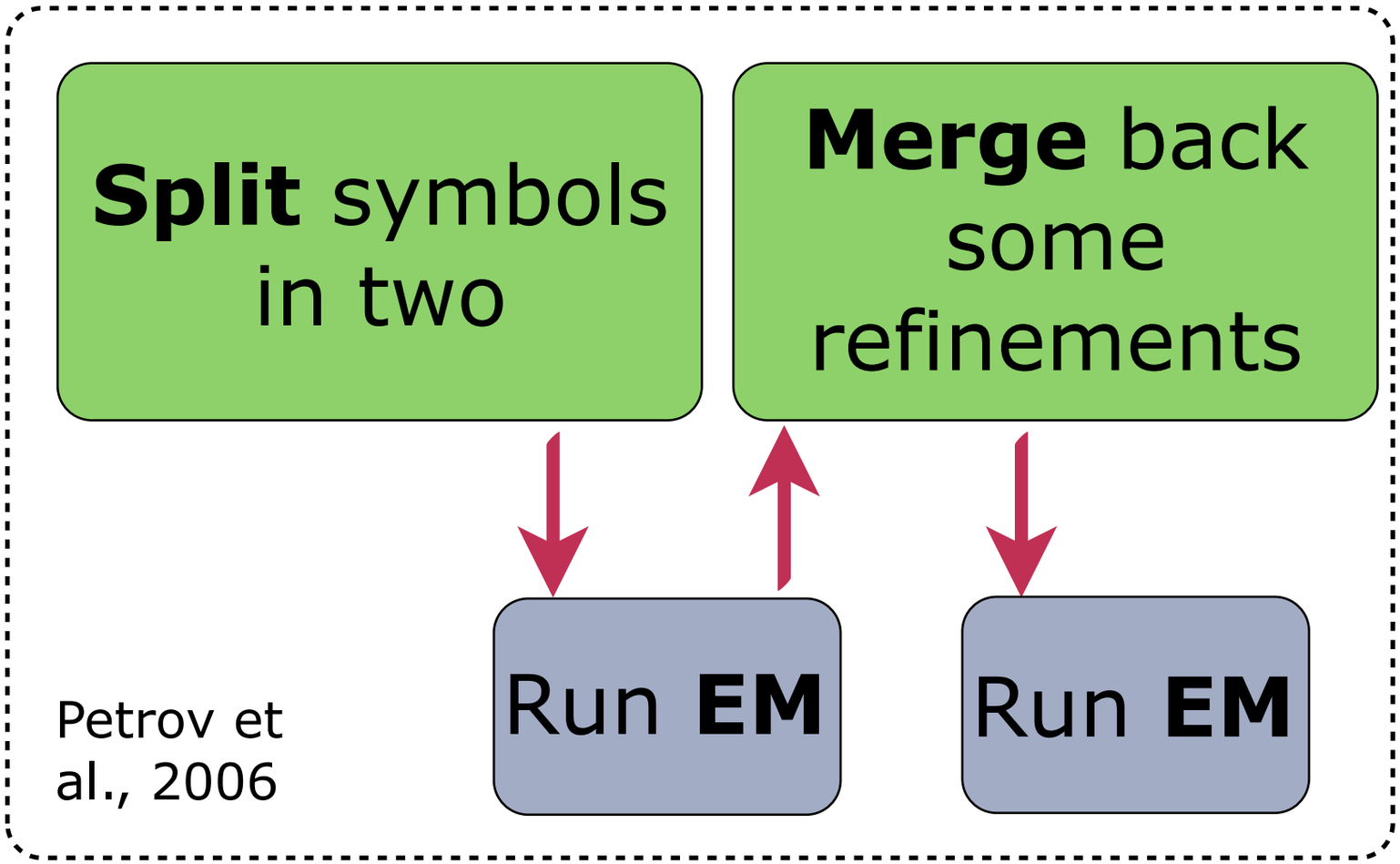
Framework

- Build within the Berkeley parser codebase
- Make the root of every internal depth-one subtree unique
  - Place the entirety of the TSG weight on the root depth-one rule.

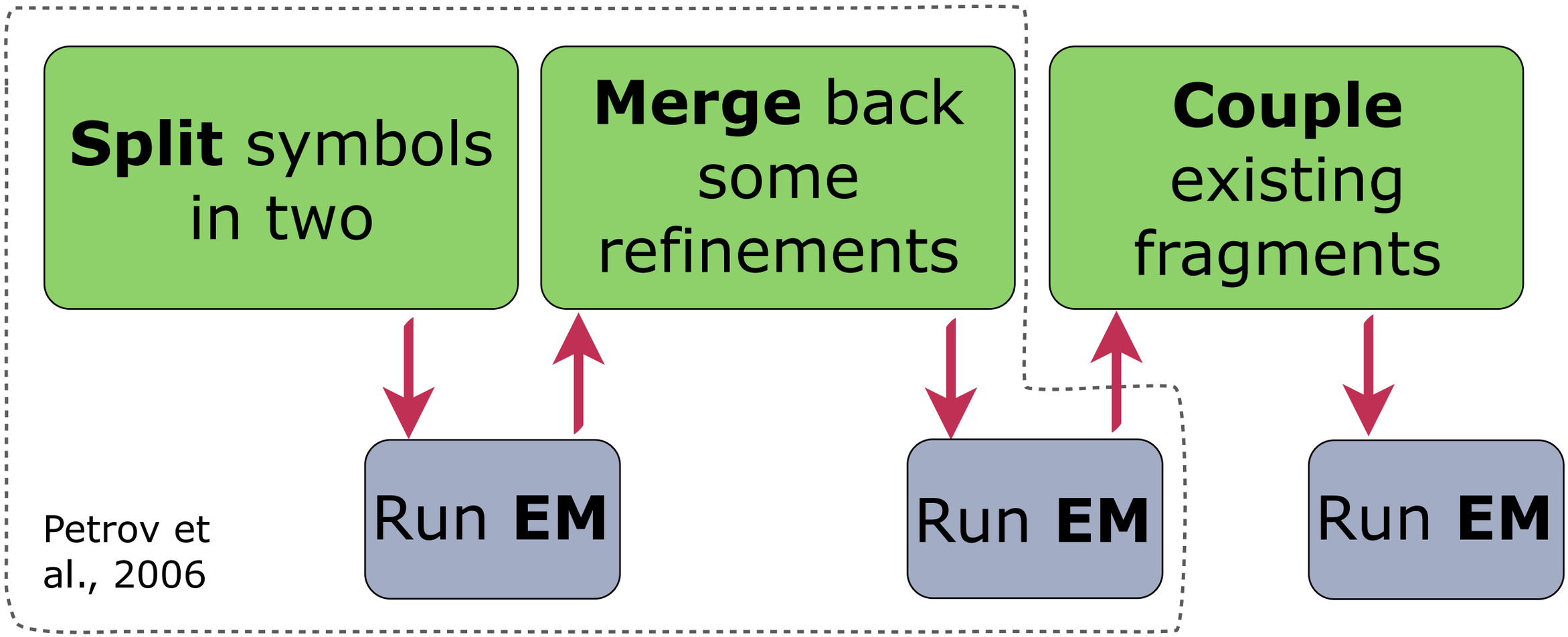




Algorithm Overview



Algorithm Overview



Control Exponential Growth

- Use binary trees
- Forbid multiple frontier nodes from simultaneously becoming internal nodes ("chained" couplings)
- Allow couplings only if permitted by a constraint set **c**.

