Left-corner Transitions on Dependency Parsing

Hiroshi Noji and Yusuke Miyao

National Institute of Informatics (NII)
Tokyo

Big picture

Big picture

Goal: Cognitively plausible parser with human like memory constraint

Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

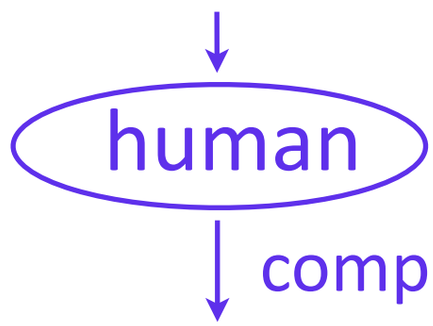
This is a parsing paper

Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

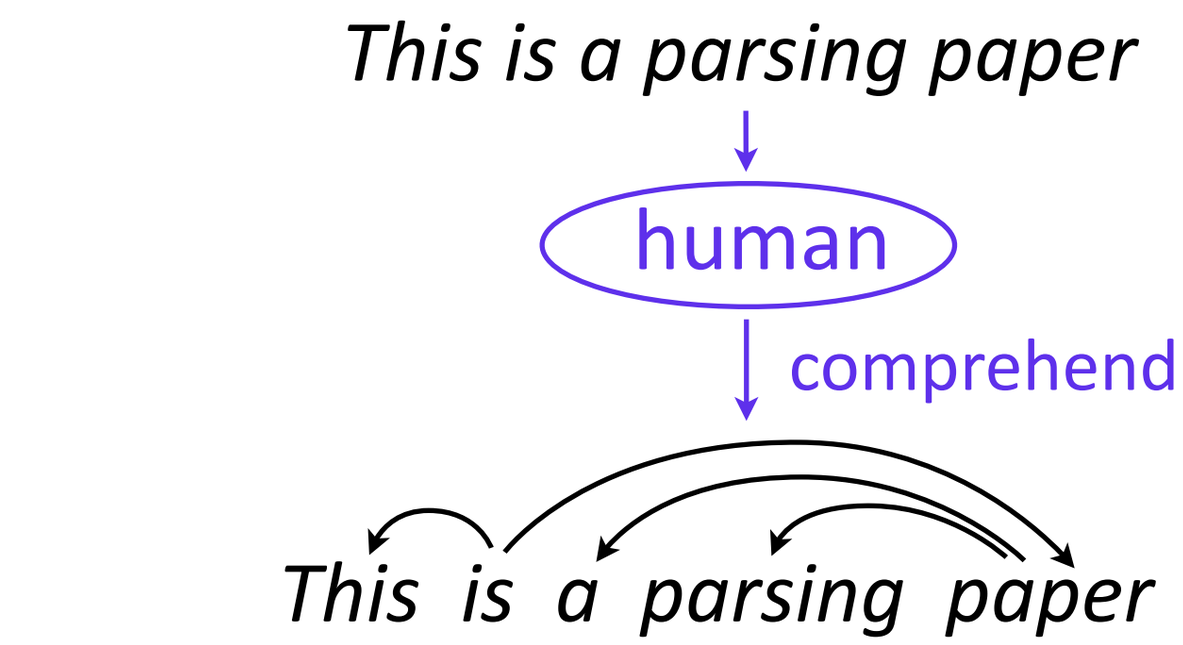
This is a parsing paper



Big picture

Goal: Cognitively plausible parser with human like memory constraint

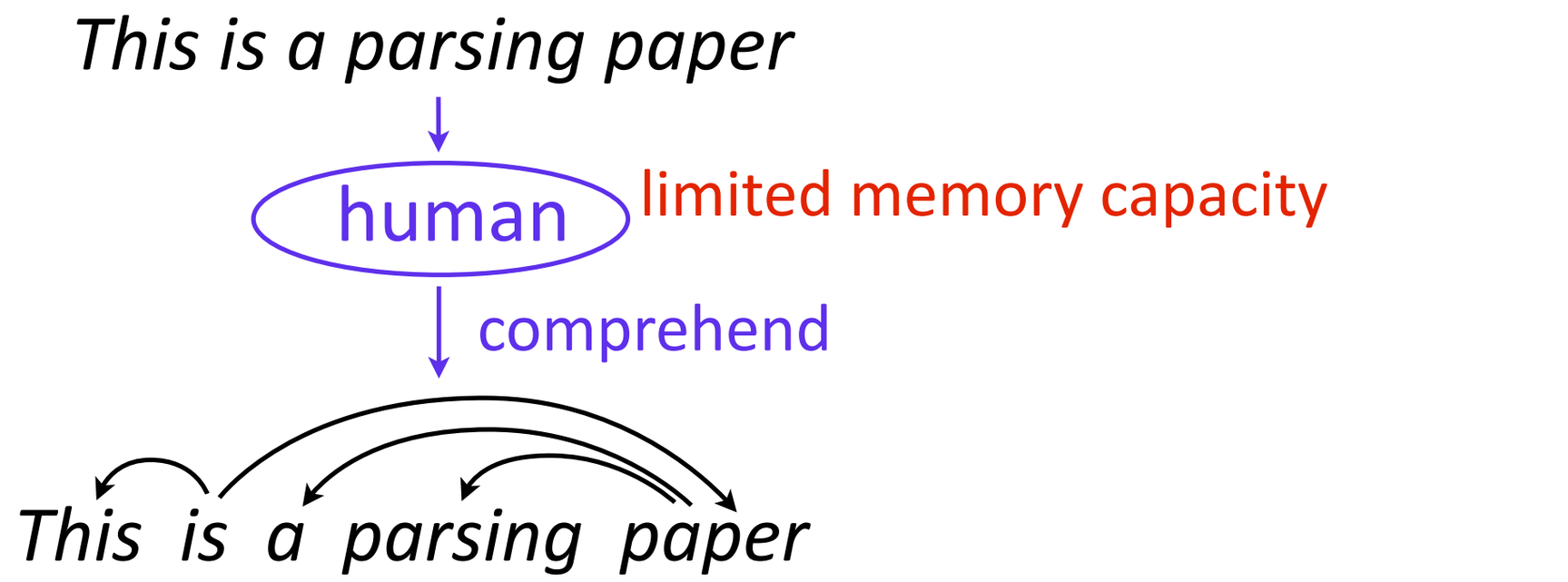
Assumption: People parse sentences within limited memory capacity



Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity



Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

The rat that the cat that the dog chased bit ate the cheese

Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

The rat that the cat that the dog chased bit ate the cheese

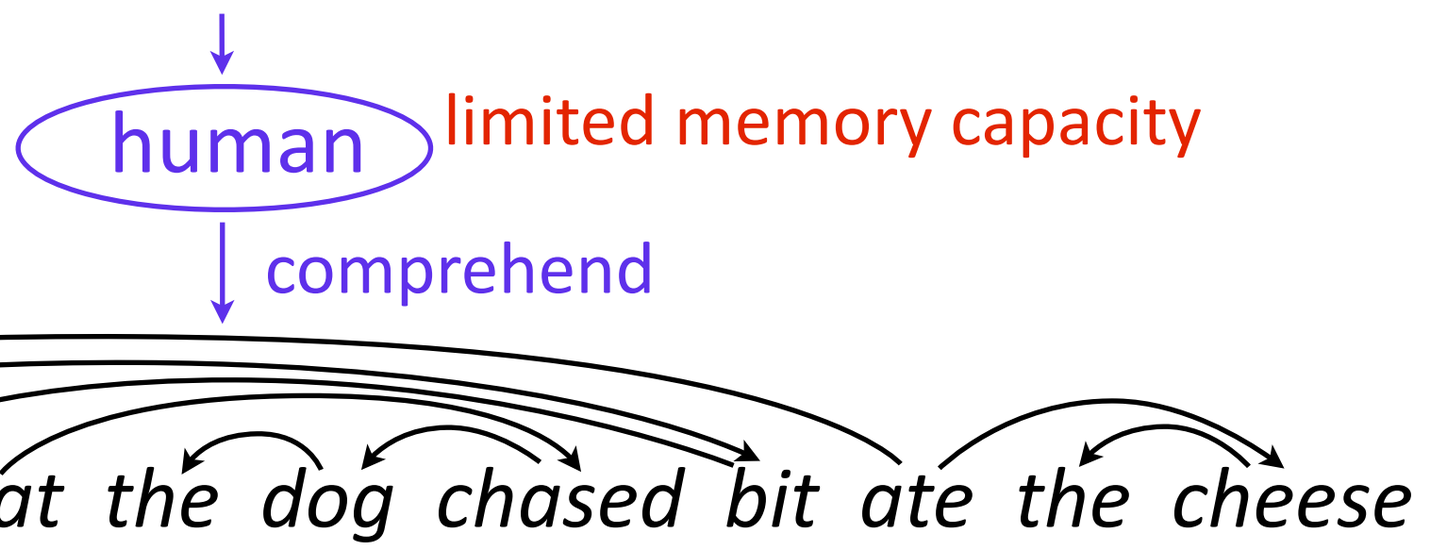
???

Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

The rat that the cat that the dog chased bit ate the cheese



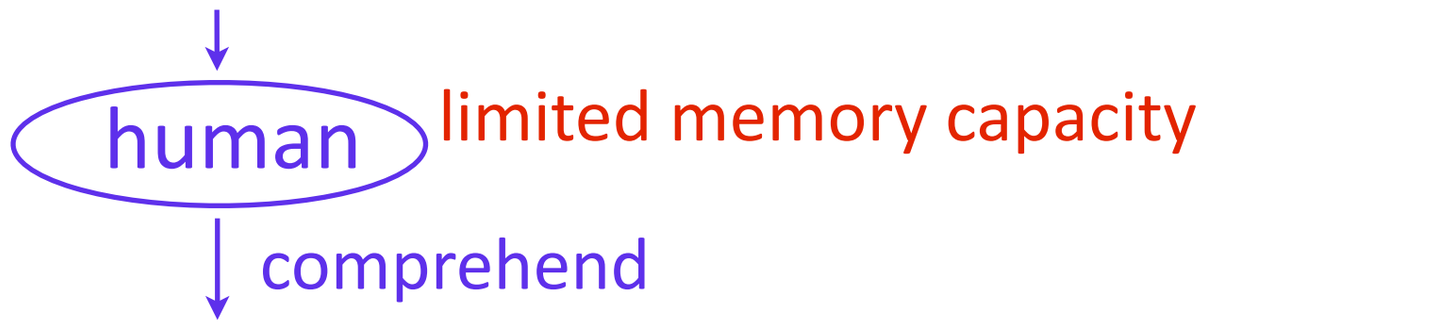
Big picture

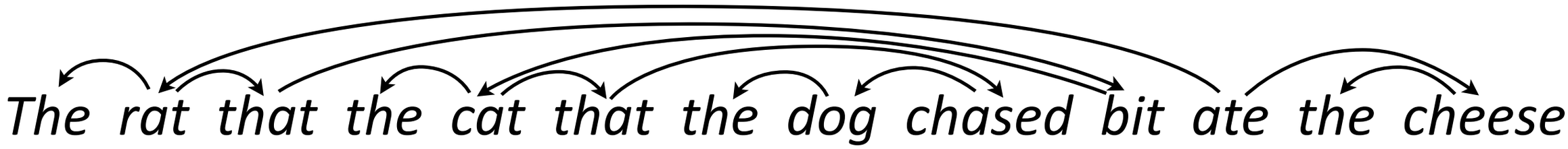
Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

⇒ Some syntactic construction causes memory overflow

The rat that the cat that the dog chased bit ate the cheese





Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

⇒ Some syntactic construction causes memory overflow

The dog chased the cat that bit the rat that ate the cheese

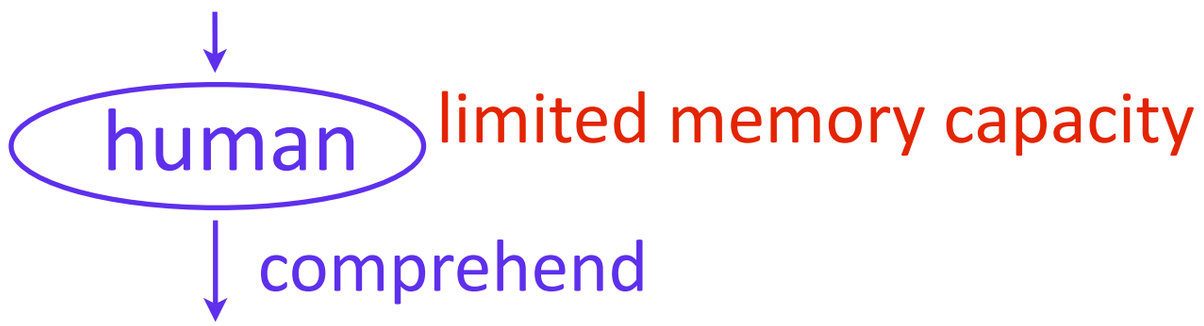
Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

⇒ Some syntactic construction causes memory overflow

The dog chased the cat that bit the rat that ate the cheese



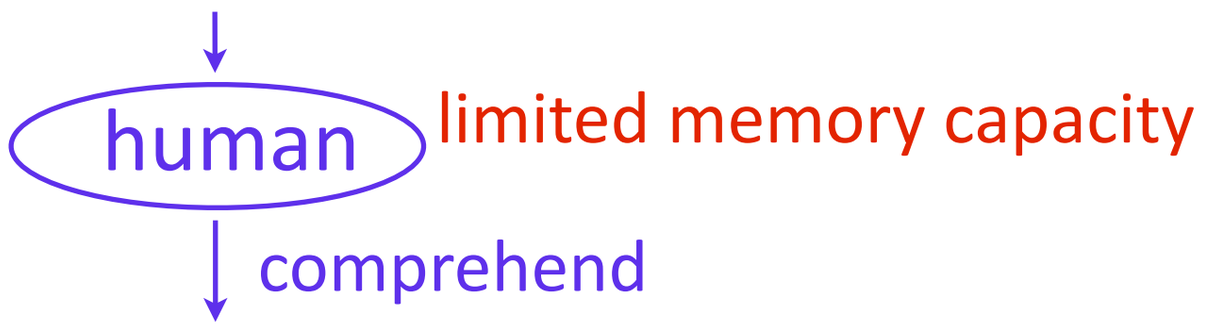
Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

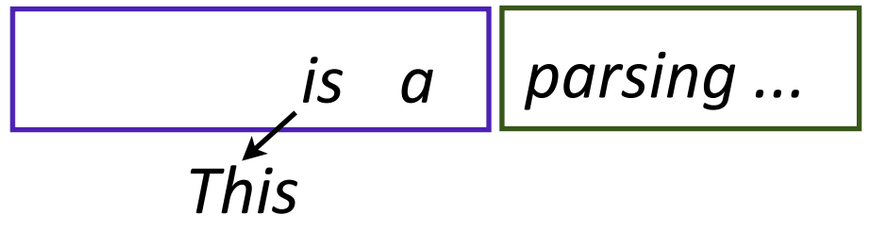
⇒ Some syntactic construction causes memory overflow

The dog chased the cat that bit the rat that ate the cheese



Framework: Stack-based incremental parser

- parse tree is constructed in a stack



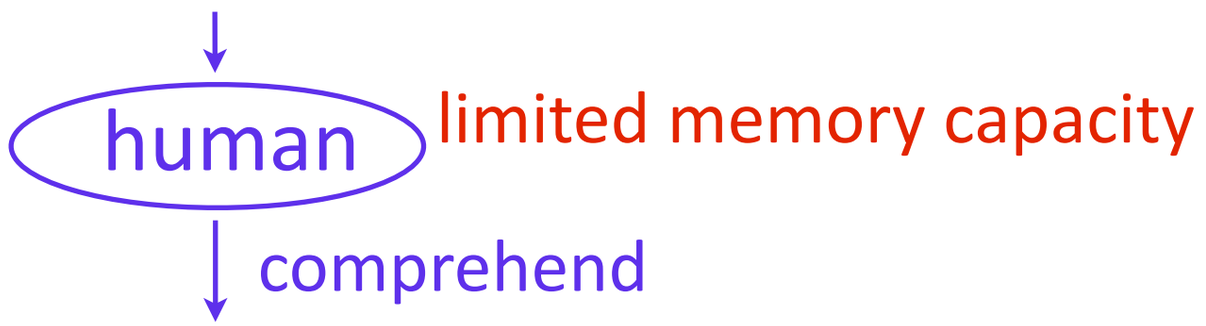
Big picture

Goal: Cognitively plausible parser with human like memory constraint

Assumption: People parse sentences within limited memory capacity

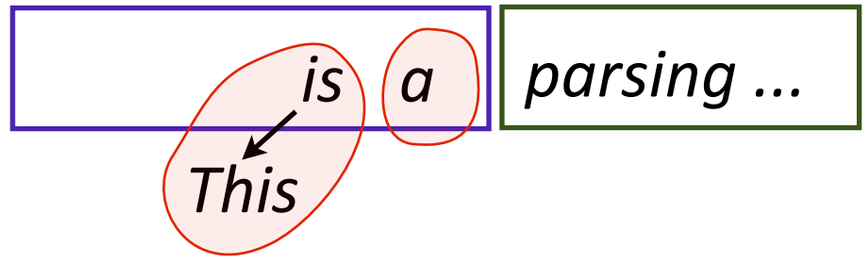
⇒ Some syntactic construction causes memory overflow

