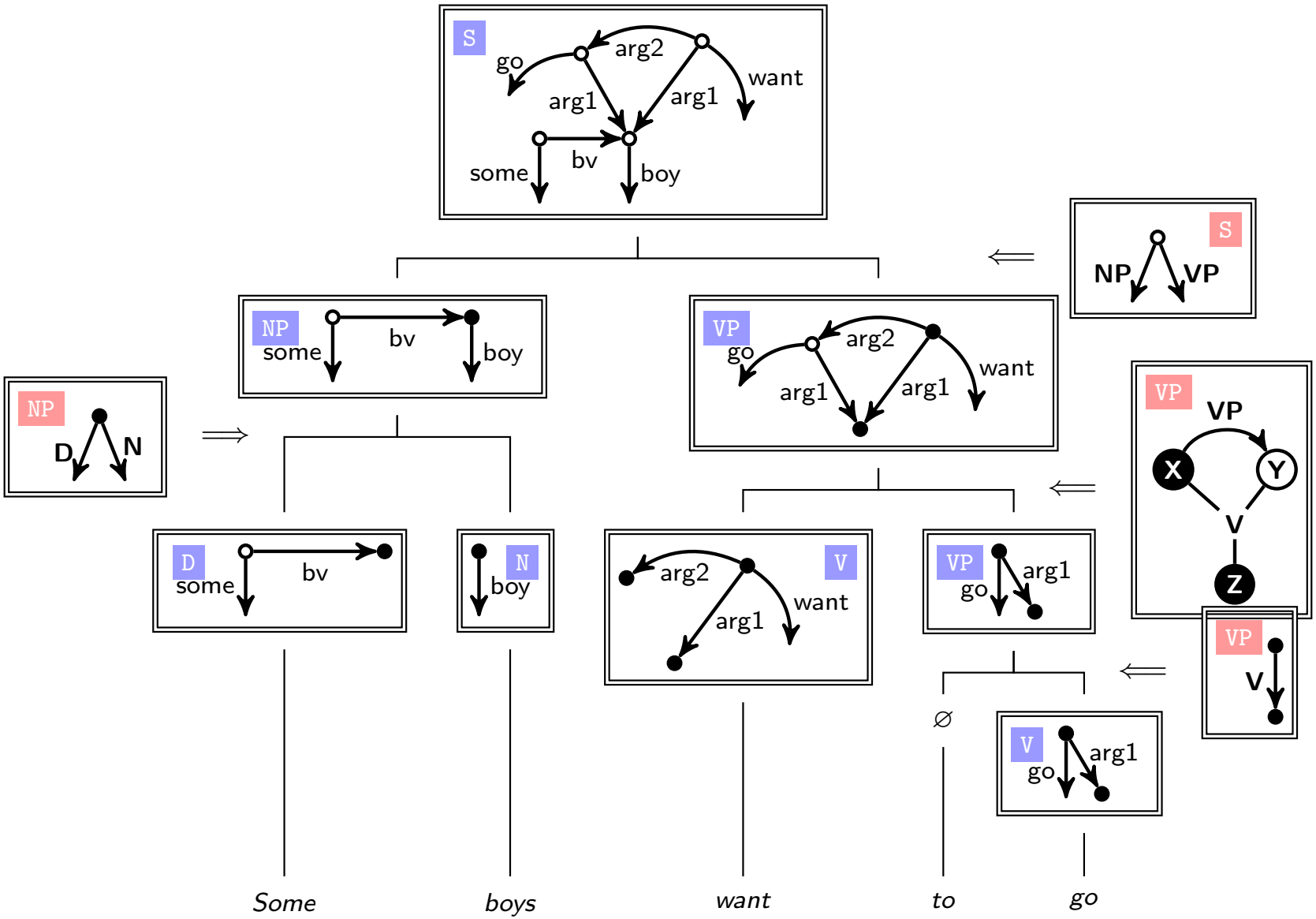
Lecture 8: Bidirectional Syntactico-Semantic Composition

Weiwei Sun

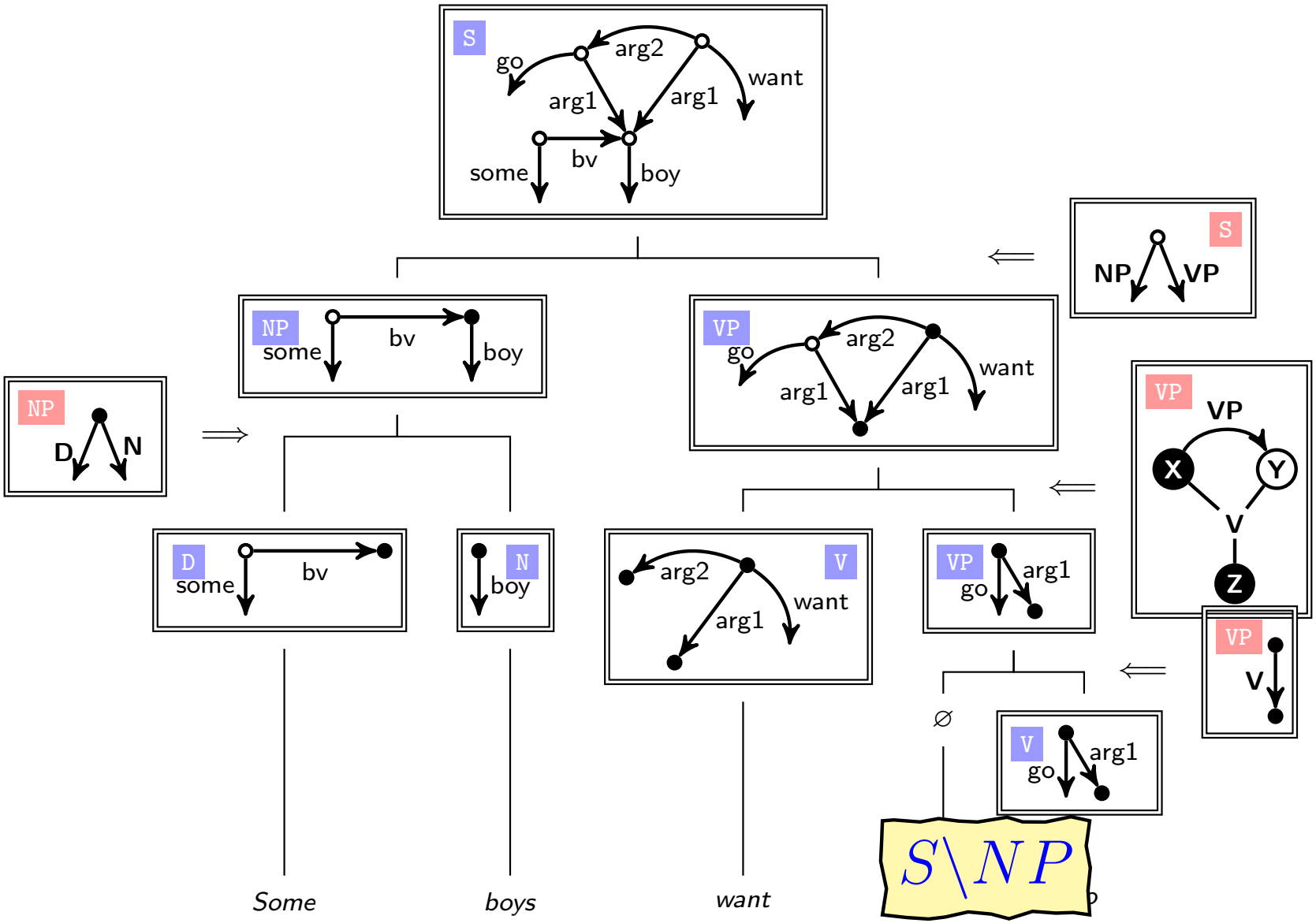
Department of Computer Science and Technology
University of Cambridge