Control Exponential Growth

- Use binary trees
- Forbid multiple frontier nodes from simultaneously becoming internal nodes ("chained" couplings)
- Allow couplings only if permitted by a constraint set **c**.

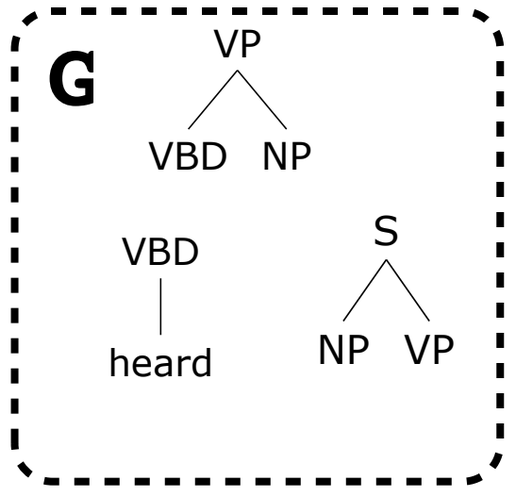
Coupling

Coupling

Given a grammar $\mathfrak{G}$, constraint set $\mathfrak{C}$:

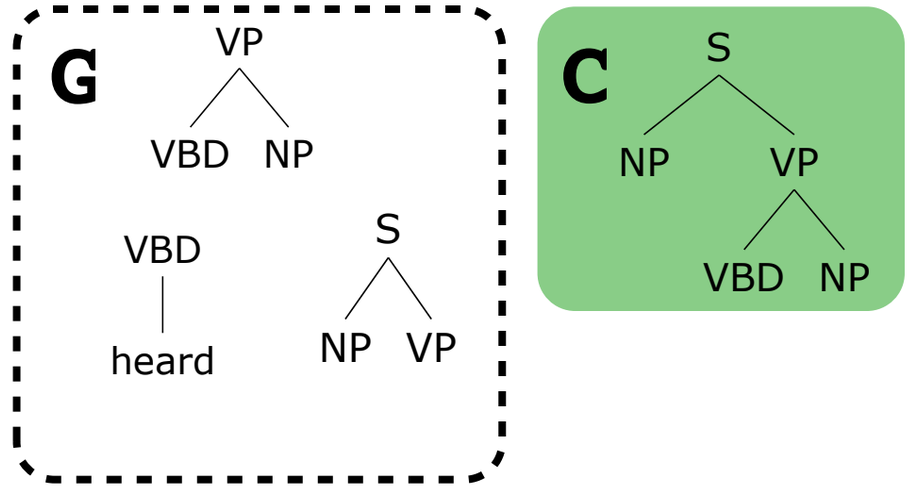
Coupling

Given a grammar **G**, constraint set **C**:



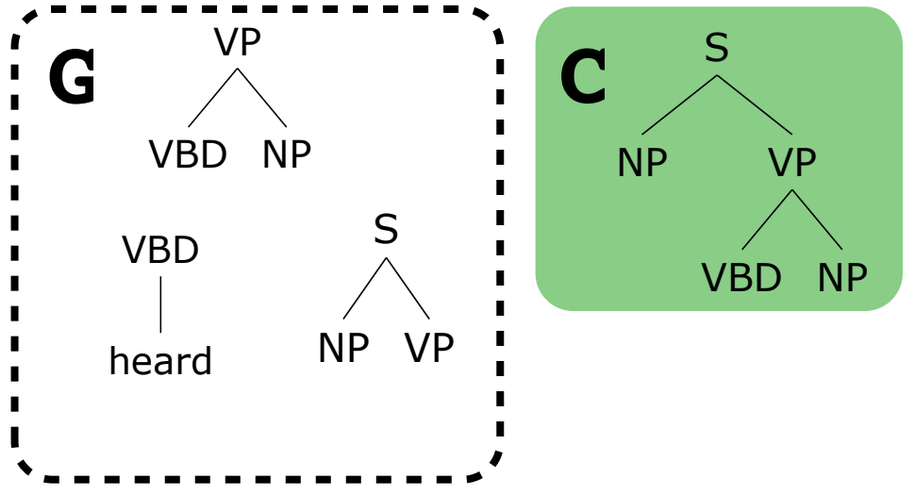
Coupling

Given a grammar **G**, constraint set **C**:



Coupling

Given a grammar **G**, constraint set **C**:



1. Construct a grammar **G′** from **G** and allowed couplings from **C**

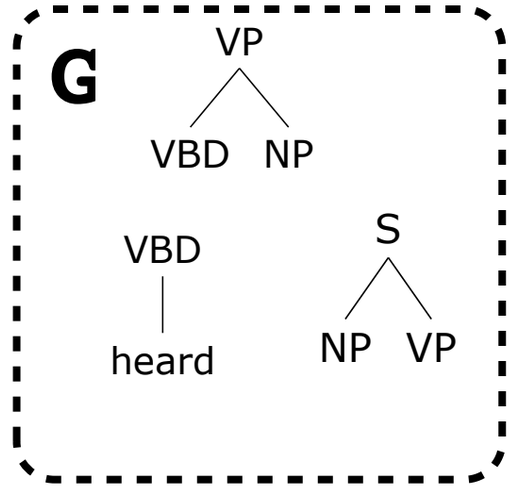
$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$

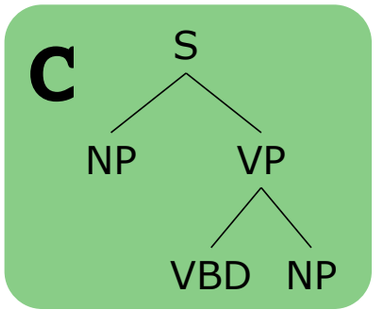
Coupling

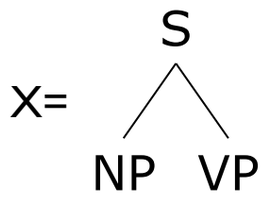
Given a grammar **G**, constraint set **C**:

1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$





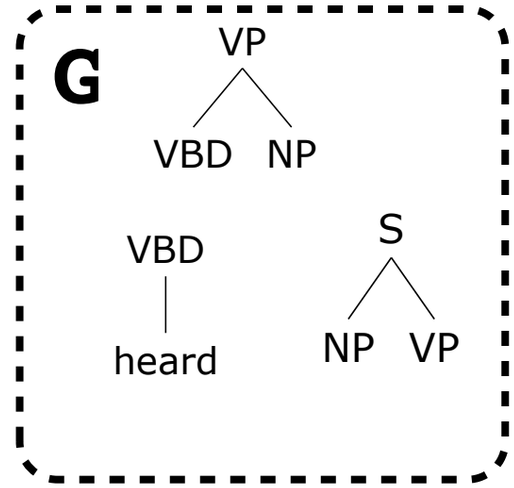


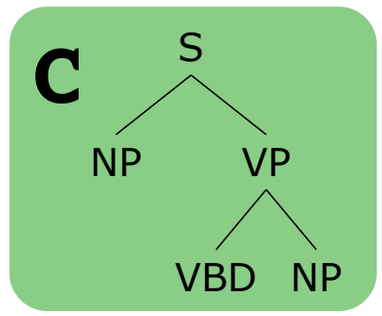
Coupling

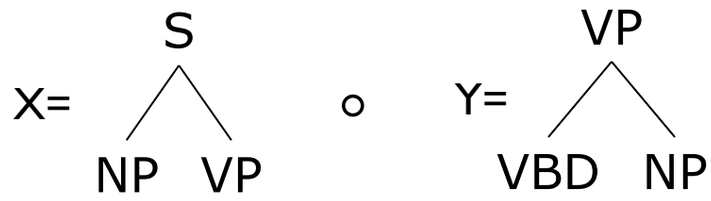
Given a grammar **G**, constraint set **C**:

1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$

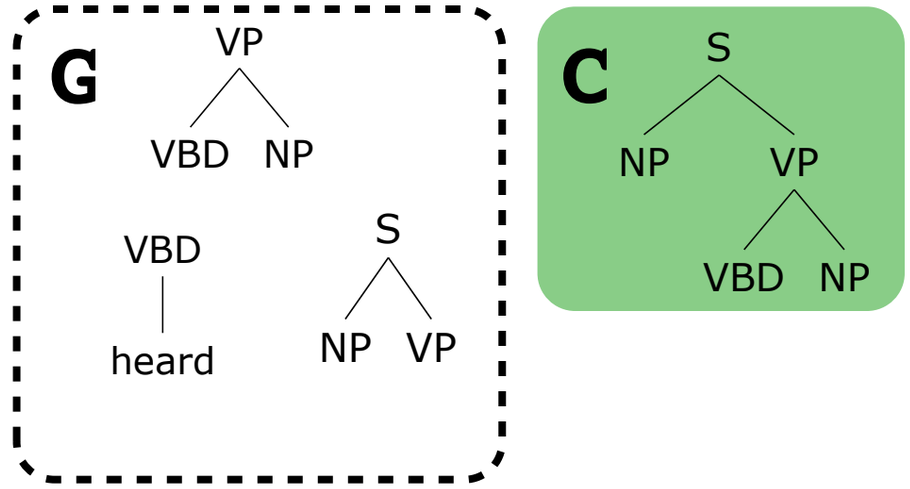






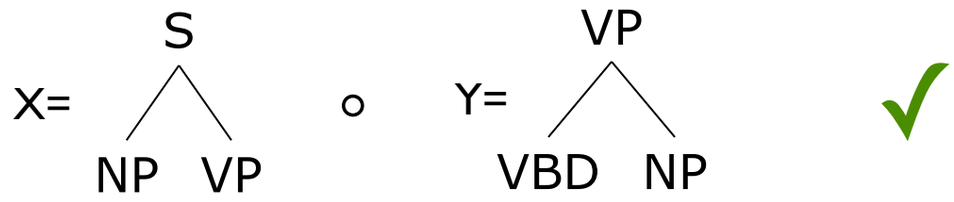
Coupling

Given a grammar **G**, constraint set **C**:



1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$

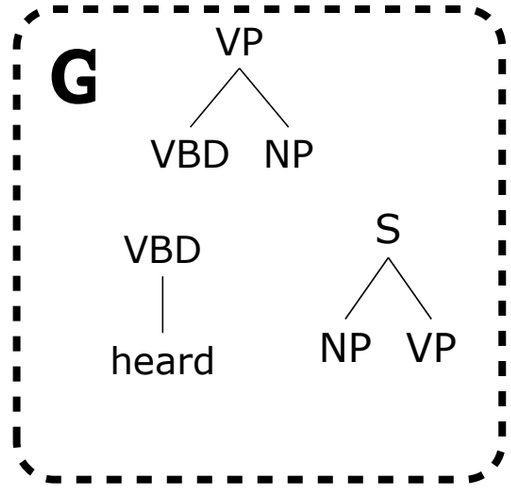


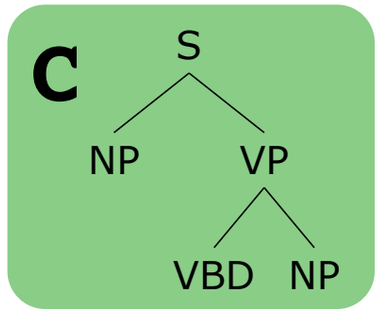
Coupling

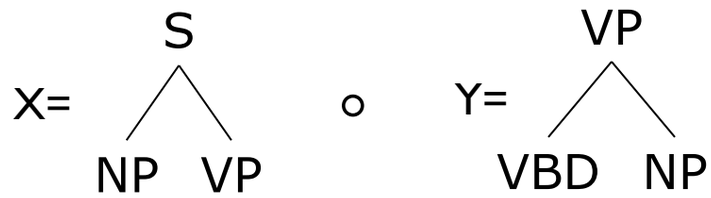
Given a grammar **G**, constraint set **C**:

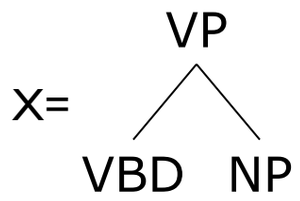
1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$