The dog chased the cat that bit the rat that ate the cheese



Framework: Stack-based incremental parser

- parse tree is constructed in a stack
- not incur large memory cost for daily occurring sentences



memory cost:

trees in the stack

Our approach: Left-corner for dependency

Left-corner parser [Resnik, 1992; Johnson, 1998]

- A cognitively motivated stack-based parser [Schuler et al., 2010]

Our approach: Left-corner for dependency

Left-corner parser [Resnik, 1992; Johnson, 1998]

- A cognitively motivated stack-based parser [Schuler et al., 2010]

	Syntax	**Language**
Previous works	Constituency	English only

Our approach: Left-corner for dependency

Left-corner parser [Resnik, 1992; Johnson, 1998]

- A cognitively motivated stack-based parser [Schuler et al., 2010]

	Syntax	**Language**
Previous works	Constituency	English only
This work	Dependency	18 languages

Our approach: Left-corner for dependency

Left-corner parser [Resnik, 1992; Johnson, 1998]

- A cognitively motivated stack-based parser [Schuler et al., 2010]

	Syntax	**Language**
Previous works	Constituency	English only
This work	Dependency	18 languages

Contributions:

Our approach: Left-corner for dependency

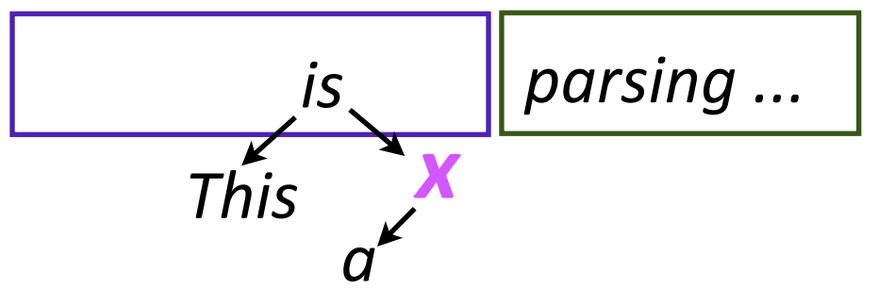
Left-corner parser [Resnik, 1992; Johnson, 1998]

- A cognitively motivated stack-based parser [Schuler et al., 2010]

	Syntax	**Language**
Previous works	Constituency	English only
This work	Dependency	18 languages

Contributions:

Technical: New transition-based dependency parser with left-corner parsing strategy



Our approach: Left-corner for dependency

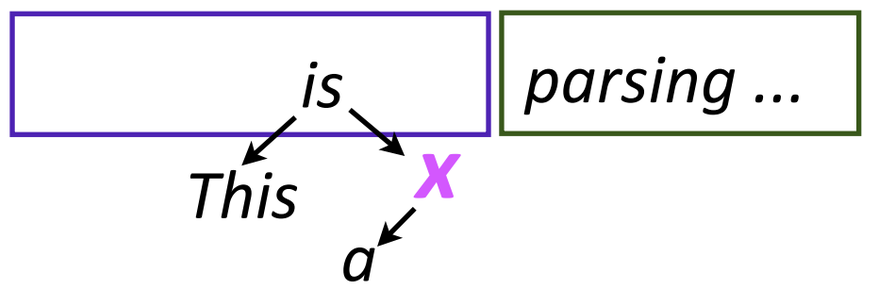
Left-corner parser [Resnik, 1992; Johnson, 1998]

- A cognitively motivated stack-based parser [Schuler et al., 2010]

	Syntax	**Language**
Previous works	Constituency	English only
This work	Dependency	18 languages

Contributions:

Technical: New transition-based dependency parser with left-corner parsing strategy



Empirical: Cross-linguistic analysis for its cognitive plausibility

- **Our finding: 17/18 languages rarely require memory cost > 4**

Our approach: Left-corner for dependency

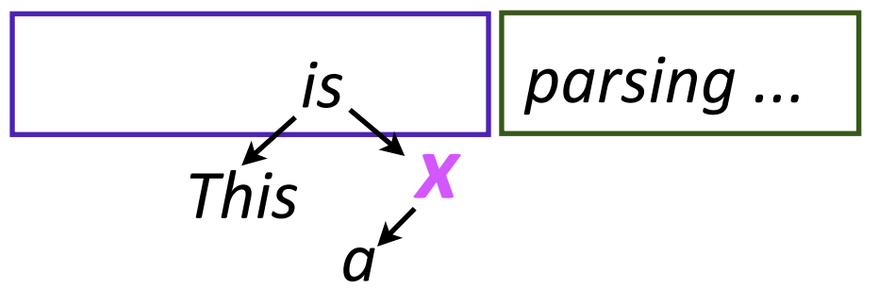
Left-corner parser [Resnik, 1992; Johnson, 1998]

- A cognitively motivated stack-based parser [Schuler et al., 2010]

	Syntax	**Language**
Previous works	Constituency	English only
This work	Dependency	18 languages

Contributions:

Technical: New transition-based dependency parser with left-corner parsing strategy



Empirical: Cross-linguistic analysis for its cognitive plausibility

- **Our finding: 17/18 languages rarely require memory cost > 4**

NOTE: We don't report any automatic parsing results in this work

Outline

Setup

- Psycholinguistic observation
- Left-corner constituency parsing
- Memory cost of dependency parsers

Left-corner dependency parsing

- Key concept: dummy node
- Parse example and action definition

Experiments

- Memory cost during oracle transitions
- Exploration of syntactic biases

Outline

Setup

- Psycholinguistic observation
- Left-corner constituency parsing
- Memory cost of dependency parsers

Left-corner dependency parsing

- Key concept: dummy node
- Parse example and action definition

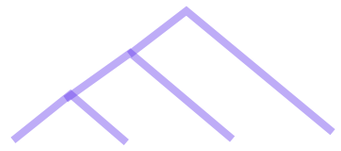
Experiments

- Memory cost during oracle transitions
- Exploration of syntactic biases

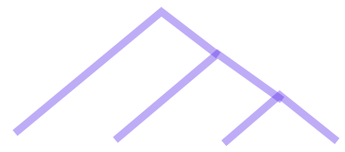
Setup: psycholinguistic observation

Correlations between tree structure and human memory load

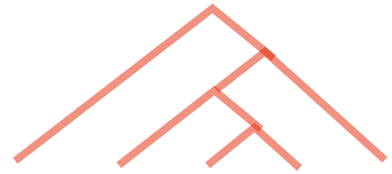
- left-branching



- right-branching



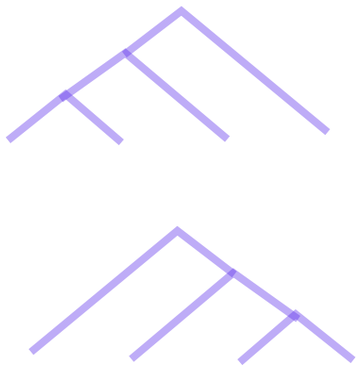
- center-embedding



Setup: psycholinguistic observation

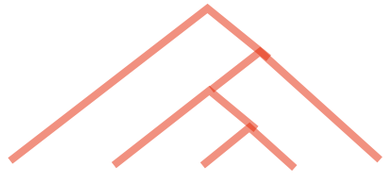
Correlations between tree structure and human memory load

- left-branching
- right-branching



} easy to parse for people

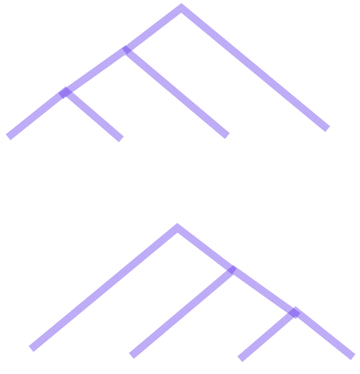
- center-embedding



Setup: psycholinguistic observation

Correlations between tree structure and human memory load

- left-branching
- right-branching



} easy to parse for people

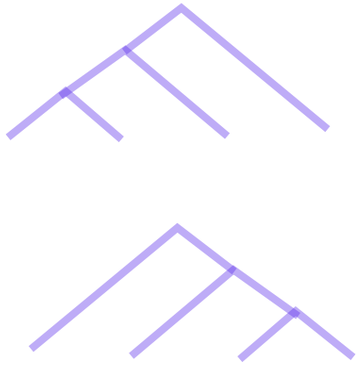
- center-embedding

} extremely difficult

Setup: psycholinguistic observation

Correlations between tree structure and human memory load

- left-branching
- right-branching



} easy to parse for people

- center-embedding

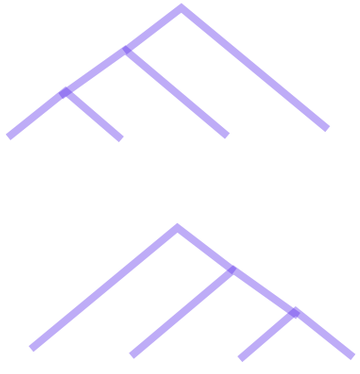
} extremely difficult

Example:

Setup: psycholinguistic observation

Correlations between tree structure and human memory load

- left-branching
- right-branching



} easy to parse for people

- center-embedding

} extremely difficult

Example:

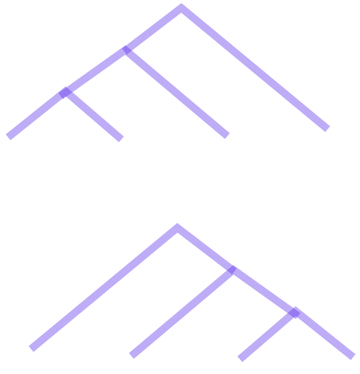
Right-branching:

The dog chased the cat that bit the rat that ate the cheese

Setup: psycholinguistic observation

Correlations between tree structure and human memory load

- left-branching
- right-branching



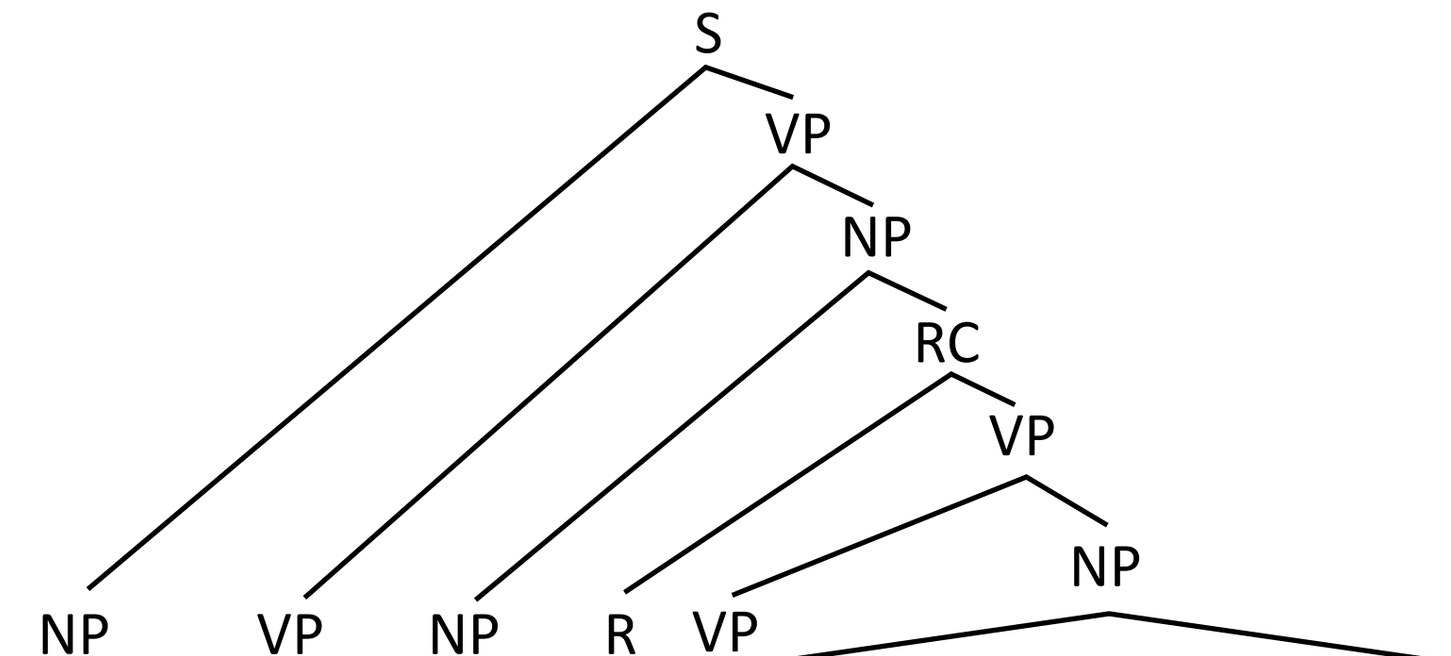
} easy to parse for people

- center-embedding

} extremely difficult

Example:

Right-branching:

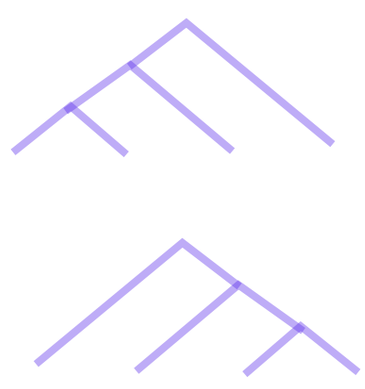


The dog chased the cat that bit the rat that ate the cheese

Setup: psycholinguistic observation

Correlations between tree structure and human memory load

- left-branching
- right-branching



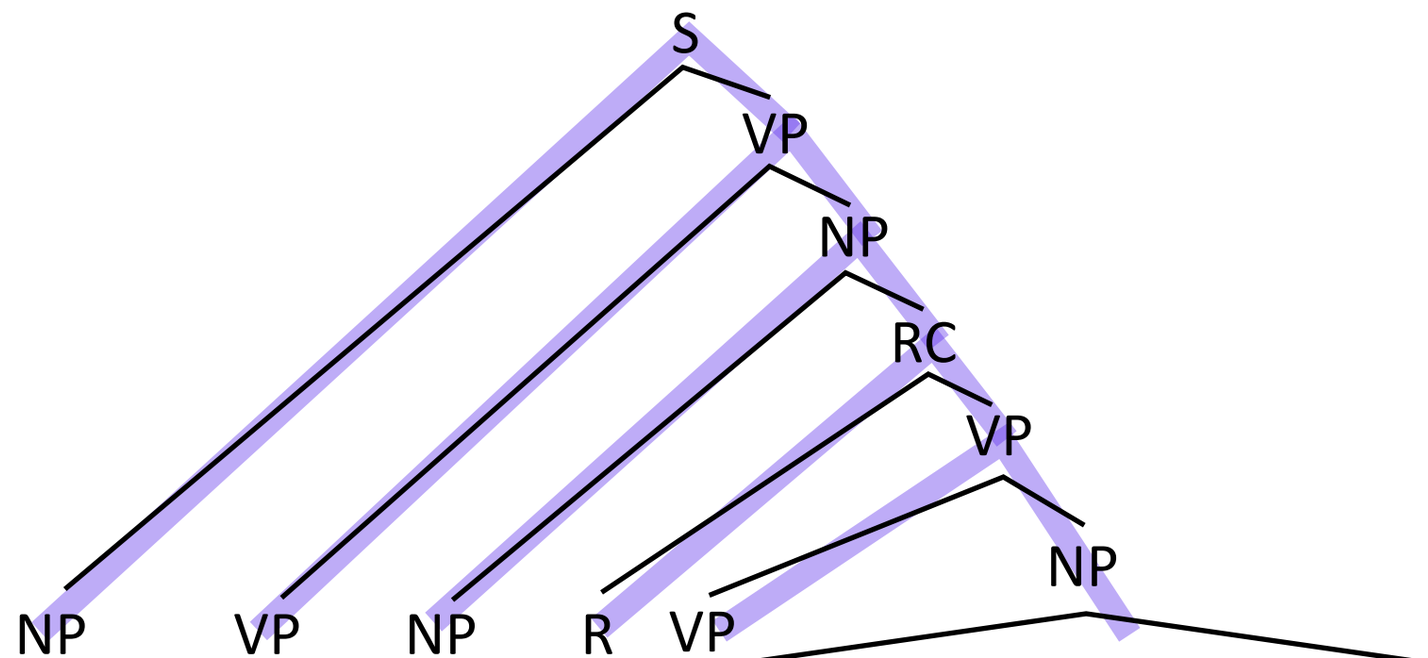
} easy to parse for people

- center-embedding

} extremely difficult

Example:

Right-branching:

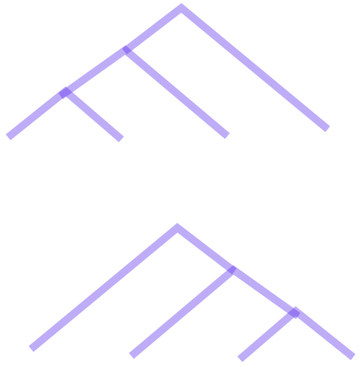


The dog chased the cat that bit the rat that ate the cheese

Setup: psycholinguistic observation

Correlations between tree structure and human memory load

- left-branching
- right-branching



} easy to parse for people

- center-embedding

} extremely difficult

Example:

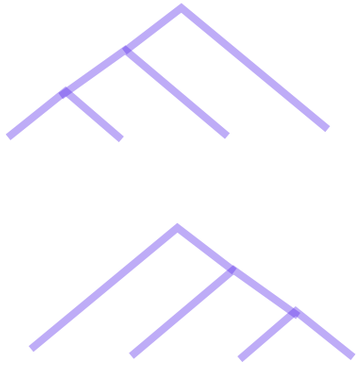
Center-embedding:

The rat that the cat that the dog chased bit ate the cheese

Setup: psycholinguistic observation

Correlations between tree structure and human memory load

- left-branching
- right-branching



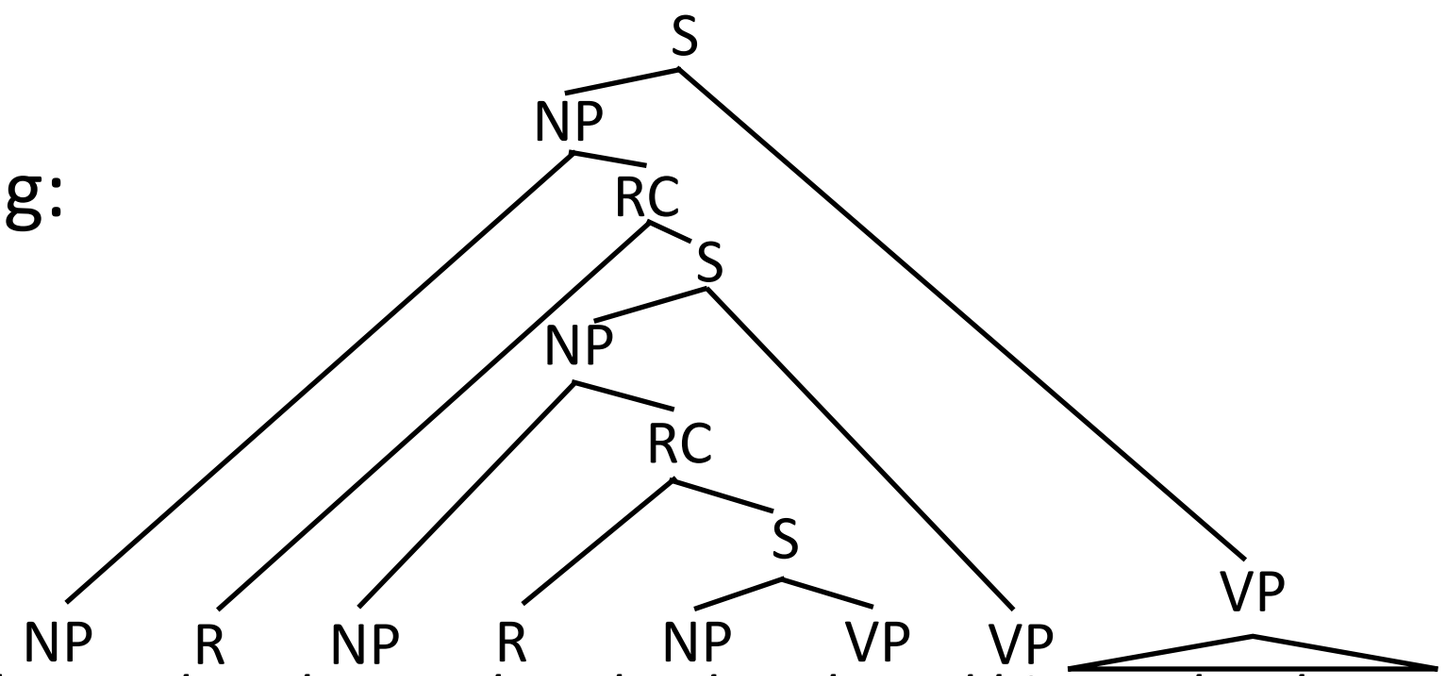
} easy to parse for people

- center-embedding

} extremely difficult

Example:

Center-embedding:

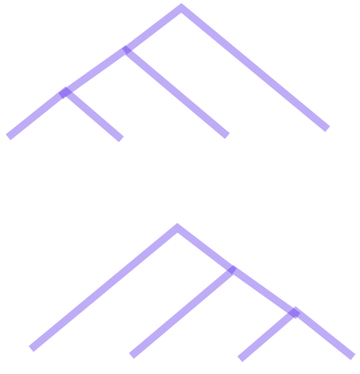


The rat that the cat that the dog chased bit ate the cheese

Setup: psycholinguistic observation

Correlations between tree structure and human memory load

- left-branching
- right-branching



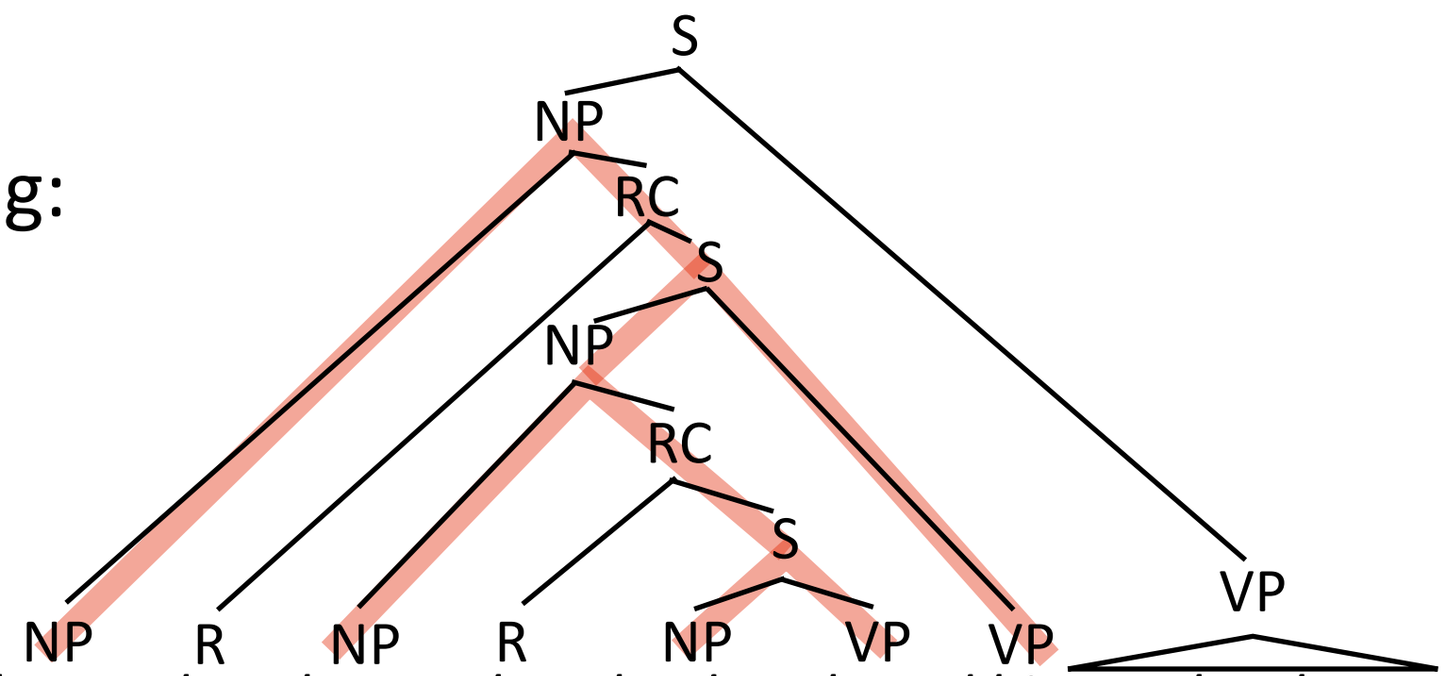
} easy to parse for people

- center-embedding

} extremely difficult

Example:

Center-embedding:



The rat that the cat that the dog chased bit ate the cheese

[Abney & Johnson, 1991; Resnik, 1992; Johnson, 1998]