Michaelmas 2024/25

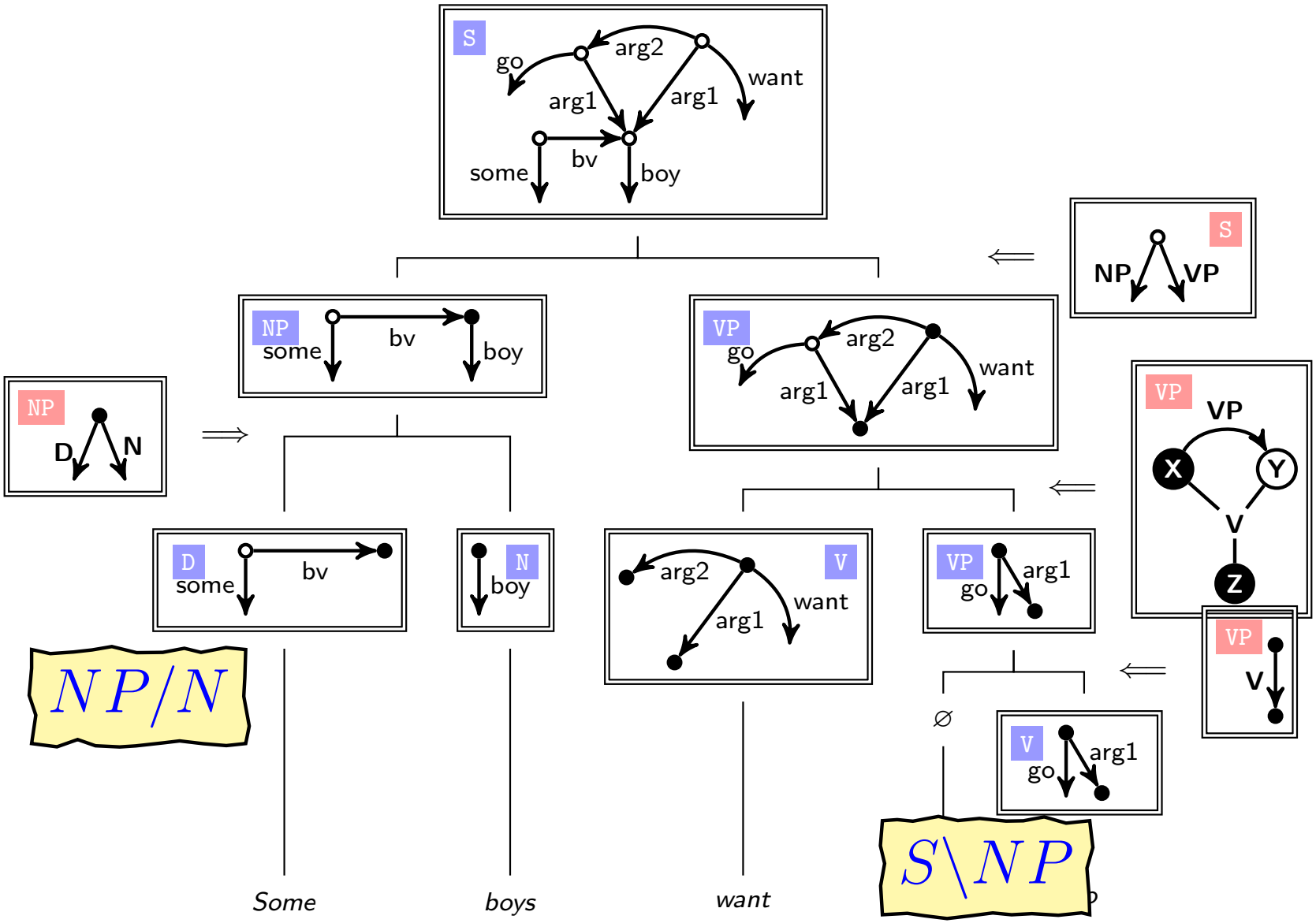
Lexicalised Grammar



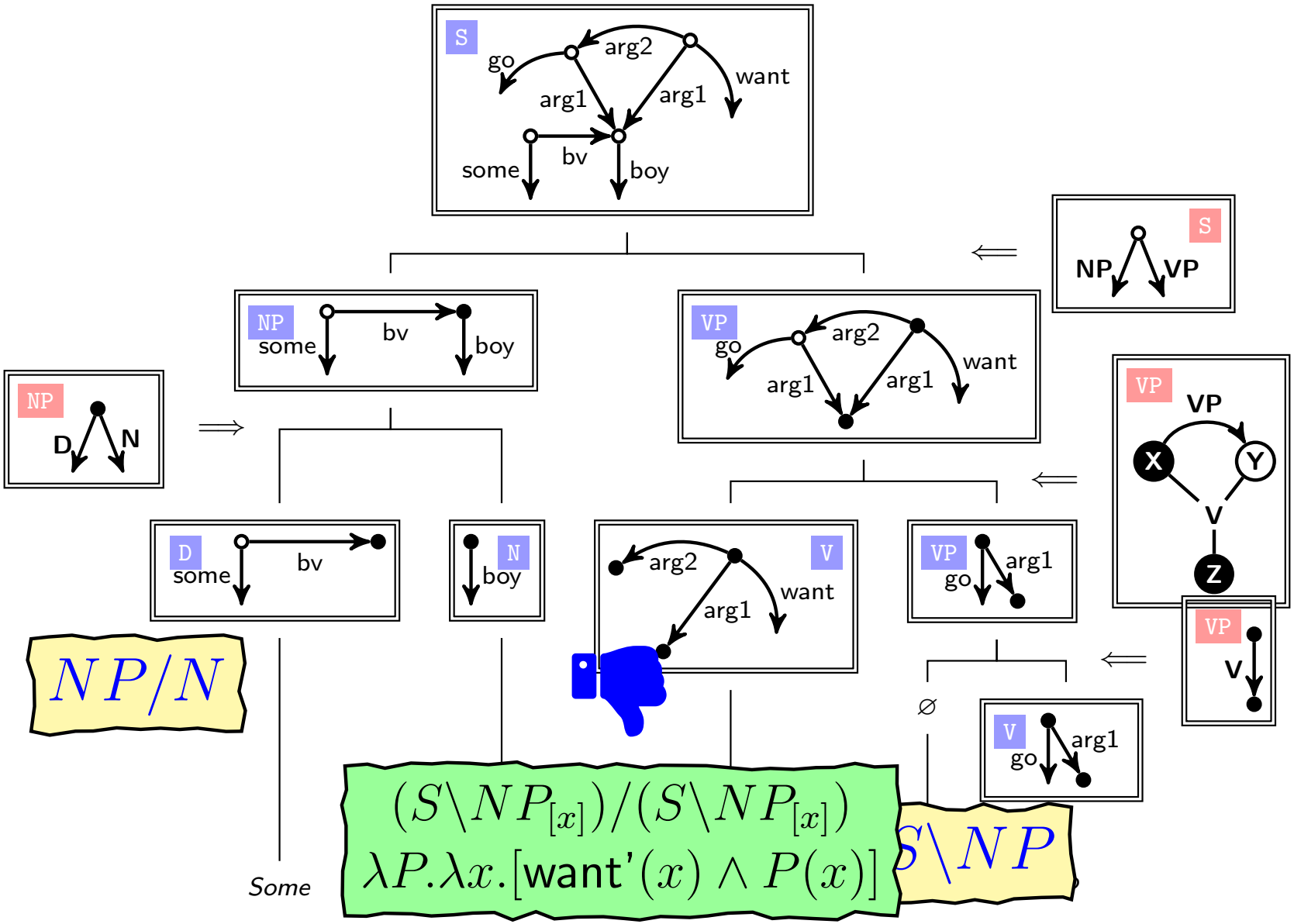
Lexicalised Grammar



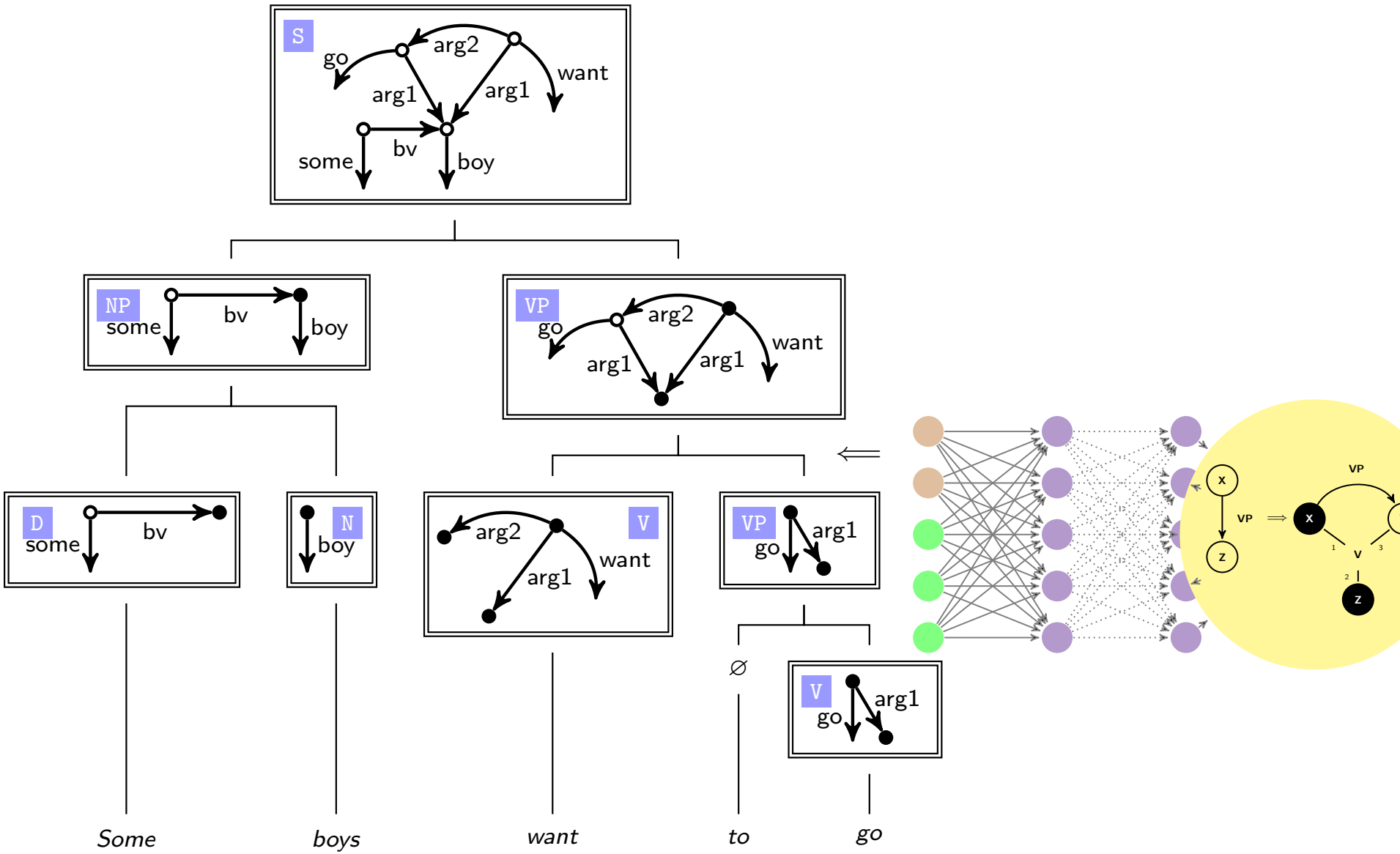
Lexicalised Grammar



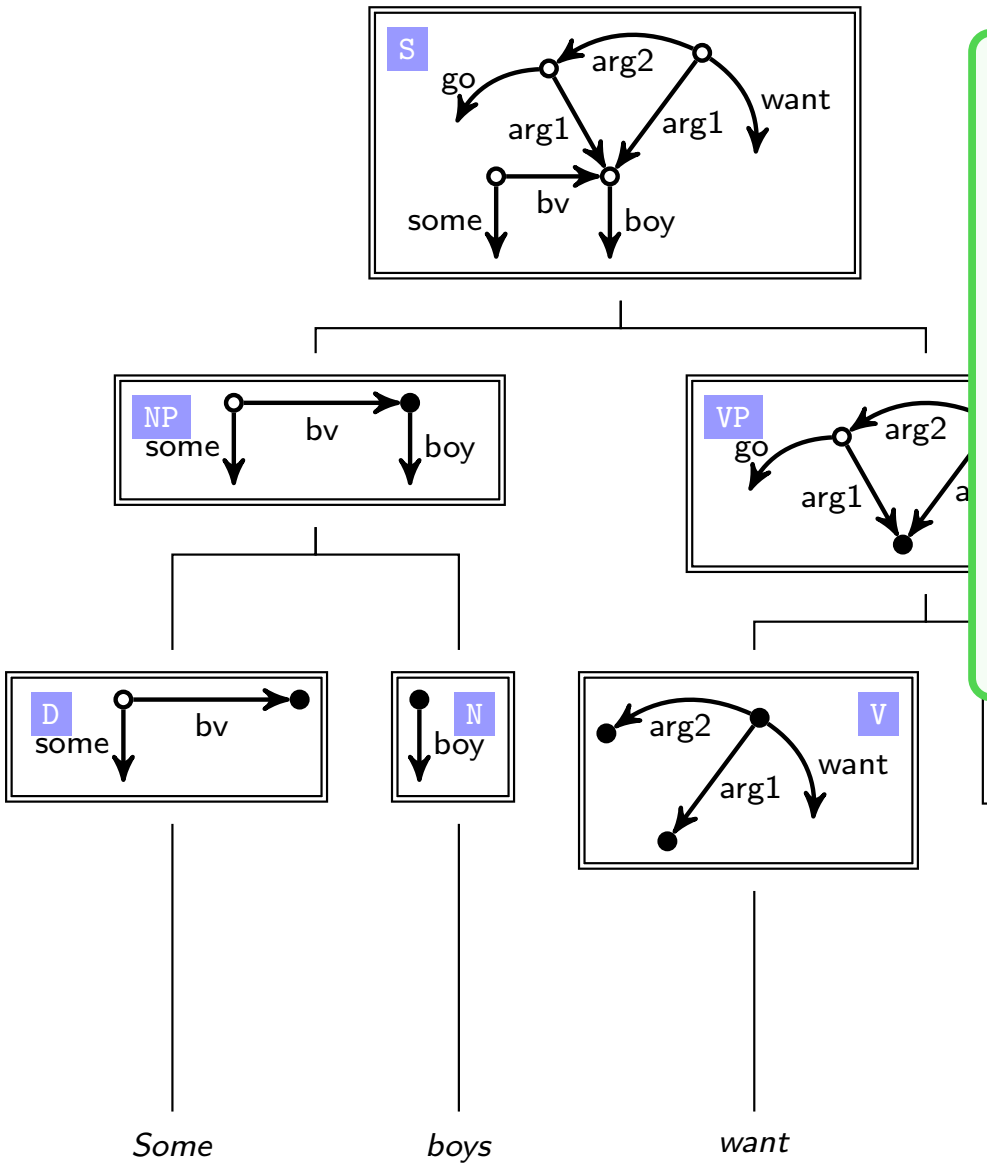
Lexicalised Grammar



Scoring a Derivation Tree



Scoring a Derivation Tree



enumerating trees

String-to-graph parsing:

$$\arg\max_{T \in \mathcal{T}(x)} \text{SCORE}(T)$$

Graph-to-string Parsing:

$$\arg\max_{T \in \mathcal{T}(G)} \text{SCORE}(T)$$

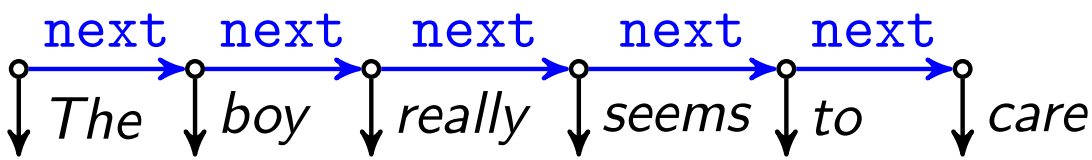
A Unified Parsing Problem

Semantic Graph Parsing

$$\arg \max_{T \in \mathcal{T}(x)} \textsc{score}(T)$$

Parsing Semantic Graphs

$$\arg \max_{T \in \mathcal{T}(G)} \textsc{score}(T)$$



Computing all possible / the best derivation(s) of a given graph

$$\max_{T \in \mathcal{T}(G)} \textsc{score}(T) = \max_{T \in \mathcal{T}(G)} \sum_{\gamma \text{ in } T} \textsc{score}(\gamma, G)$$