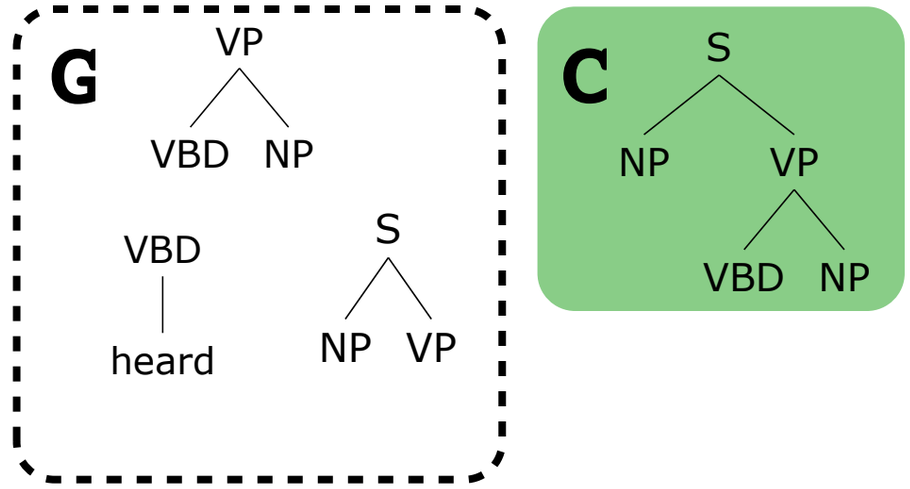






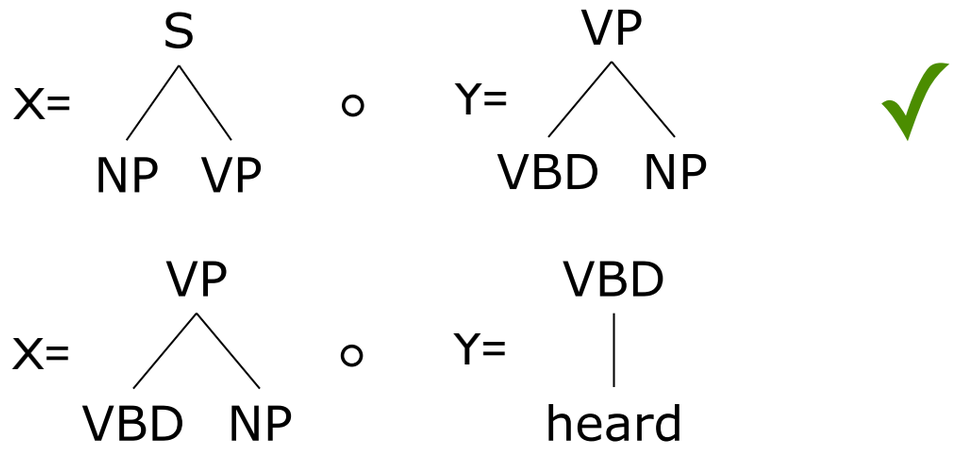
Coupling

Given a grammar **G**, constraint set **C**:



1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$

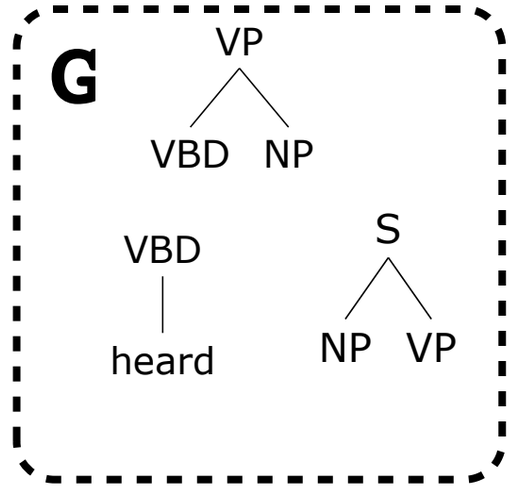


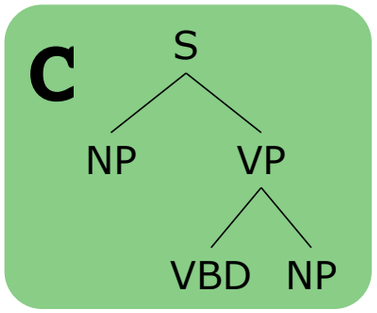
Coupling

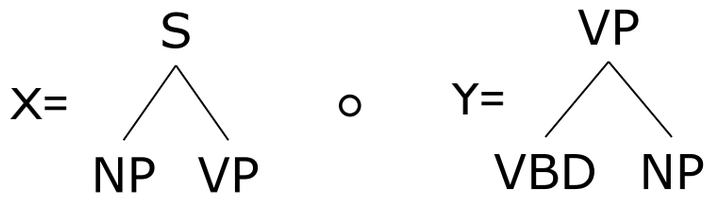
Given a grammar **G**, constraint set **C**:

1. Construct a grammar **G′** from **G** and allowed couplings from **C**

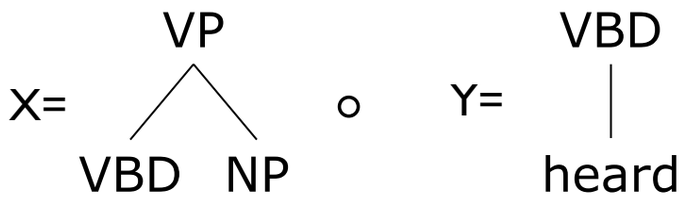
$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$







✓



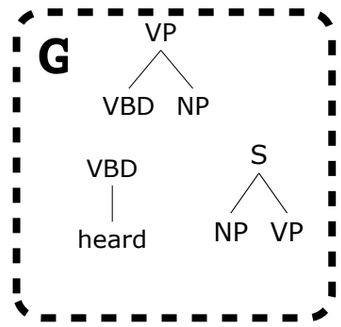
✗

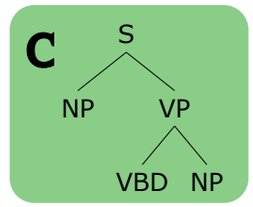
Coupling

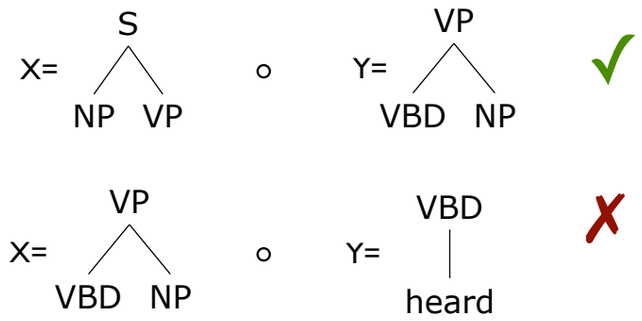
Given a grammar **G**, constraint set **C**:

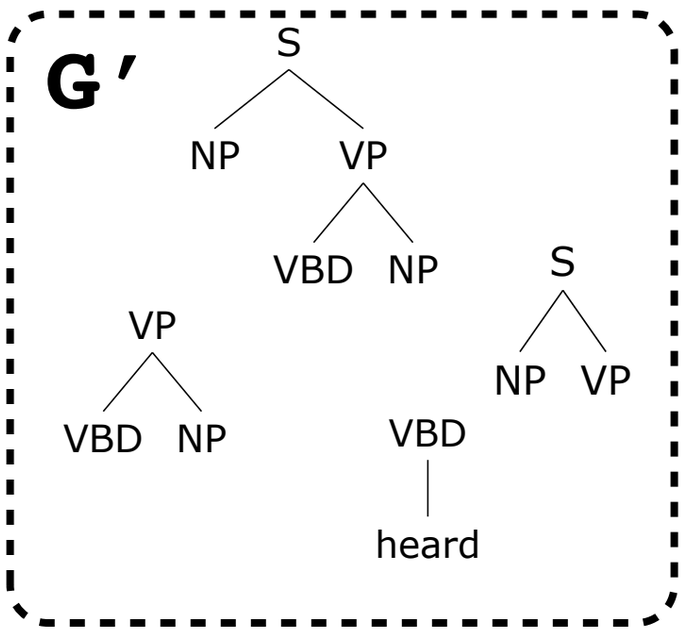
1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$