Left-corner constituency parsing

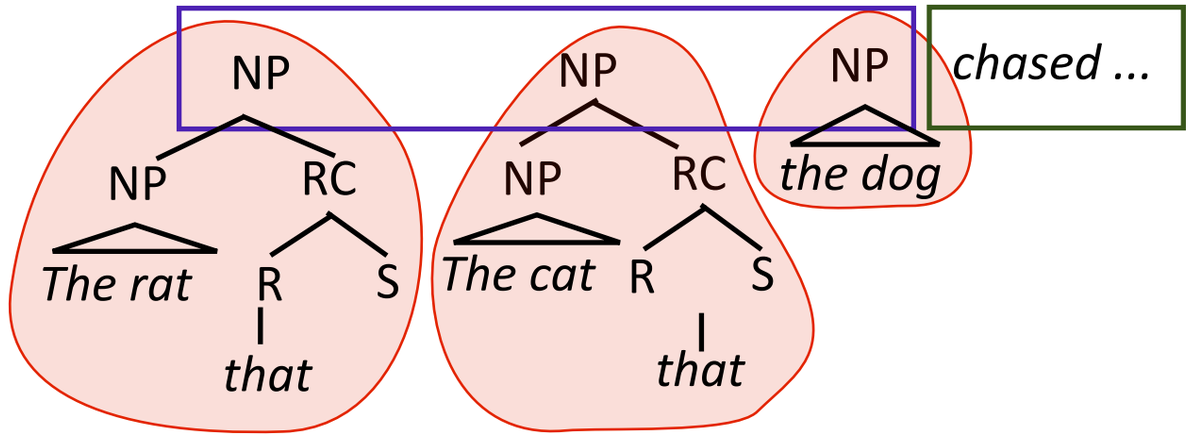
[Abney & Johnson, 1991; Resnik, 1992; Johnson, 1998]

Left-corner constituency parsing

Stack-based incremental parsing

- larger stack depth is required only with center-embedded structures

⇒ matches to the human observation



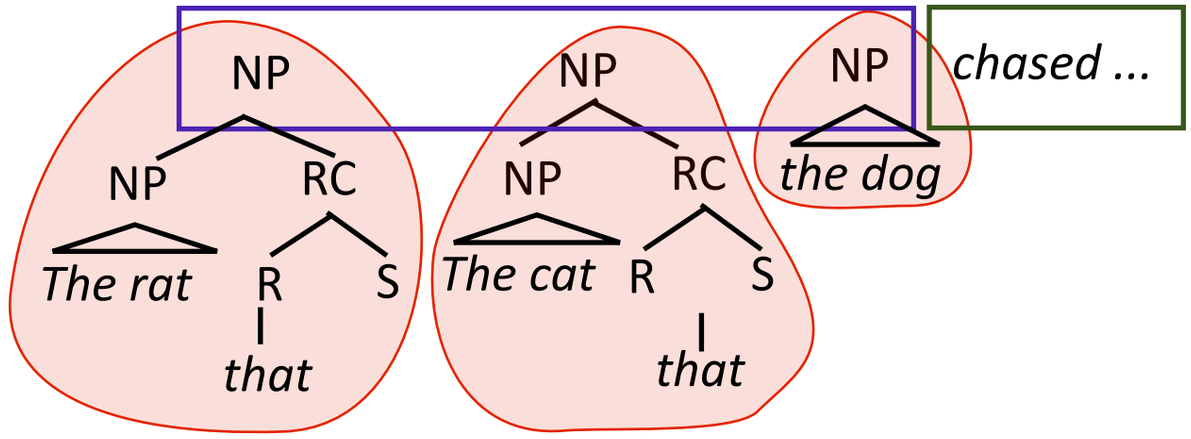
[Abney & Johnson, 1991; Resnik, 1992; Johnson, 1998]

Left-corner constituency parsing

Stack-based incremental parsing

- larger stack depth is required only with center-embedded structures

⇒ matches to the human observation



[Schuler et al., 2010] argues it is cognitively plausible, because...

- Almost all **English** treebank sentences don't require cost > 3
- High parsing accuracy within such memory constraint

[Abney & Johnson, 1991; Resnik, 1992; Johnson, 1998]

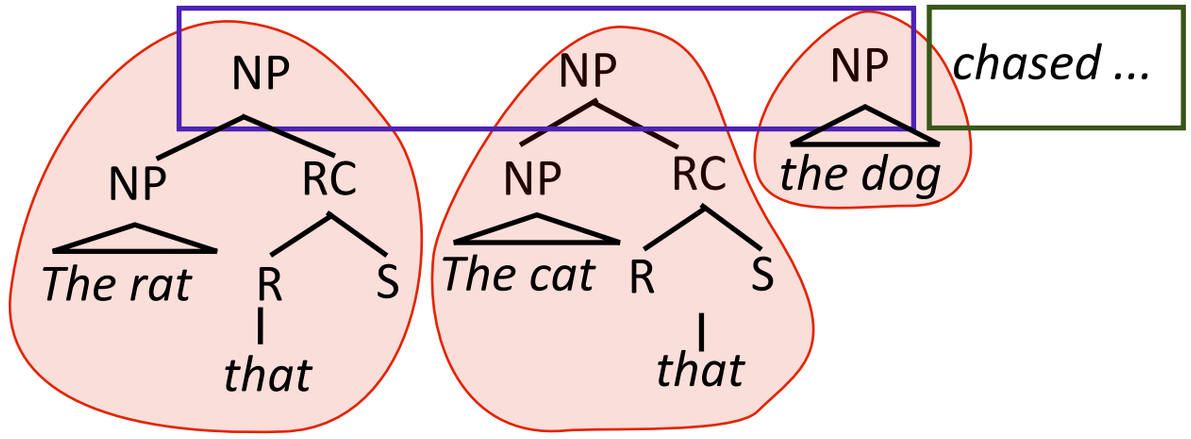
Left-corner constituency parsing

Stack-based incremental parsing

- larger stack depth is required only

with center-embedded structures

⇒ matches to the human observation



[Schuler et al., 2010] argues it is cognitively plausible, because...

- Almost all **English** treebank sentences don't require cost > 3
- High parsing accuracy within such memory constraint

Our goal: Stack-based **dependency** parser with cognitive plausibility

- Transition-based parser is a stack-based parser for dependency
- **What is the memory cost property of existing parsers?**

Outline

Setup

- Psycholinguistic observation
- Left-corner constituency parsing
- Memory cost of dependency parsers

Left-corner dependency parsing

- Key concept: dummy node
- Parse examples and action definitions

Experiments

- Memory cost during oracle transitions
- Exploration of syntactic biases

Setup: tree form for dependency

To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

Setup: tree form for dependency

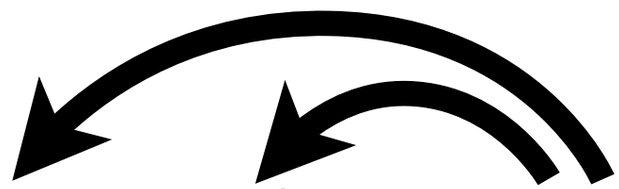
To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

Basic idea: we convert a dependency tree
into **an unlabeled CNF**

Setup: tree form for dependency

To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

Basic idea: we convert a dependency tree into **an unlabeled CNF**

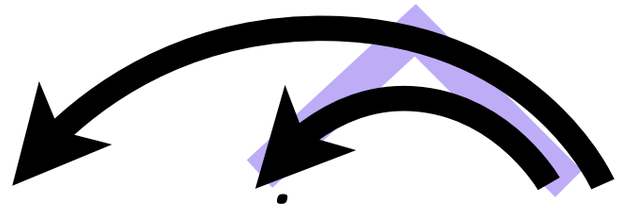


a parsing paper

Setup: tree form for dependency

To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

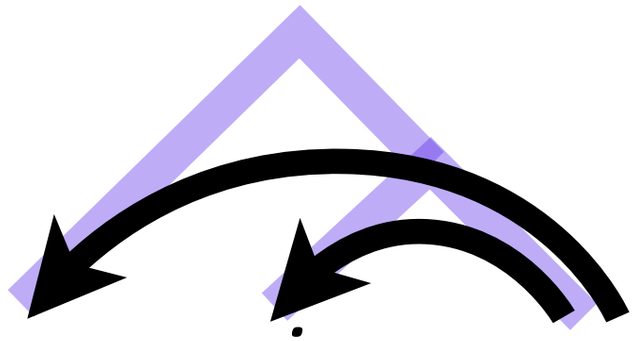
Basic idea: we convert a dependency tree into **an unlabeled CNF**



Setup: tree form for dependency

To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

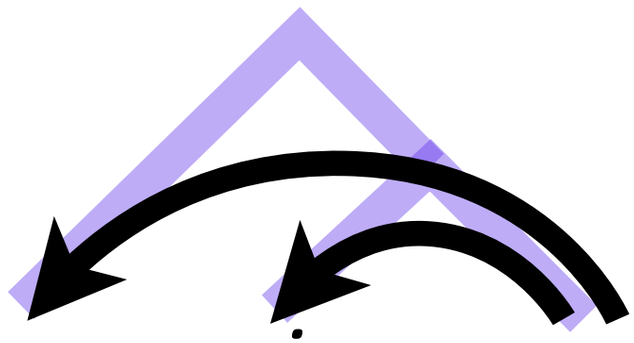
Basic idea: we convert a dependency tree into **an unlabeled CNF**