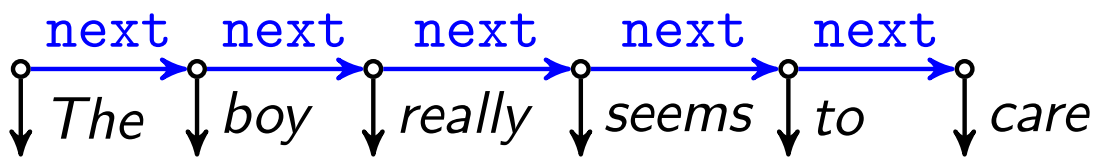
A Unified Parsing Problem

Semantic Graph Parsing

$$\arg \max_{T \in \mathcal{T}(x)} \textsc{score}(T)$$

Parsing Semantic Graphs

$$\arg \max_{T \in \mathcal{T}(G)} \textsc{score}(T)$$



Computing all possible / the best derivation(s) of a given graph

$$\max_{T \in \mathcal{T}(G)} \textsc{score}(T) = \max_{T \in \mathcal{T}(G)} \sum_{\gamma \text{ in } T} \textsc{score}(\gamma, G)$$

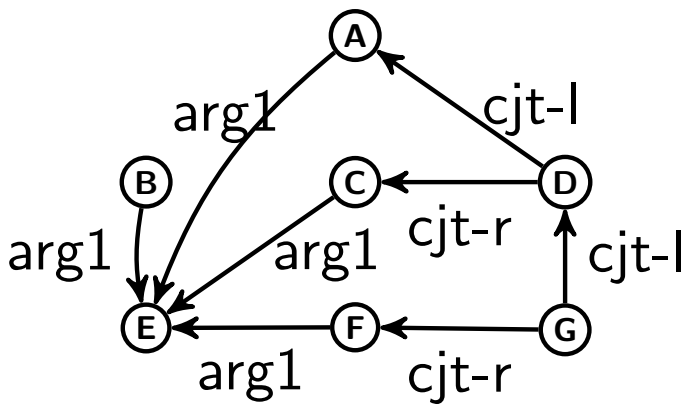
$$\bigoplus_{T \in \mathcal{T}(G)} \left(\bigotimes_{\gamma \text{ in } T} \textsc{score}(\gamma, G) \right)$$

	Inside	Viterbi
$\oplus$	$a + b$	$\max(a, b)$
$\otimes$	$a \times b$	$a \times b$

Semiring parsing

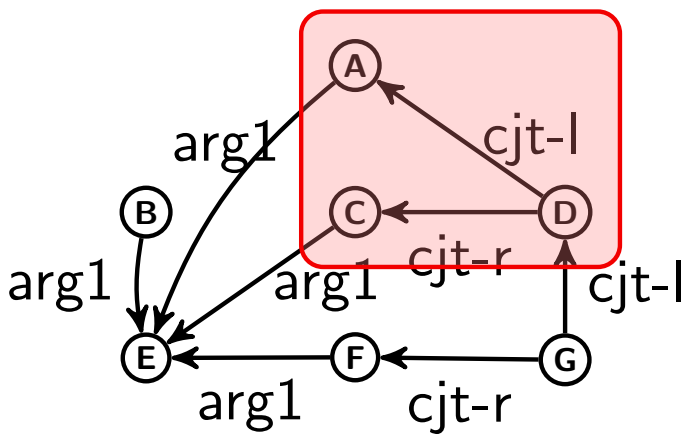
A Unified Parsing Problem; A Unified Parsing Algorithm

A dynamic programming algorithm



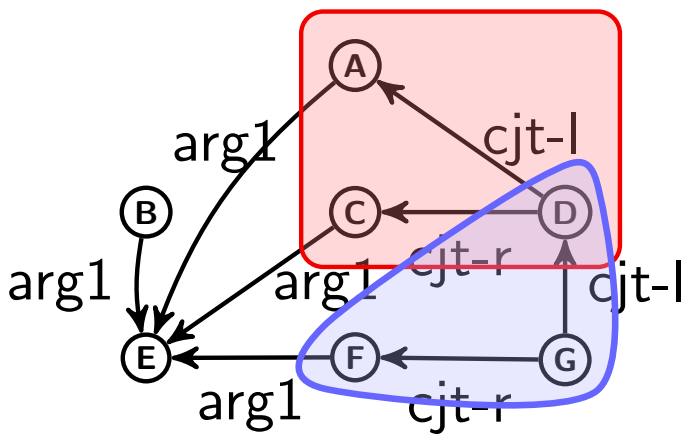
A Unified Parsing Problem; A Unified Parsing Algorithm

A dynamic programming algorithm



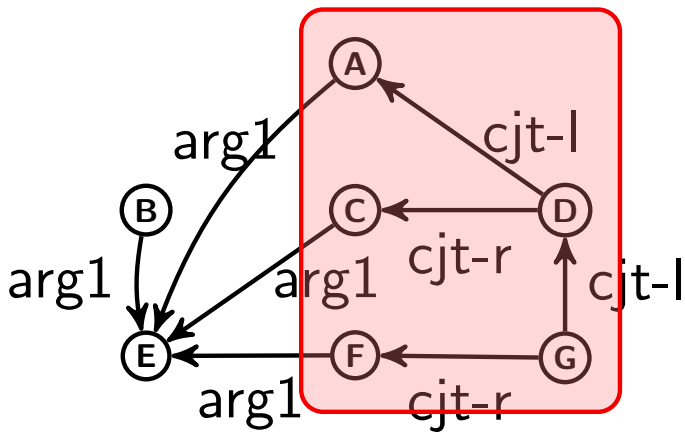
A Unified Parsing Problem; A Unified Parsing Algorithm

A dynamic programming algorithm



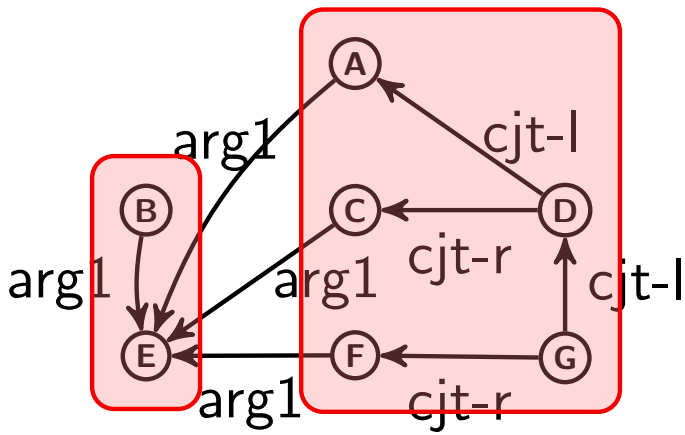
A Unified Parsing Problem; A Unified Parsing Algorithm

A dynamic programming algorithm



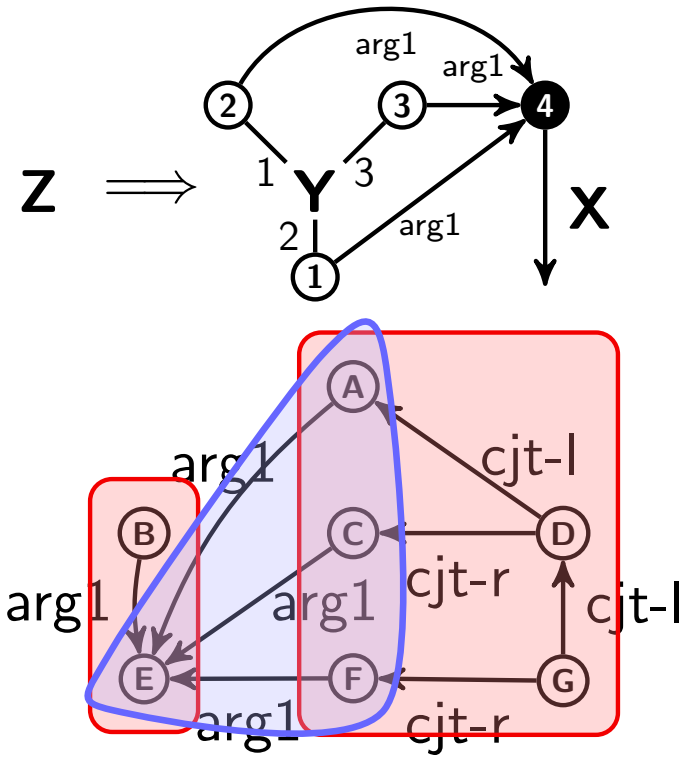
A Unified Parsing Problem; A Unified Parsing Algorithm

A dynamic programming algorithm



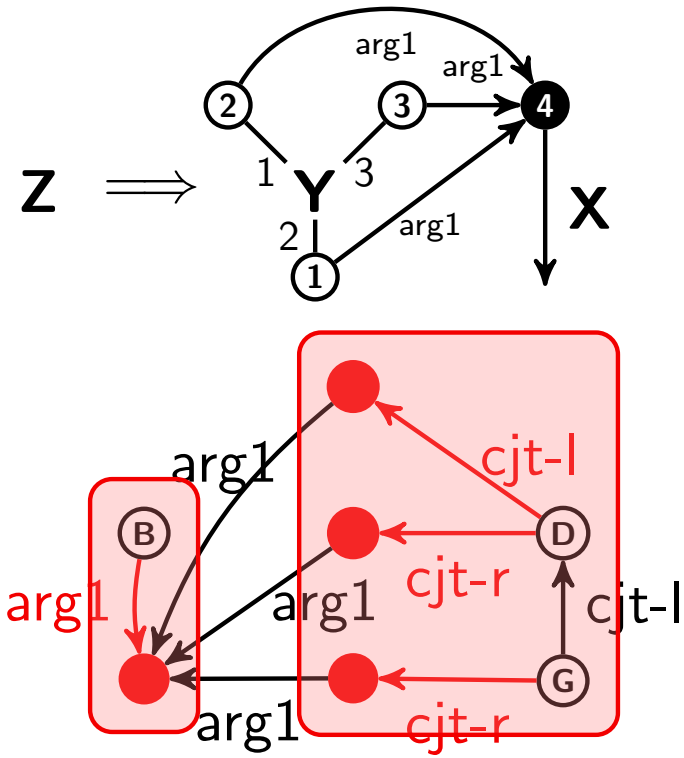
A Unified Parsing Problem; A Unified Parsing Algorithm

A dynamic programming algorithm



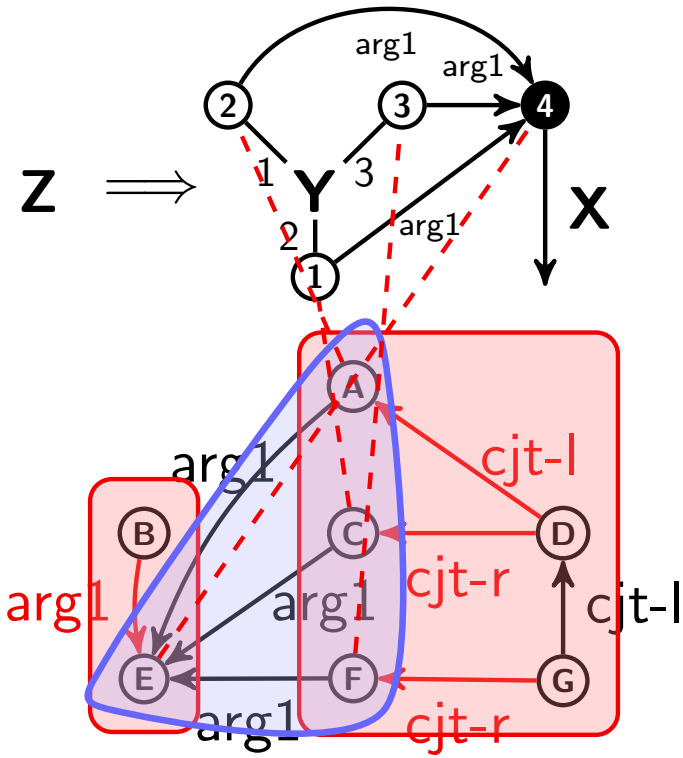
A Unified Parsing Problem; A Unified Parsing Algorithm