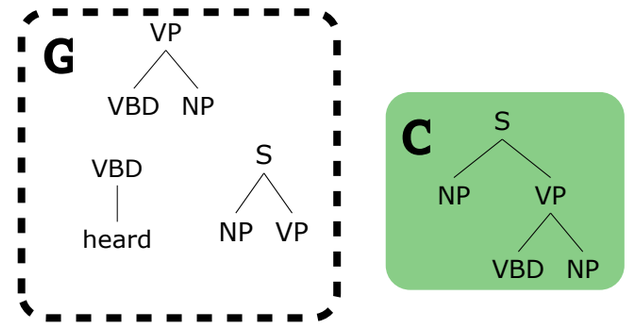
Coupling

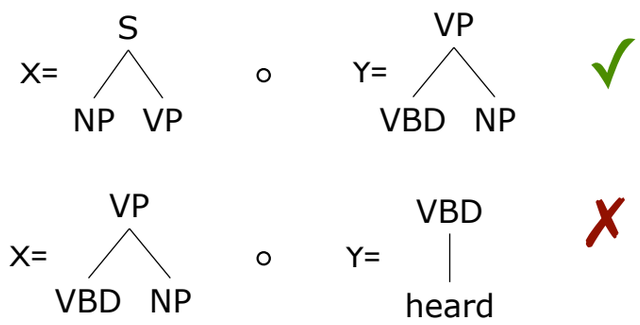
Given a grammar **G**, constraint set **C**:

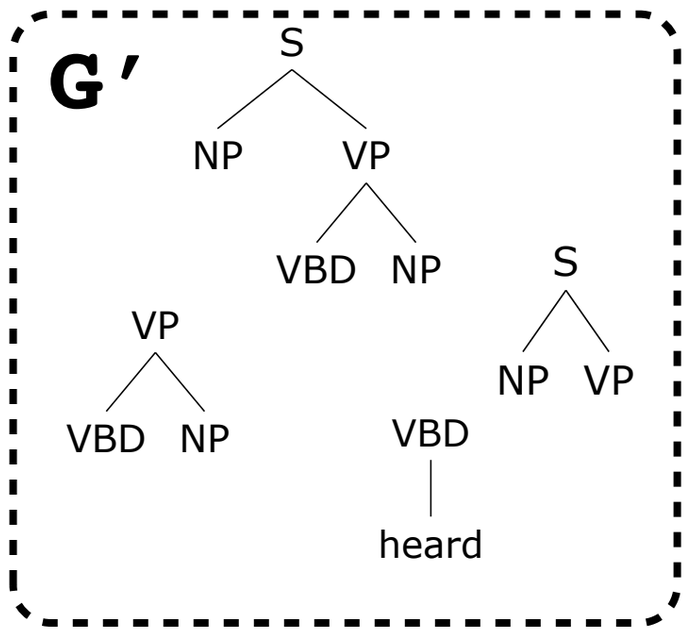
1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$

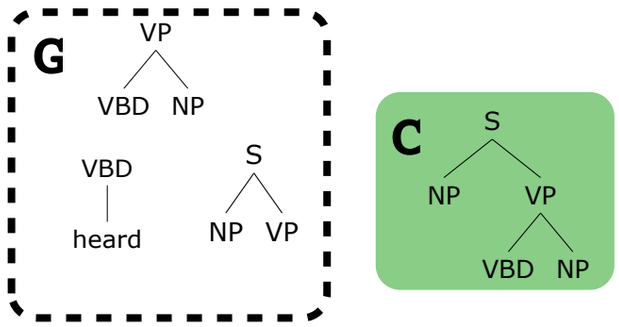
2. Estimate initial **G′** fragment weights







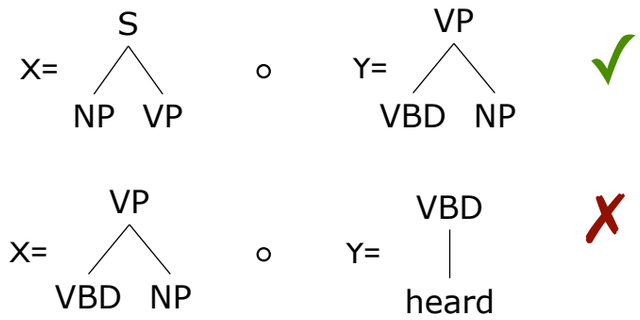
Coupling



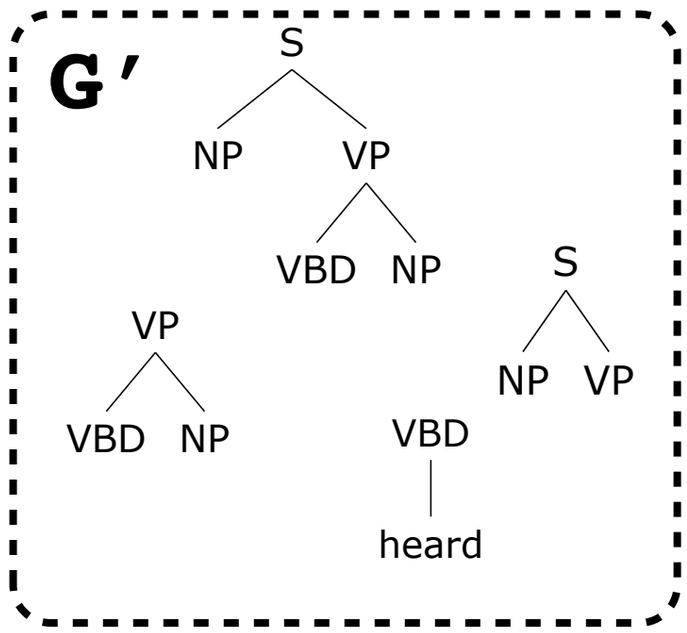
Given a grammar **G**, constraint set **C**:

1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$G' = G \cup \{X \circ Y \in C \mid X \in G\}$$