Setup: tree form for dependency

To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

Basic idea: we convert a dependency tree into **an unlabeled CNF**

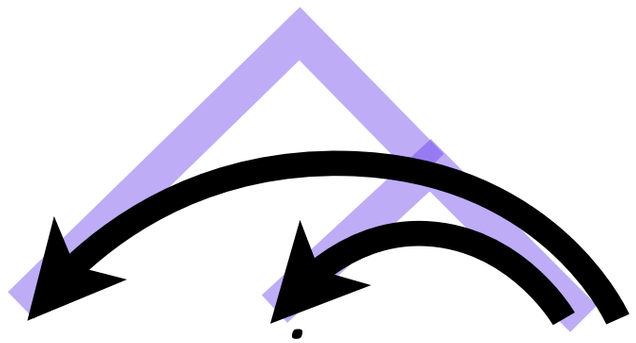


Example:

Setup: tree form for dependency

To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

Basic idea: we convert a dependency tree into **an unlabeled CNF**



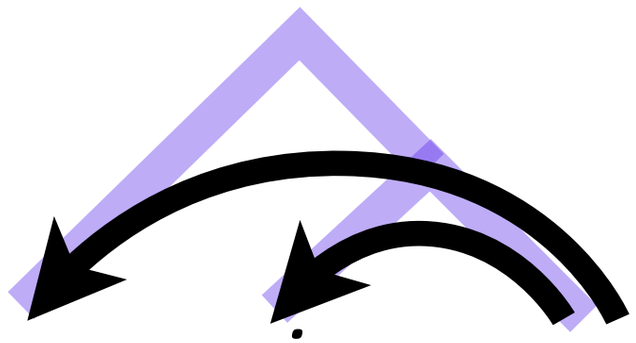
Example:

Left-branching:

Setup: tree form for dependency

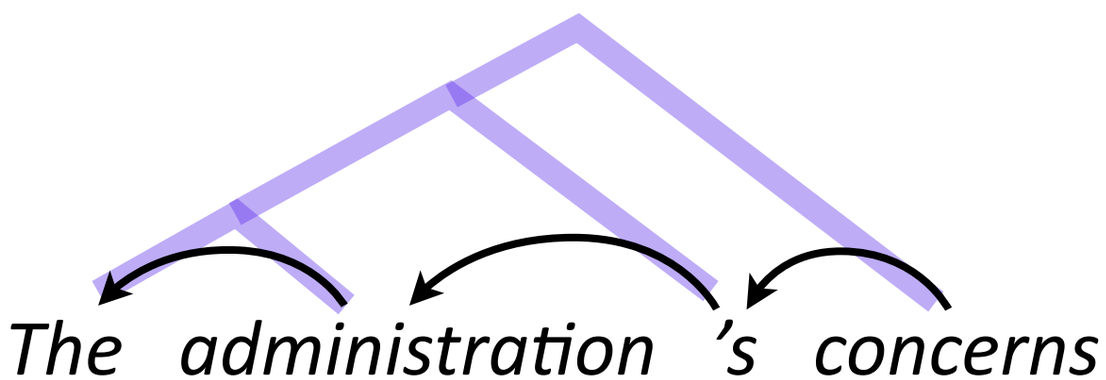
To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

Basic idea: we convert a dependency tree into **an unlabeled CNF**



Example:

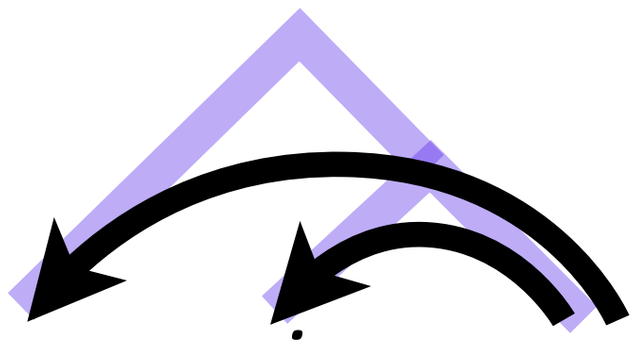
Left-branching:



Setup: tree form for dependency

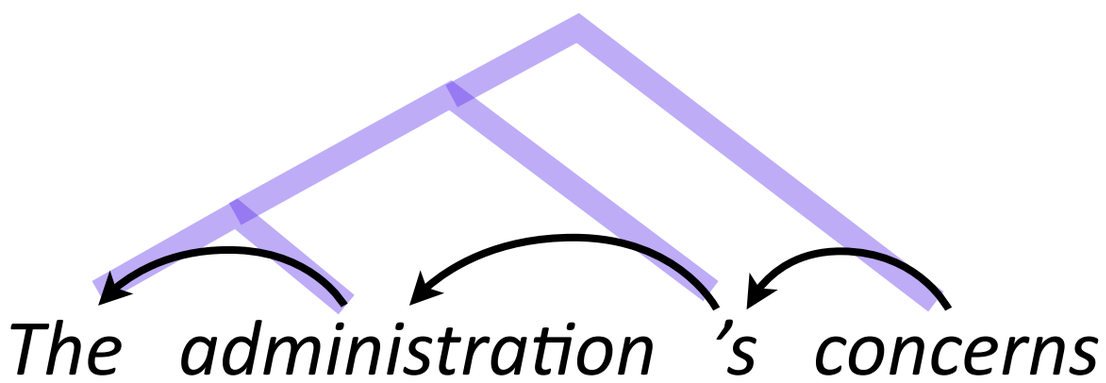
To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

Basic idea: we convert a dependency tree into **an unlabeled CNF**

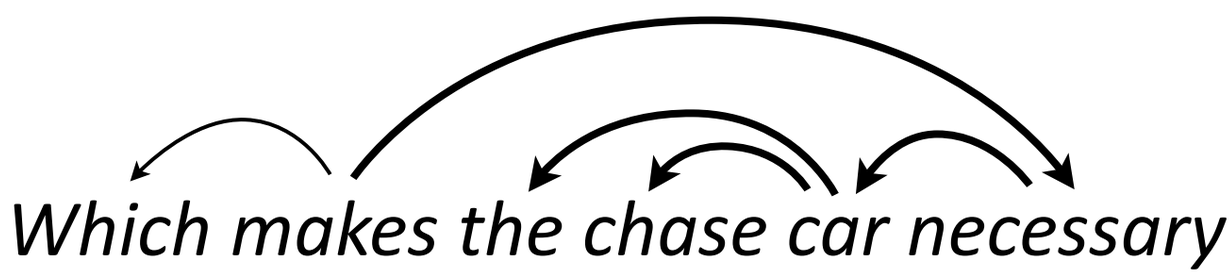


Example:

Left-branching:



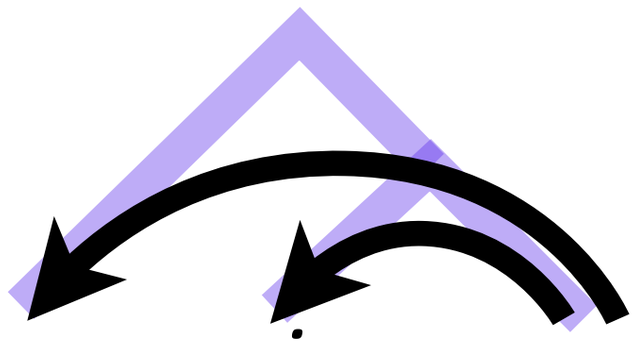
Center-embedding:



Setup: tree form for dependency

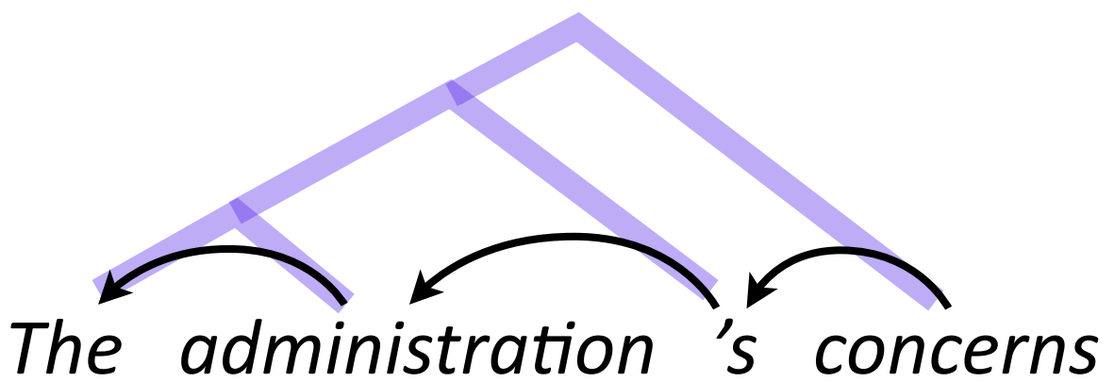
To discuss memory cost property of dependency parsers, we need to define, e.g., center-embedding for dependency structure