A dynamic programming algorithm



A Unified Parsing Problem; A Unified Parsing Algorithm

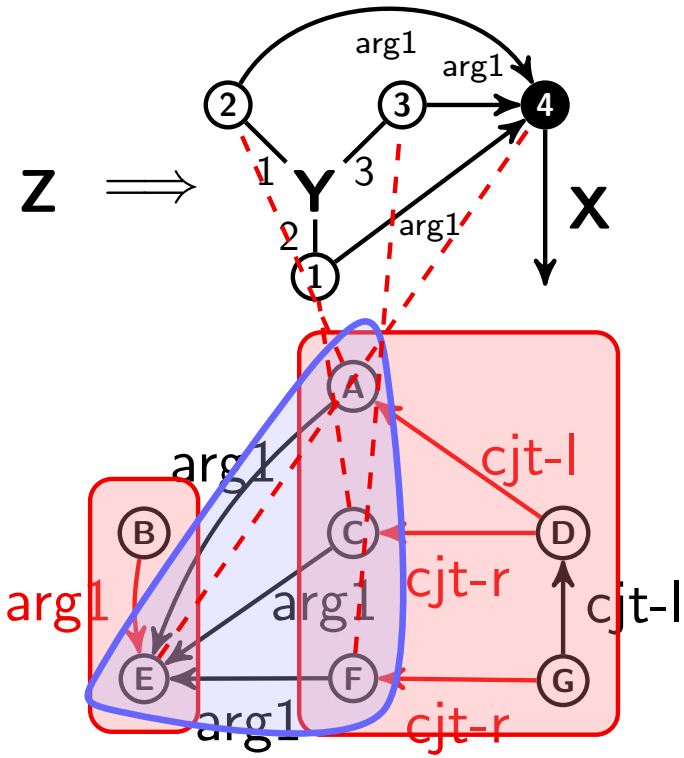
A dynamic programming algorithm

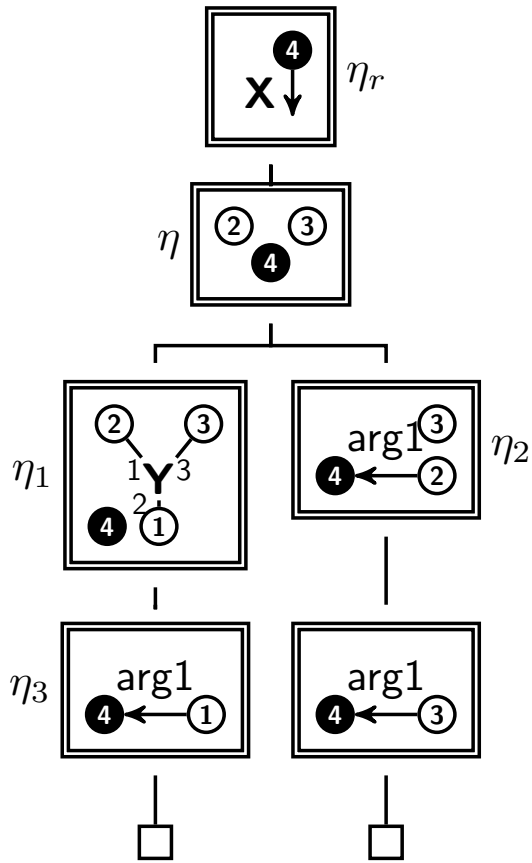


A Unified Parsing Problem; A Unified Parsing Algorithm

A dynamic programming algorithm

Binarizing rules via tree decomposition

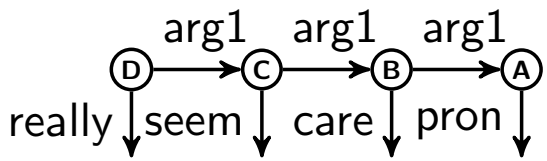




Ambiguity: Building the Derivation Forest

Semantic Graph Parsing

$$\arg \max_{T \in \mathcal{T}(x)} \textsc{score}(T)$$



Parsing Semantic Graphs

$$\arg \max_{T \in \mathcal{T}(G)} \textsc{score}(T)$$

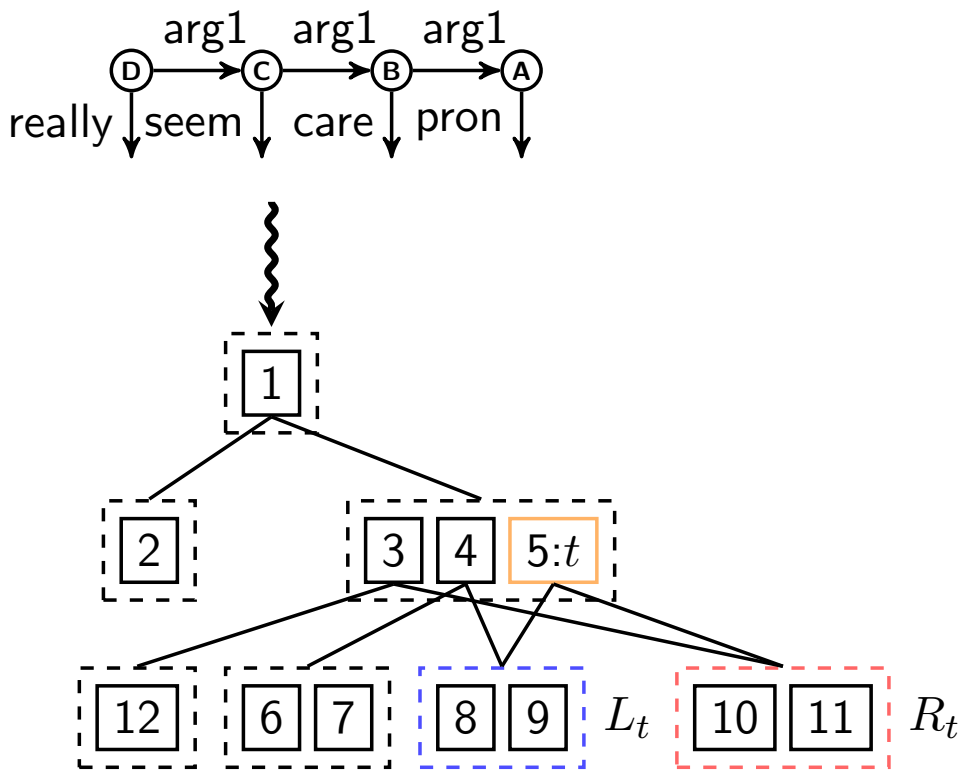
Ambiguity: Building the Derivation Forest

Semantic Graph Parsing

$$\arg \max_{T \in \mathcal{T}(x)} \textsc{score}(T)$$

Parsing Semantic Graphs

$$\arg \max_{T \in \mathcal{T}(G)} \textsc{score}(T)$$



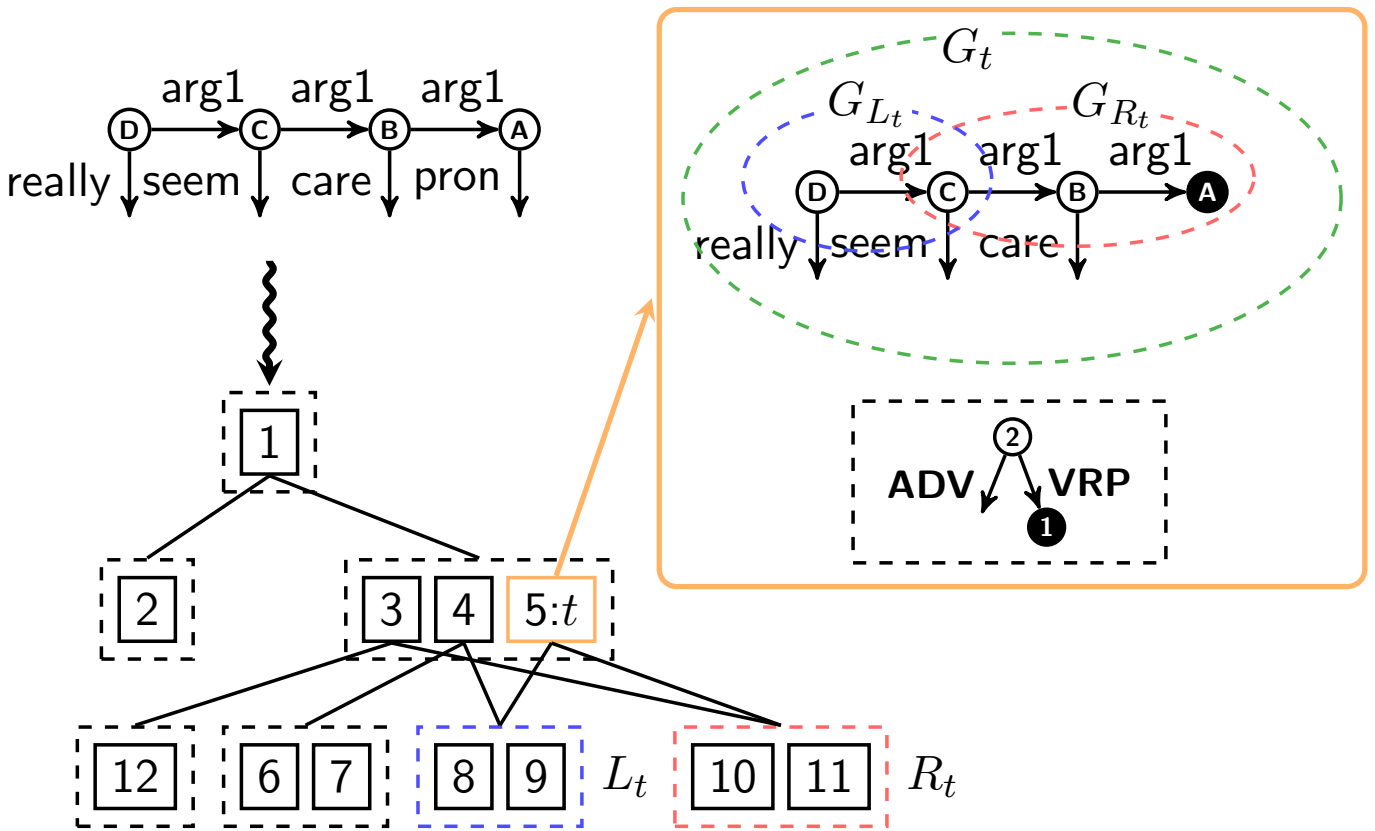
Ambiguity: Building the Derivation Forest

Semantic Graph Parsing

$$\arg \max_{T \in \mathcal{T}(x)} \textsc{score}(T)$$

Parsing Semantic Graphs

$$\arg \max_{T \in \mathcal{T}(G)} \textsc{score}(T)$$



Real-World Grammar; Real Running Time

A wide-coverage linguistically-meaningful grammar is indiced by applying an automatic grammar extraction algorithm. The grammar is lexicalized (LxG), in that argument-structures are lexically encoded.

LxG	**#Rule**		**Treewidth**	**#Node**	**#Terminal**
Lexical	46,101	avg.	1.07	2.15	2.47
		max.	4	10	18
Phrasal	8,594	avg.	1.62	2.94	0.79
		max.	6	7	10

The time for parsing graphs is highly depended on the implementation and hardware. We report two exact numbers:

- the number of successful (**#Succ**) integrations for chart items.
- the number of total (=successful+failed, **#Total**) integrations.