2. Estimate initial **G′** fragment weights
   - Uniform redistribution not appropriate



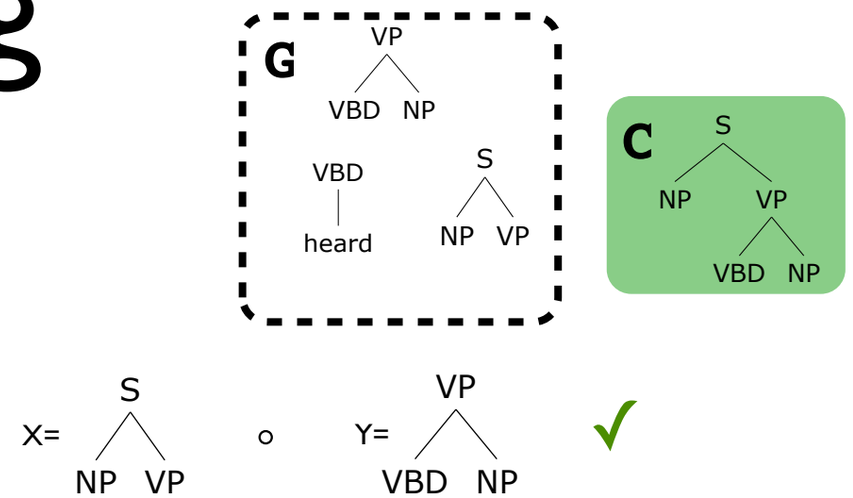
Coupling

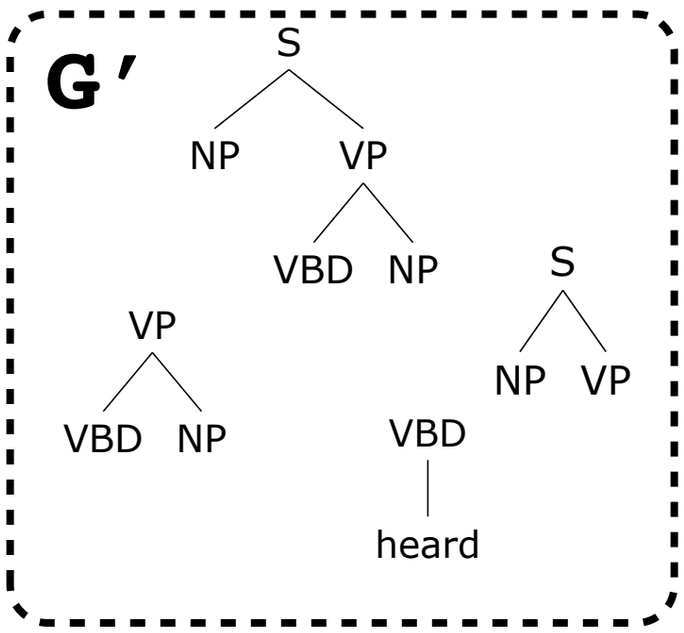
Given a grammar **G**, constraint set **C**:

1. Construct a grammar **G′** from **G** and allowed couplings from **C**

$$\mathbf{G'} = \mathbf{G} \cup \{X \circ Y \in \mathbf{C} \mid X \in \mathbf{G}\}$$

2. Estimate initial **G′** fragment weights
   - Uniform redistribution not appropriate
3. Fit weights of **G′** via inside/ outside





Deriving the Constraint Set

- Deterministically count compact TSG fragments
- Iteratively extract the *K* most frequent subtrees of size *R*
- *R*, *K* enforce sparsity and control what can and cannot couple

EXTRACTFRAGMENTS(R,K)

```
S ← ∅
F(1,K) ← top K CFG rules used
for r = 2 to R do
    S ← {F ∈ F(r-1, K), extended by 1 rule}
    F(r,K) ← top K elements of F(r-1, K) U S
end for
```

Qualitative Evaluation

Qualitative Evaluation

- Korean Treebank v2.0
- Sect. 2-3 of Penn Treebank (WSJ)

Qualitative Evaluation

- Korean Treebank v2.0
- Sect. 2-3 of Penn Treebank (WSJ)
- Petrov et al. (2011)’s universal tag set

Qualitative Evaluation

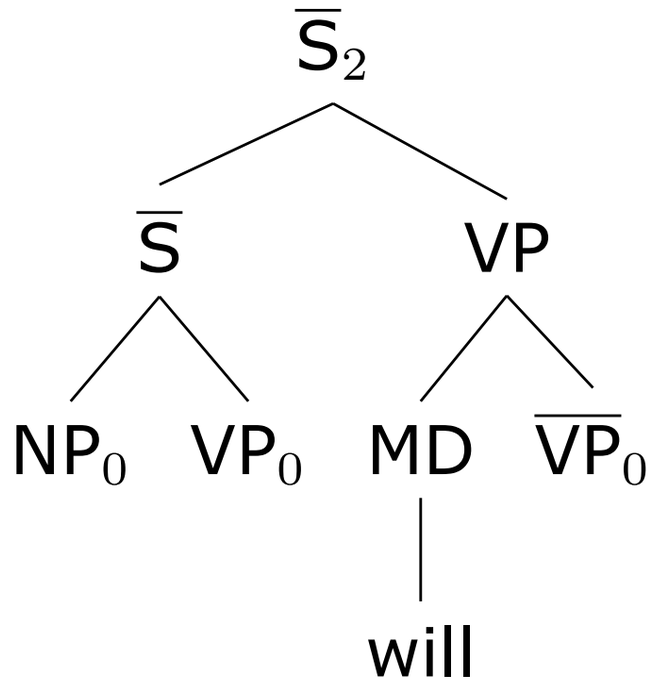
- Korean Treebank v2.0
- Sect. 2-3 of Penn Treebank (WSJ)
  - Petrov et al. (2011)'s universal tag set
  - Replace all preterminals with a single symbol, X.

Qualitative Evaluation

- Korean Treebank v2.0
- Sect. 2-3 of Penn Treebank (WSJ)
- Petrov et al. (2011)'s universal tag set
- Replace all preterminals with a single symbol, X.