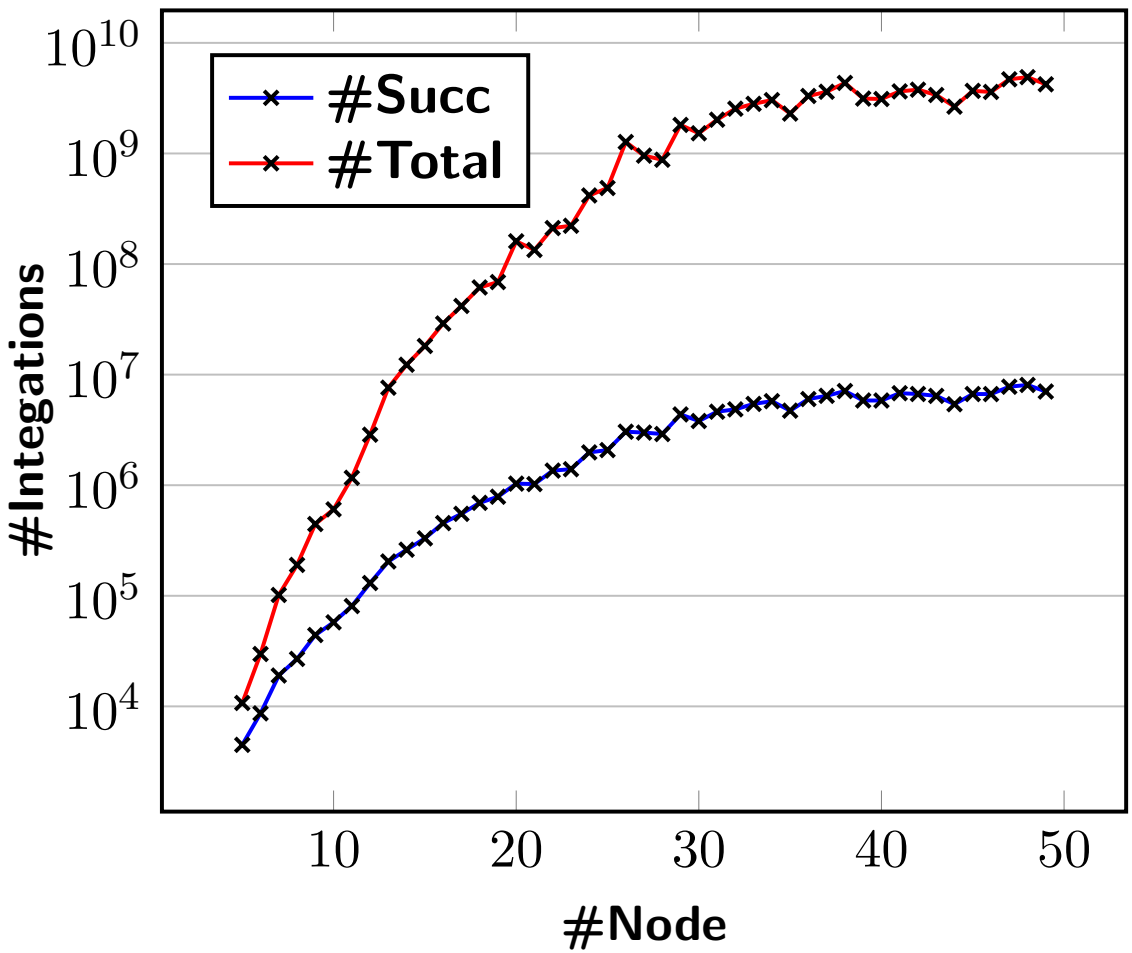
Evaluation with a Realistic Grammar

We test the efficiency of the algorithm on 4500 EDS graphs randomly selected from DeepBank 1.1 with the size in the range of 5 to 50.

#Node	**Time**(s)	**#Succ/#Total**	**OOM**	**#Graph**
All	21.64	0.21%	305	4500
<10	0.02	12.55%	0	500
10∼20	0.45	1.42%	0	1000
20∼30	9.36	0.34%	4	1000
30∼50	47.68	0.19%	301	2000

Evaluation with a Realistic Grammar

We test the efficiency of the algorithm on 4500 EDS graphs randomly selected from DeepBank 1.1 with the size in the range of 5 to 50.



Locality and the Constructivist Hypothesis

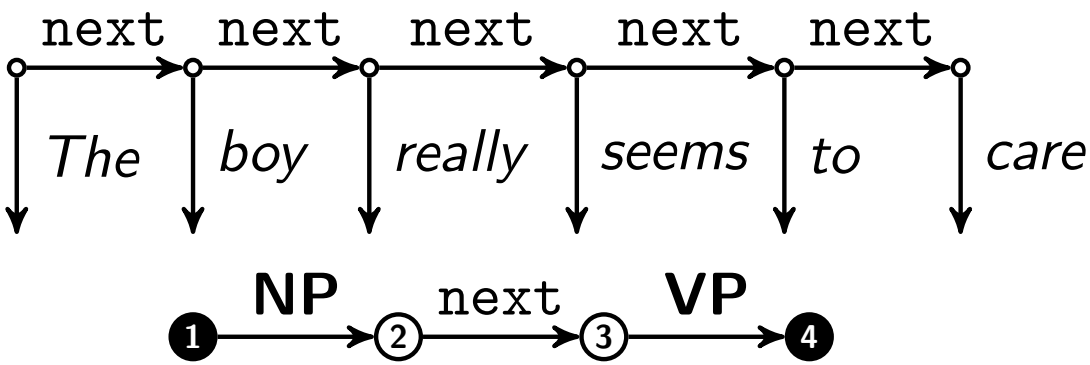
Locality as Terminal Edge-Adjacency

The Principle of Adjacency

Combinatory rules may only apply to finitely many phonologically realized and string-adjacent entities.

— *M. Steedman*

A graph-based view of string-adjacency



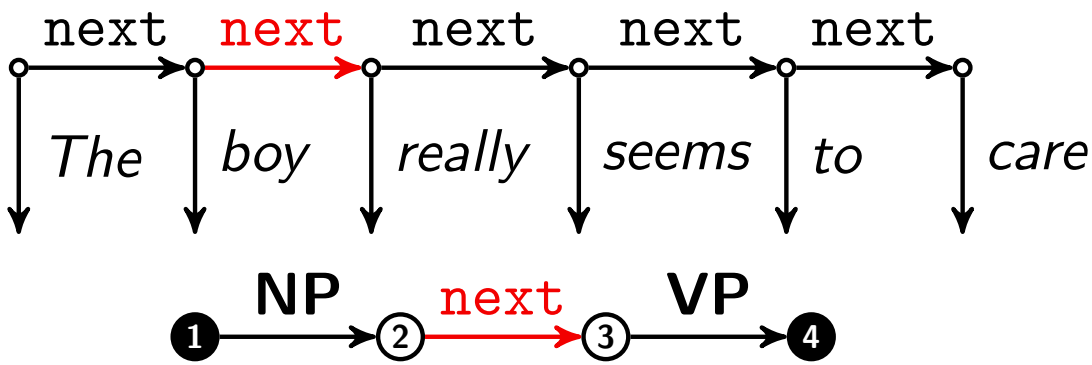
Locality as Terminal Edge-Adjacency

The Principle of Adjacency

Combinatory rules may only apply to finitely many phonologically realized and string-adjacent entities.

— *M. Steedman*

A graph-based view of string-adjacency



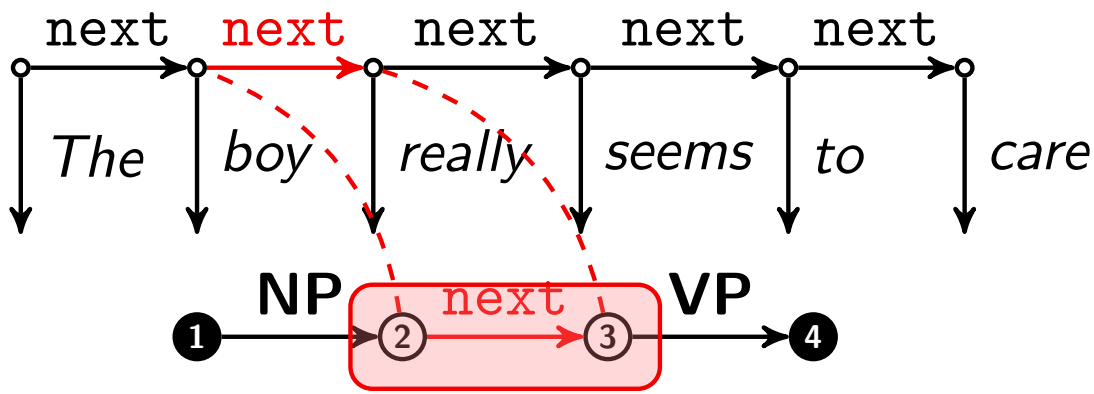
Locality as Terminal Edge-Adjacency

The Principle of Adjacency

Combinatory rules may only apply to finitely many phonologically realized and string-adjacent entities.

— *M. Steedman*

A graph-based view of string-adjacency

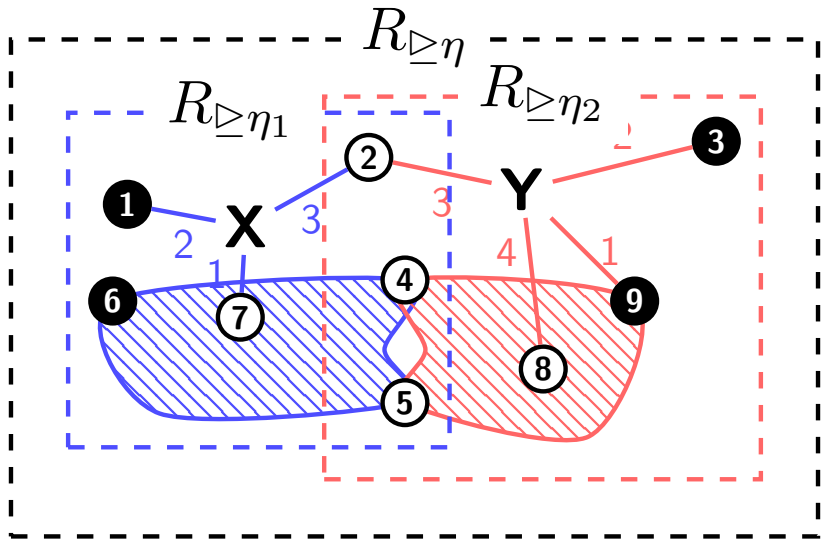


co-recognizable

A Locality-centric Complexity Analysis (1)

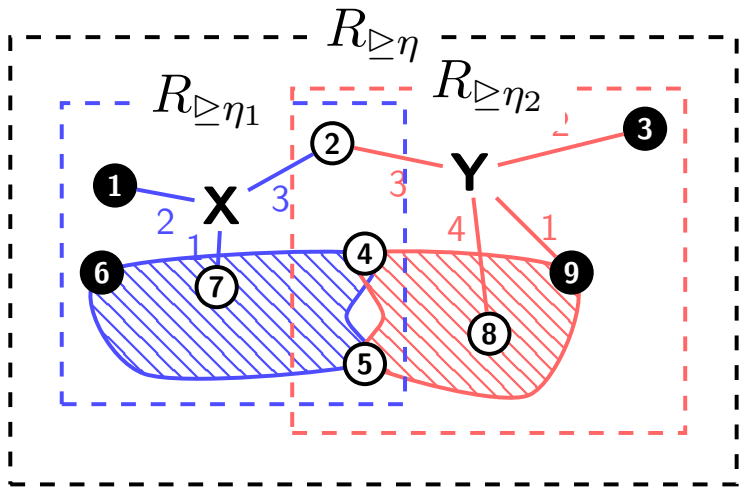
In a to-be-recognized subgraph which consists of only terminal edges, if one node is identified in an input graph, the possible positions of the other nodes are highly restricted.

Example



Shadowed areas mean subgraphs which consists of only terminal edges. If ④ is identified, then the cost to recognize ⑤ ❻ ❾ is highly restricted.

A Locality-centric Complexity Analysis (2)



Looking for a maximal subset of nodes such that all nodes in this subset is not terminal edge–adjacent to each other.

$$\{② ❸ ❶ ❻\}$$

Notation: δ

A Locality-centric Complexity Analysis (3)

The time complexity of the algorithm

- treewidth-centric: $\mathcal{O}((3^d n)^{k+1})$
- locality-centric: $\mathcal{O}(n^{\delta^*} d^{m_g} 3^{d n_g})$

Here $k/\delta^*/n_g/m_g$ is related to the grammar and d is the maximum degree of any node in the input graph.

It is provable that $\delta^* \leq k+1$. Empirical results:

		#Rule		$k+1$	δ
LxG	Phrasal	8,594	avg. max.	2.62 7	2.51 7

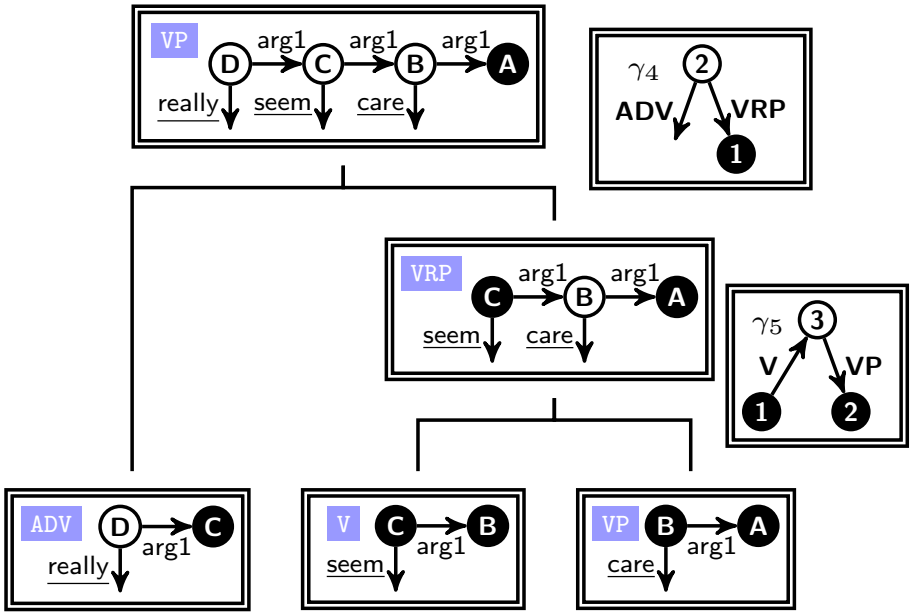
Computational Linguistics and Linguistics

Work in computational linguistics is in some cases motivated from a scientific perspective in that one is trying to provide a computational explanation for a particular linguistic or psycholinguistic phenomenon; and in other cases the motivation may be more purely technological in that one wants to provide a working component of a speech or natural language system.

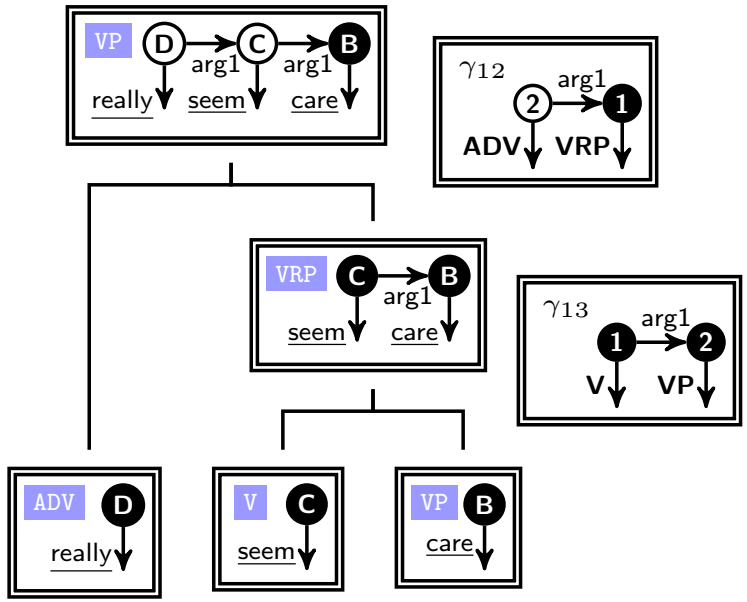
www.aclweb.org

Distributed Argument-Structure (1)

Lexicalized Grammar



Construction Grammar

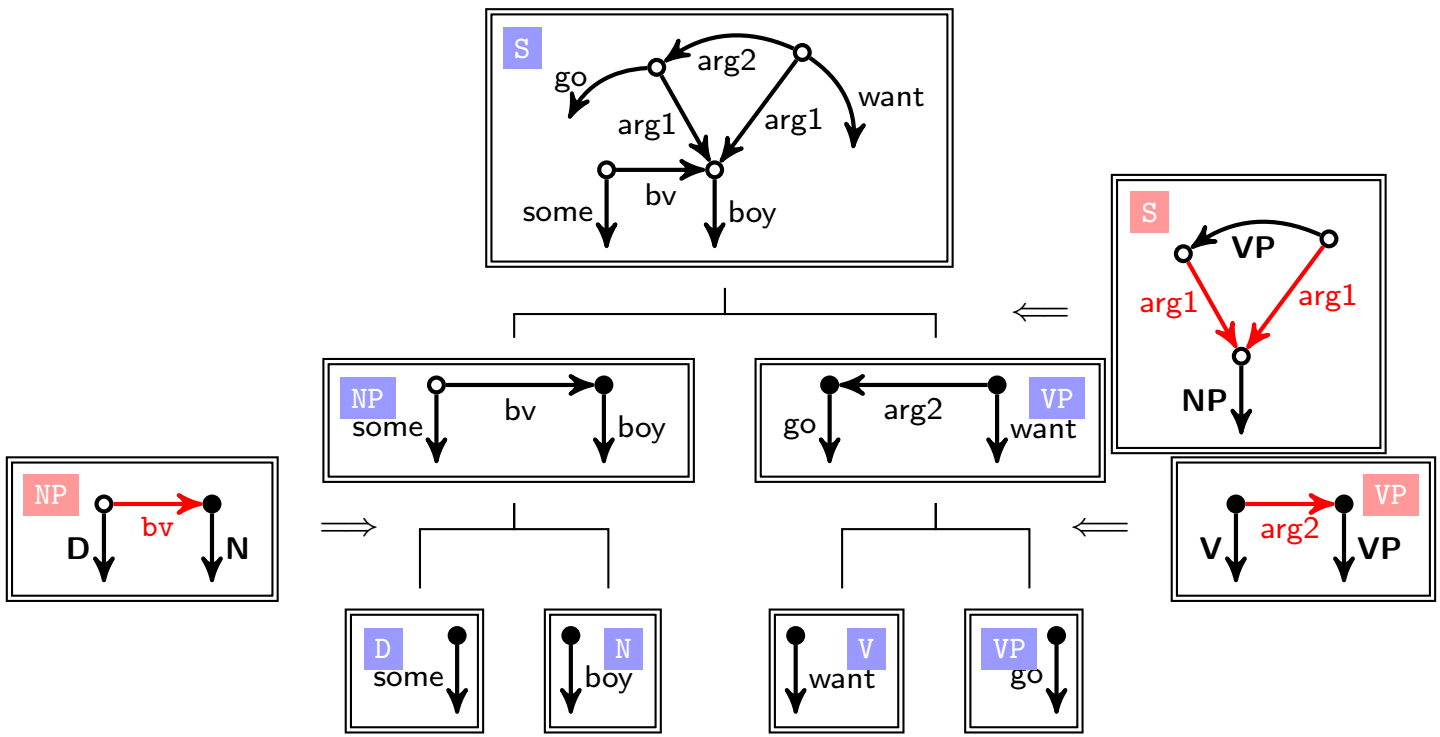


Lexicalism vs. Constructivism

- Lexicalist approaches were dominant in theoretical linguistics.
- Lexicalist approaches are dominant in computational linguistics: HPSG, LFG, CCG, ...
- HRG-based graph parsing ☺ the constructivist hypothesis.
- Roughly speaking, we lexicalise concepts and constructionalise relations of concepts.

Distributed Argument-Structure (2)

- **Lexicalist approach**
  Lexical rules try to use up all terminal edges at the initial stage of semantic composition.
- **Constructivist approach**
  By distributing terminal edges to all rules, both lexical and phrasal, δ is reduced on average.

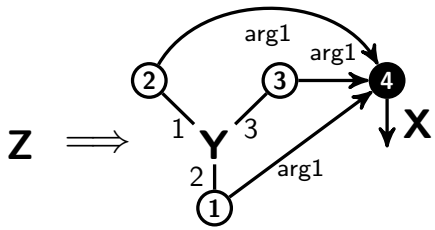


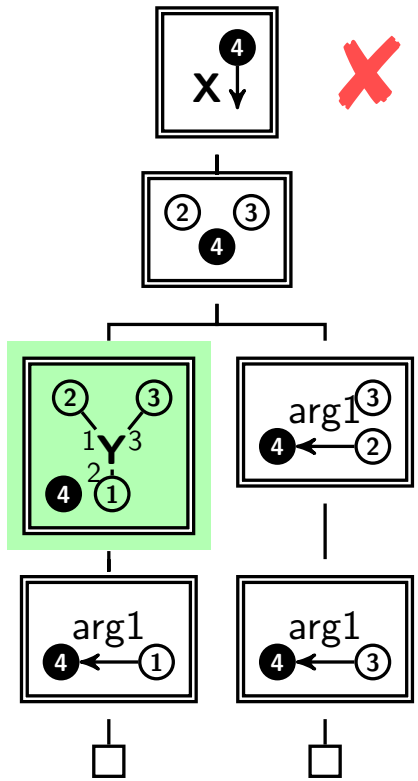
Distributed Argument-Structure (2)

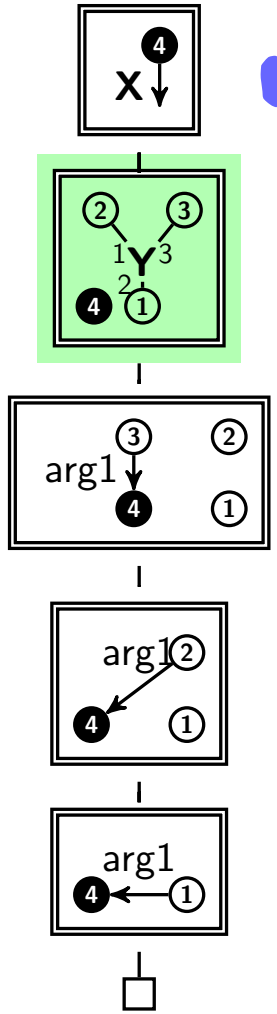
- **Lexicalist approach**
  Lexical rules try to use up all terminal edges at the initial stage of semantic composition.
- **Constructivist approach**
  By distributing terminal edges to all rules, both lexical and phrasal, δ is reduced on average.