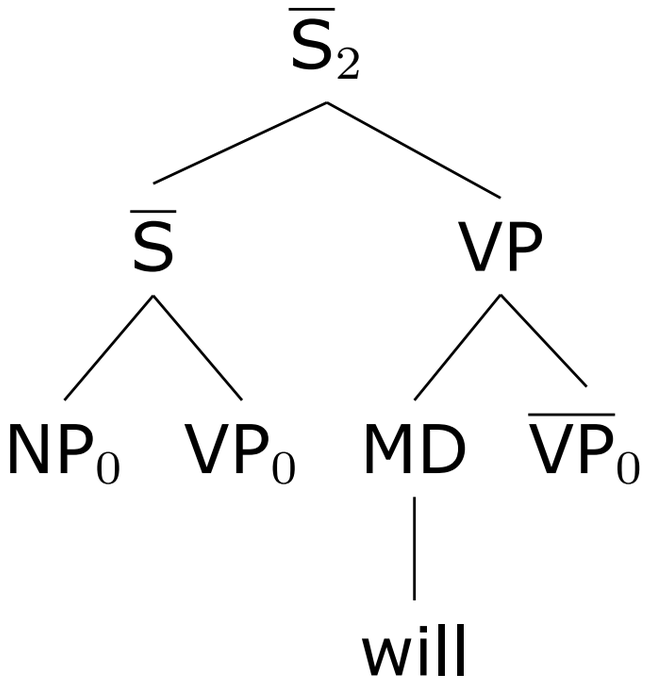
WSJ, §2-3

WSJ, §2-3

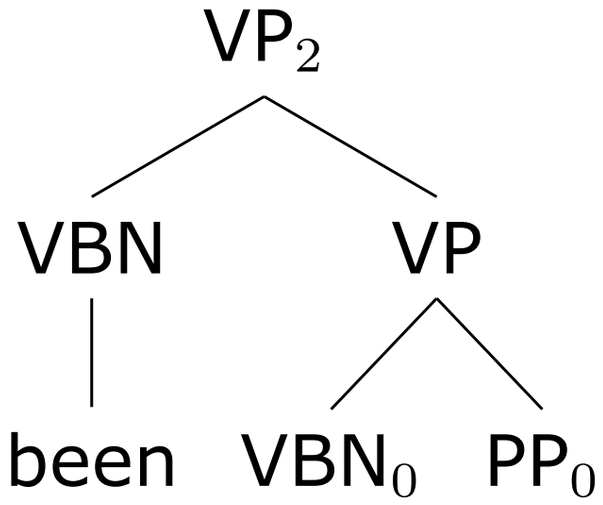


Modals

WSJ, §2-3

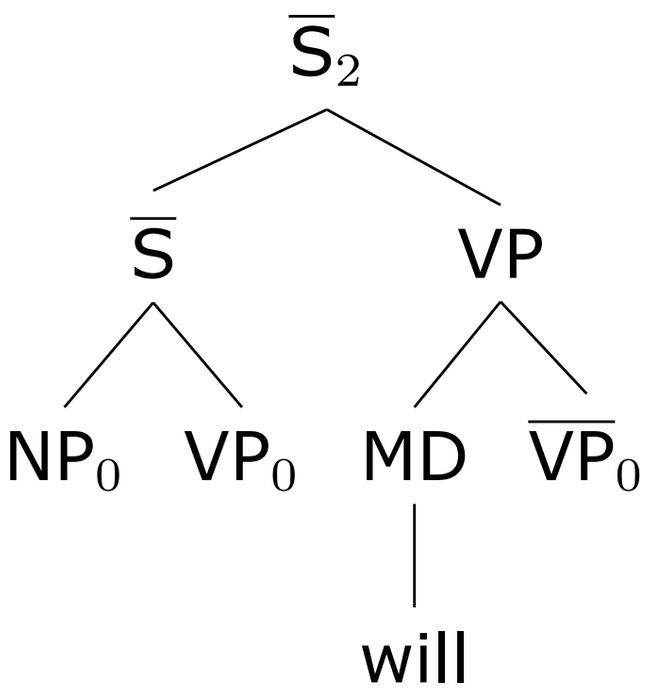


Modals

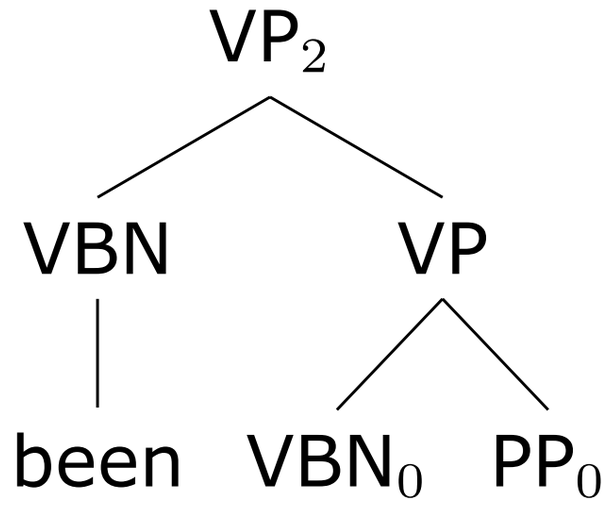


Perfectives

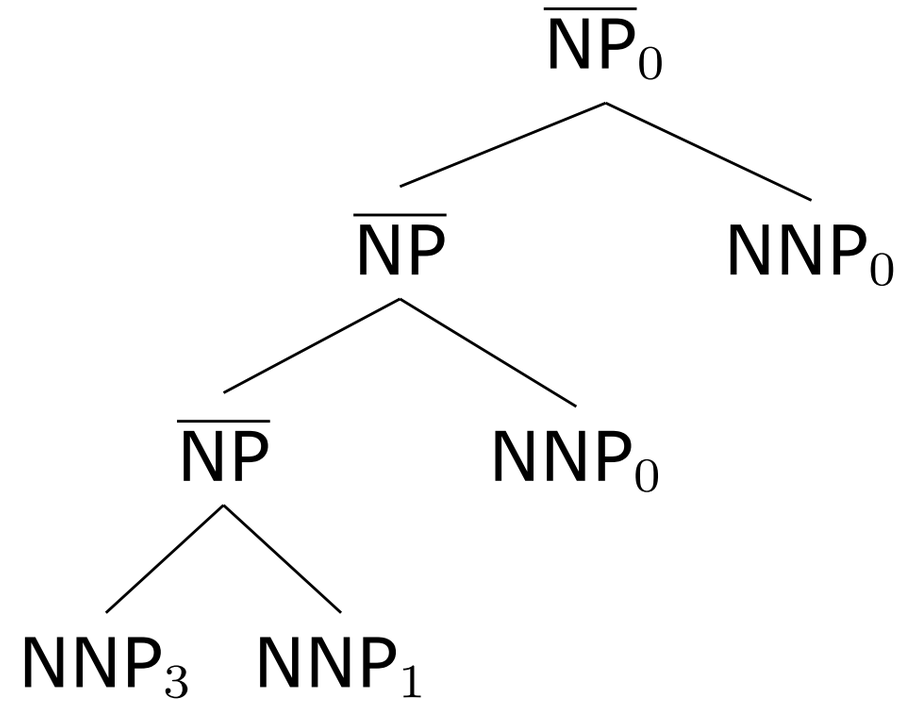
WSJ, §2-3



Modals



Perfectives



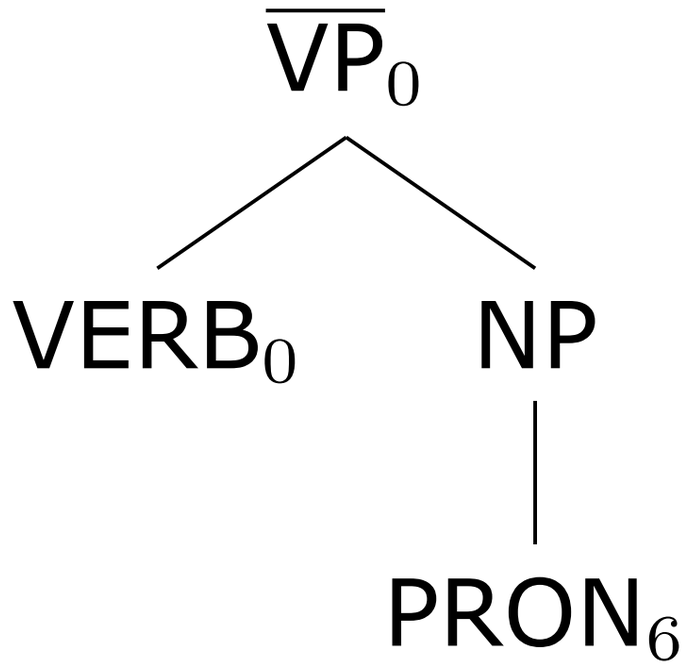
Nominals