		#Rule		$k+1$	δ
CxG	Lexical	34,348	avg.	1.36	—
			max.	5	—
	Phrasal	7,978	avg.	2.68	**1.59**
			max.	8	7
LxG	Lexical	46,101	avg.	2.07	—
			max.	5	—
	Phrasal	8,594	avg.	2.62	2.51
			max.	7	7

Terminal-First Strategy for Node Matching



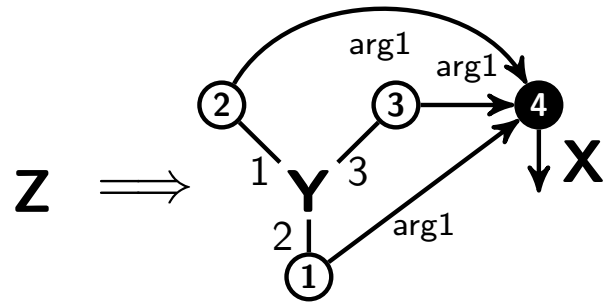


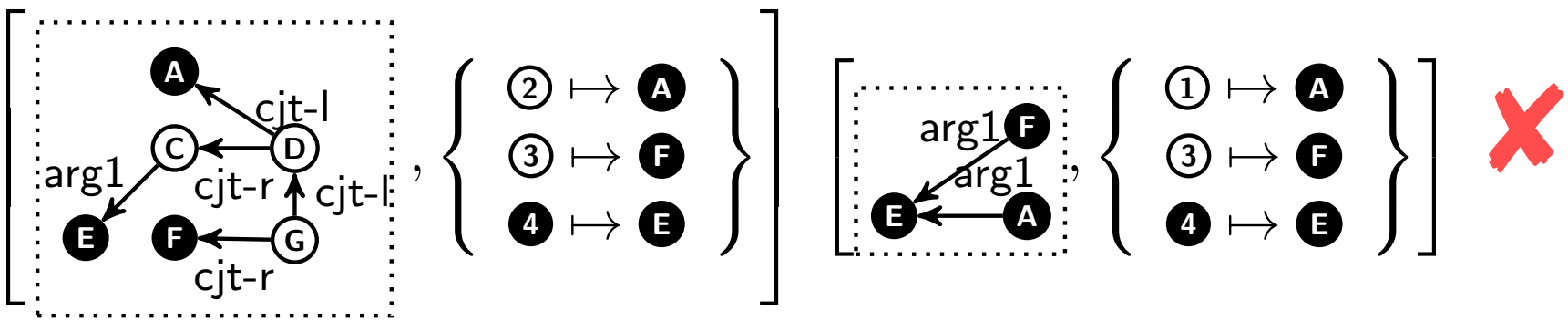


Fast Accessing of Chart Items

Edge-zero case (unifying nodes)

To integrate/glue two subgraphs, the nodes on the boundary must be identical in terms of their mapping relations to the input graph.

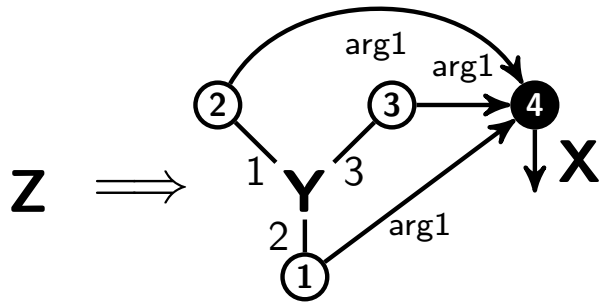


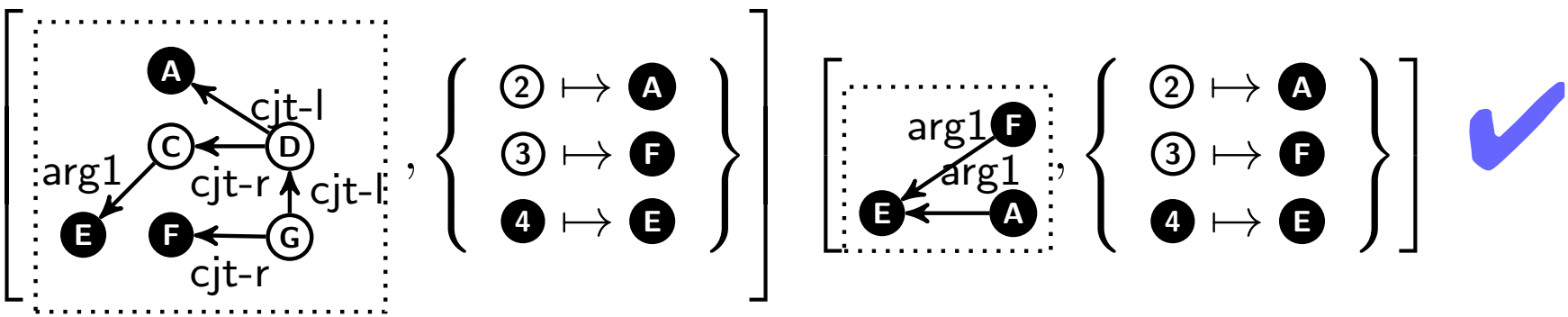


Fast Accessing of Chart Items

Edge-zero case (unifying nodes)

To integrate/glue two subgraphs, the nodes on the boundary must be identical in terms of their mapping relations to the input graph.

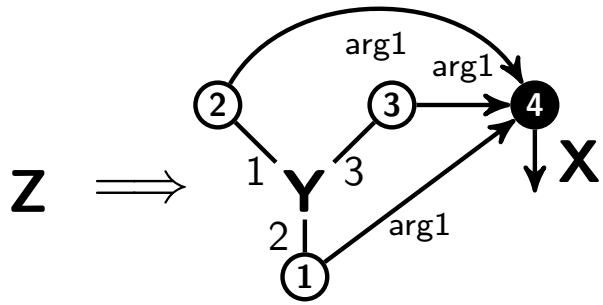


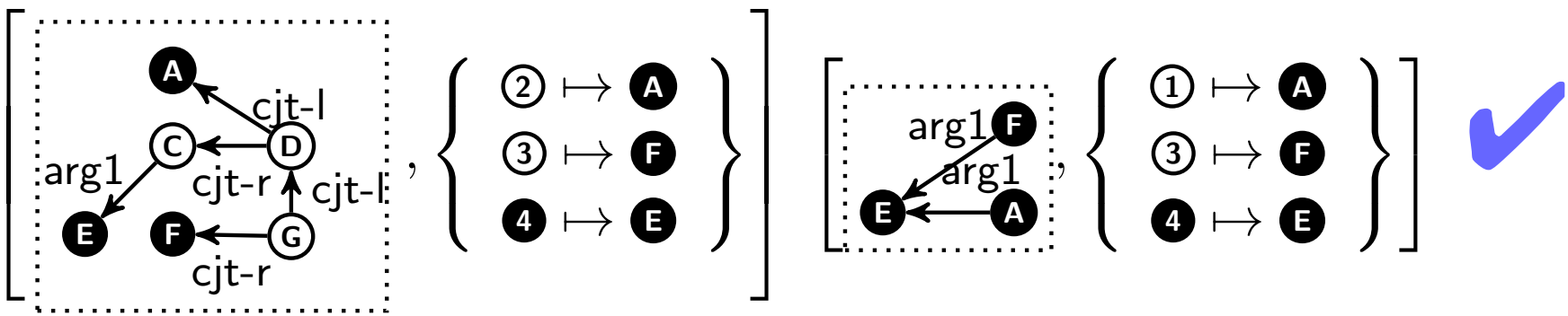


Fast Accessing of Chart Items

Edge-zero case (unifying nodes)

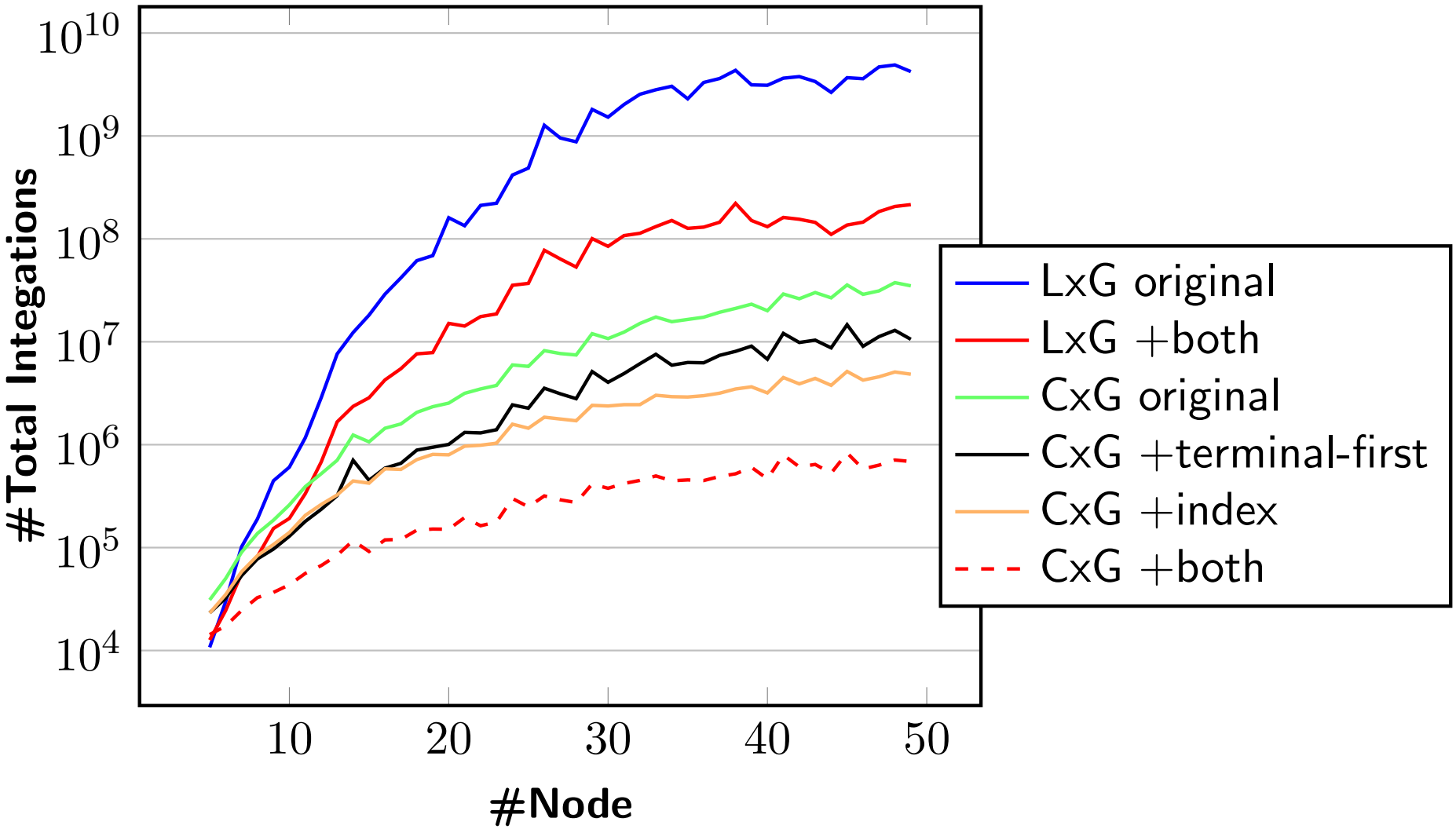
To integrate/glue two subgraphs, the nodes on the boundary must be identical in terms of their mapping relations to the input graph.





We put the mappings into the indexing keys of chart items.

Results on DeepBank (1)

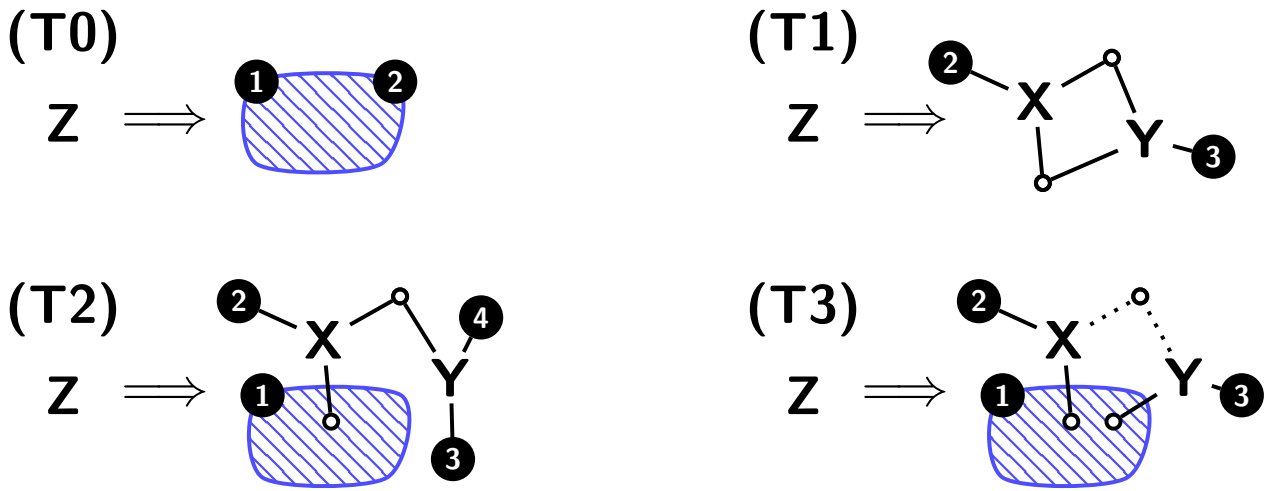


Results on DeepBank (2)

		Time(s)	**#Succ/#Total**
LxG	original	21.64	0.21%
	+terminal-first	21.02	0.12%
	+index	1.93	4.24%
	+both	1.51	2.53%
CxG	original	0.41	5.67%
	+terminal-first	0.12	2.23%
	+index	0.32	32.34%
	+both	0.07	29.34%

- **terminal-first** means the terminal-first match strategy;
- **index** means the fast item accessing method;
- **both** means to use both *terminal-first* and *index*.

Weakly Regular Graph Grammar (1)



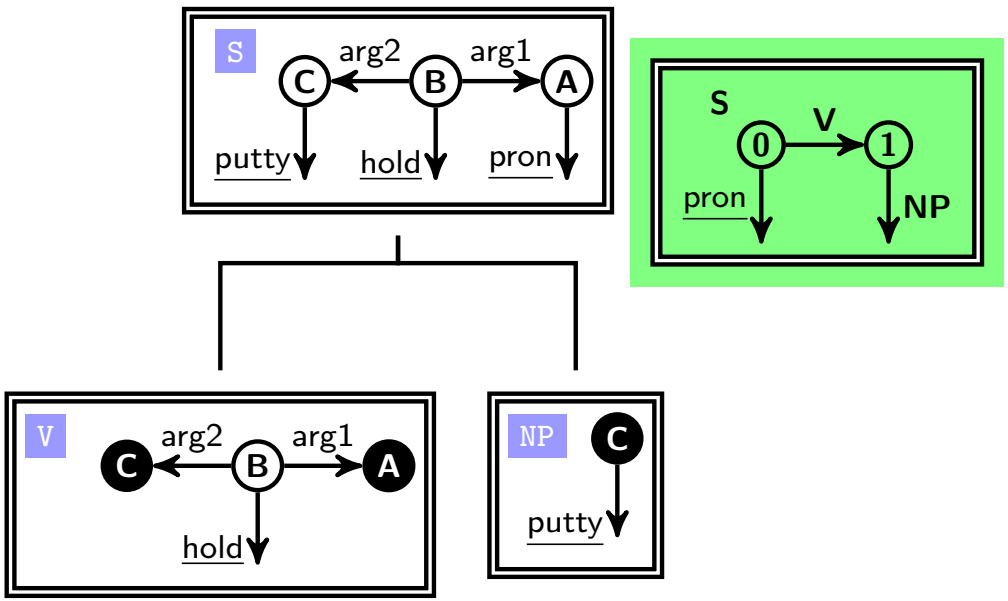
Definition

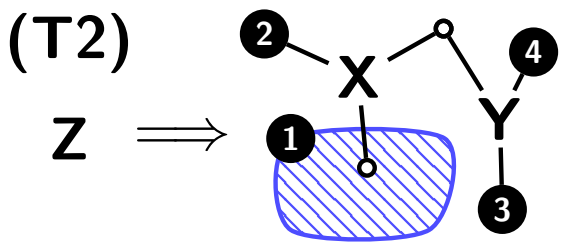
A node v in an edge-labeled graph G is **free**, if $E(v)$ contains only nonterminal edge(s). The number of those nodes is denoted by $f(G)$.

Definition

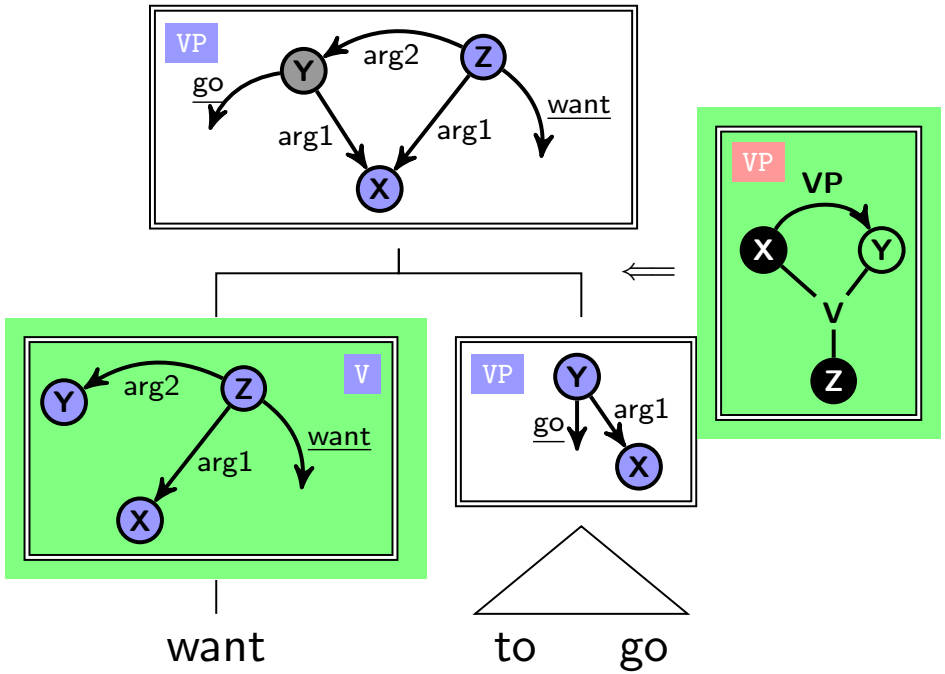
A weakly regular rule $A \to R$ satisfy the following conditions: (1) R is connected; (2) $term(R)$ is an empty graph or a connected graph; (3) if a free node of R is incident to only one edge, it is also an external node.

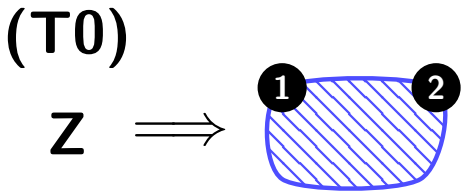
Weakly Regular Graph Grammar (2)

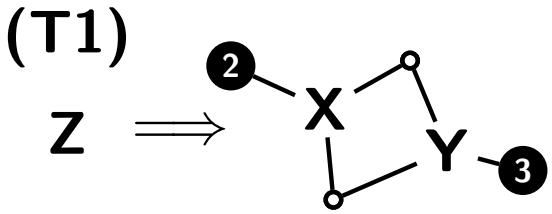


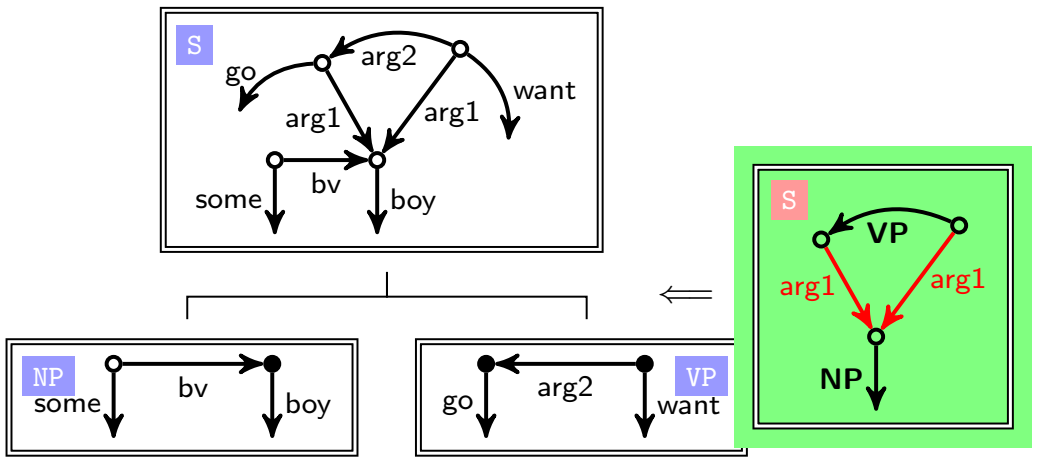


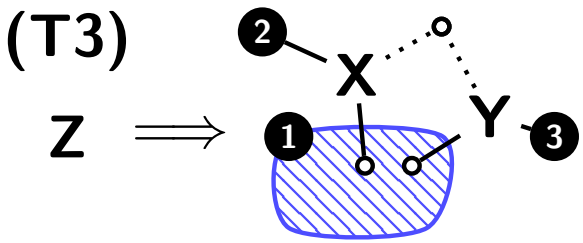
Weakly Regular Graph Grammar (3)



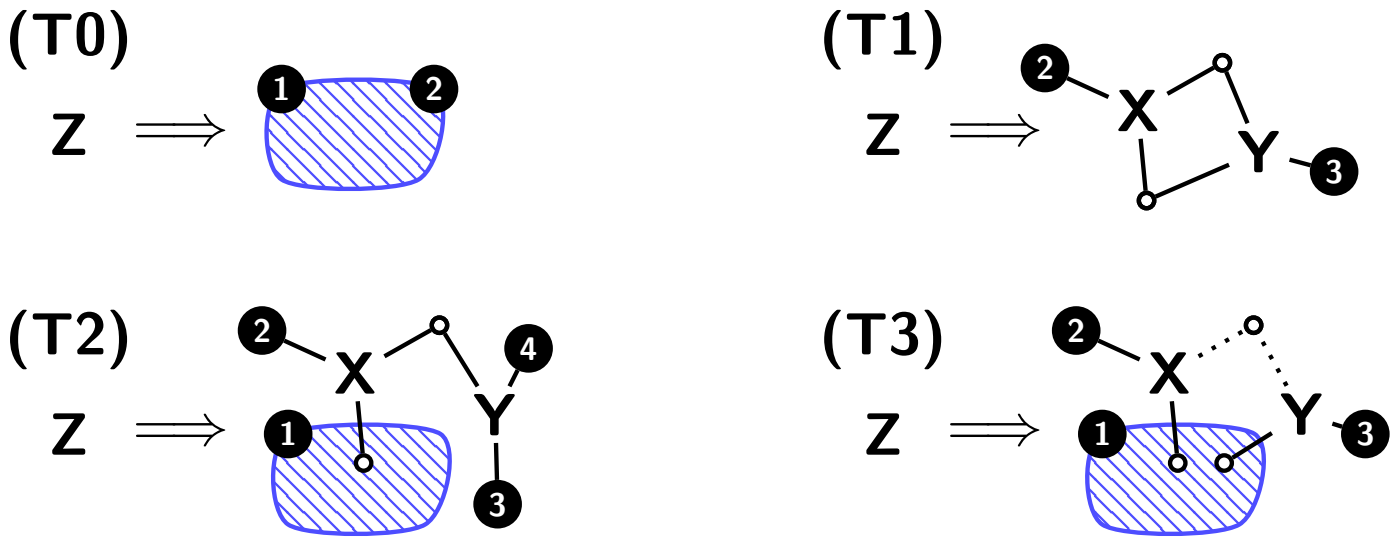








Weakly Regular Graph Grammar (4)



Proposition

If $A \to R$ is binary and weakly regular, then $\delta(R) = f(R)$ or $f(R) + 1$.

Readings

- Y. Chen and W. Sun. Parsing into Variable-in-situ Logico-Semantic Graphs.
- Y. Ye and W. Sun. Exact yet Efficient Graph Parsing, Bi-directional Locality and the Constructivist Hypothesis.