Universal Tag Set, WSJ, §2-3

	PRON
1	its, his, your
5	it, he, they
6	it, them, him

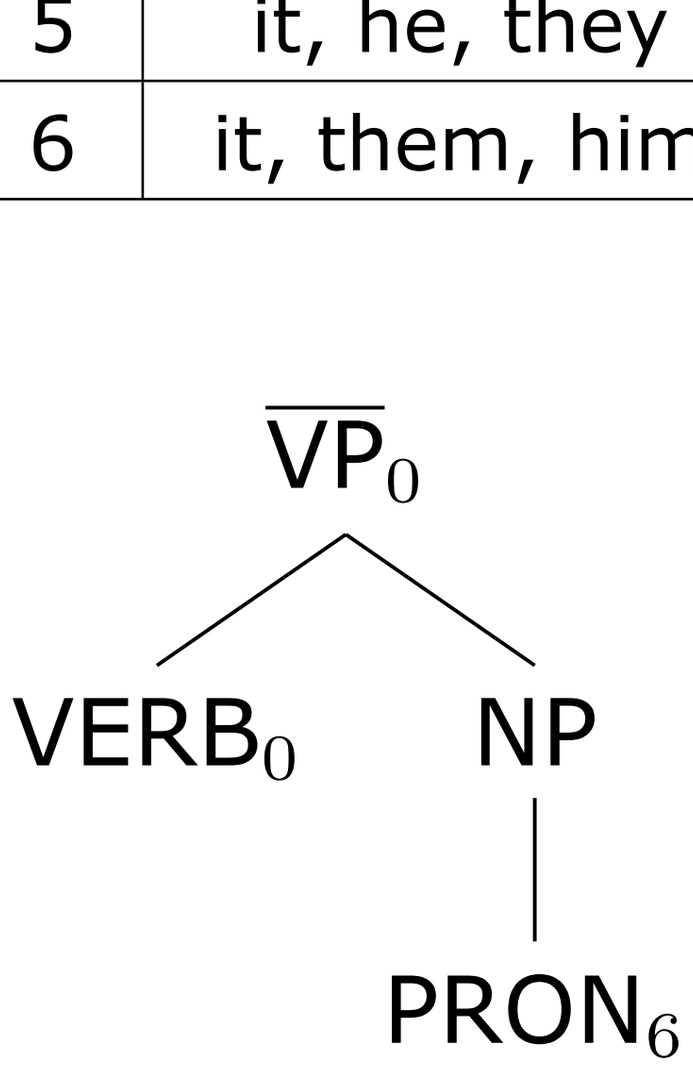
Universal Tag Set, WSJ, §2-3

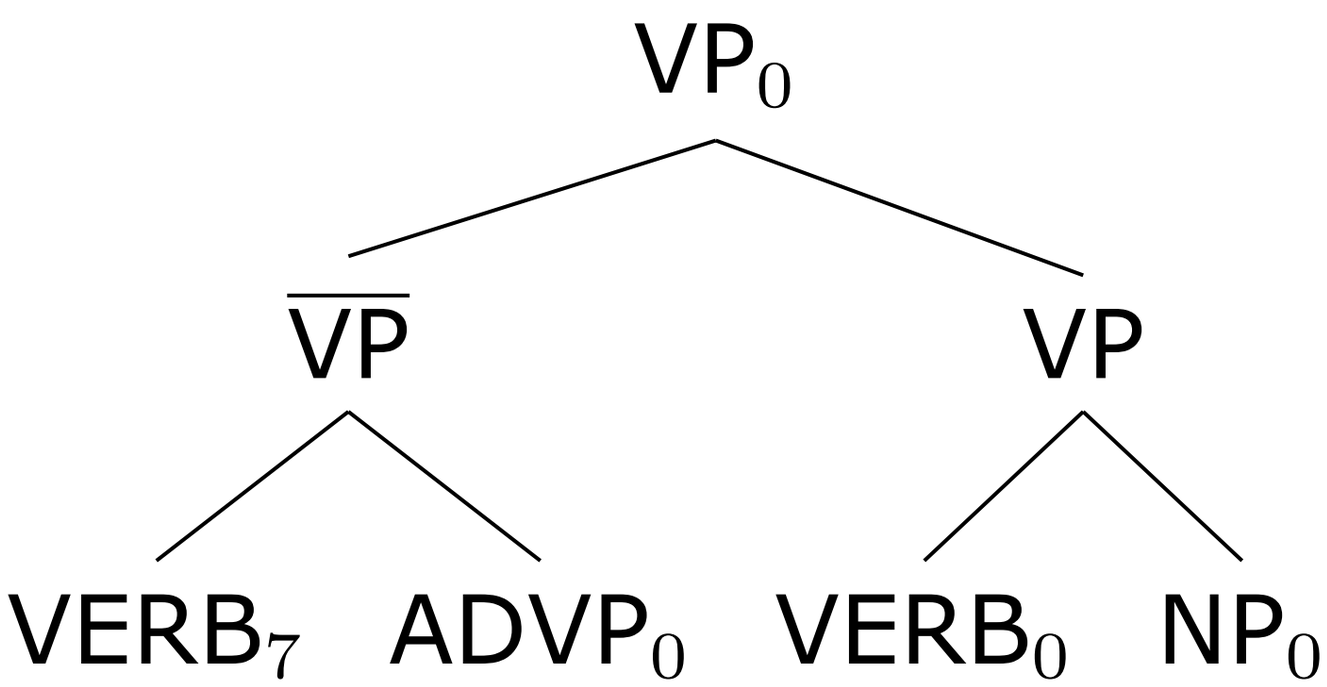
	PRON
1	its, his, your
5	it, he, they
6	it, them, him



Universal Tag Set, WSJ, §2-3

	PRON
1	its, his, your
5	it, he, they
6	it, them, him





Thank you!