Basic idea: we convert a dependency tree into **an unlabeled CNF**

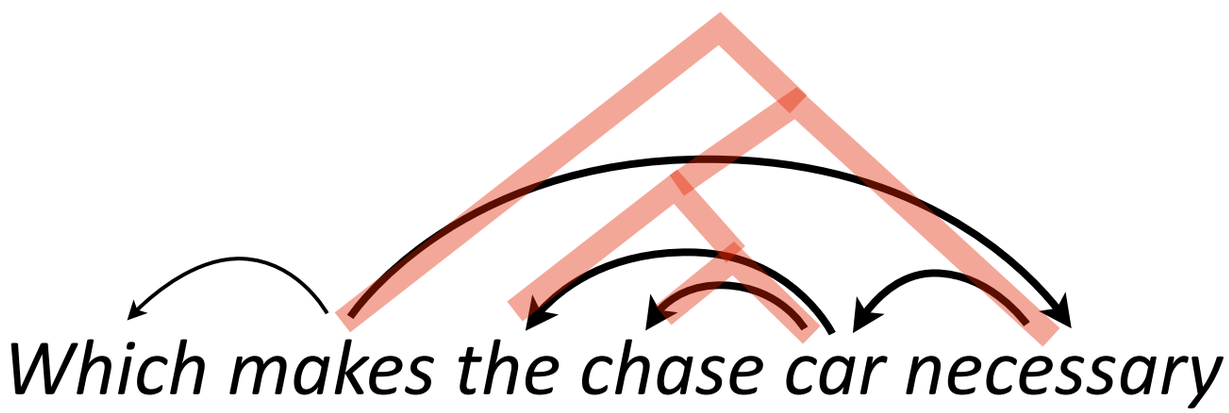


Example:

Left-branching:



Center-embedding:



Transition-based dependency parsing

Stack

most

Buffer

popular beer in Ireland

Incrementally...

Transition-based dependency parsing

Stack

most

Buffer

popular beer in Ireland

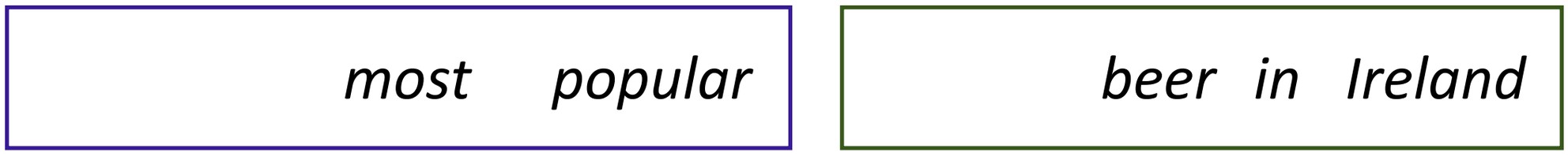
Incrementally...

- read a token from the buffer (shift)

Transition-based dependency parsing

Stack

Buffer



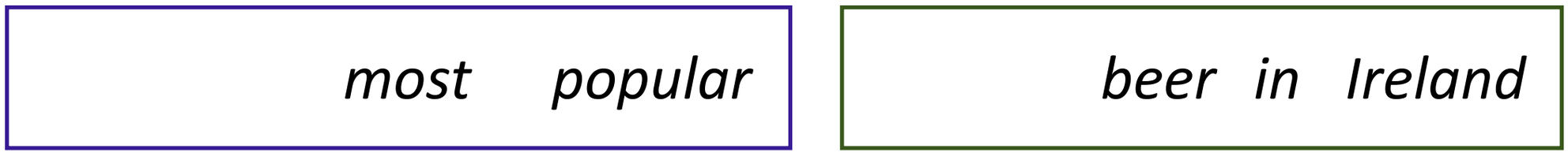
Incrementally...

- read a token from the buffer (shift)

Transition-based dependency parsing

Stack

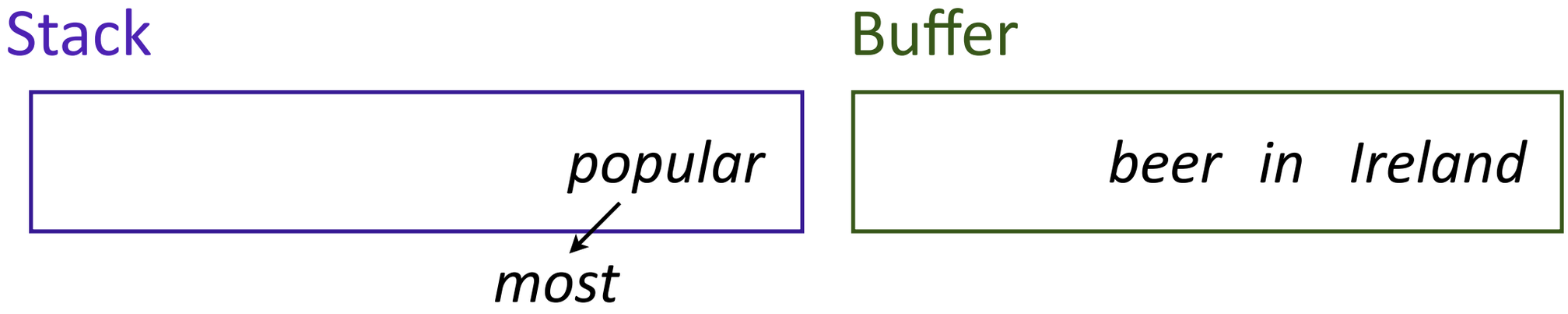
Buffer



Incrementally...

- read a token from the buffer (shift)
- create arcs in the stack (reduce)

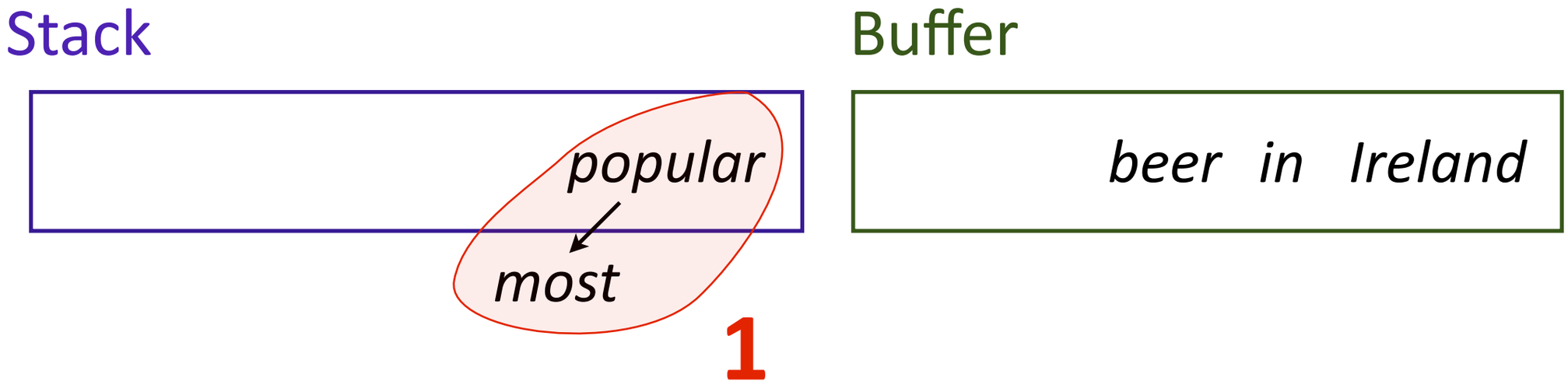
Transition-based dependency parsing



Incrementally...

- read a token from the buffer (shift)
- create arcs in the stack (reduce)

Transition-based dependency parsing



Incrementally...

- read a token from the buffer (shift)
- create arcs in the stack (reduce)

Memory cost for a parser: number of subtrees on the stack

Several transition systems

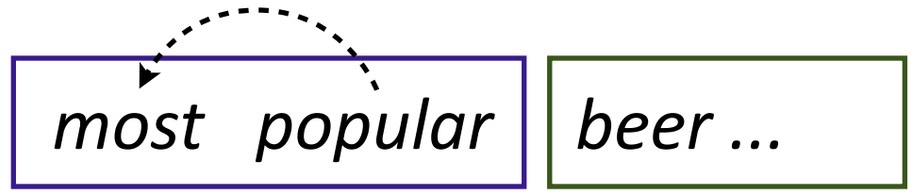
Each transition system defines **different transition actions**

Several transition systems

Each transition system defines **different transition actions**

- Arc-standard [Nivre, 2004]

  ⇒ connect two words in the stack

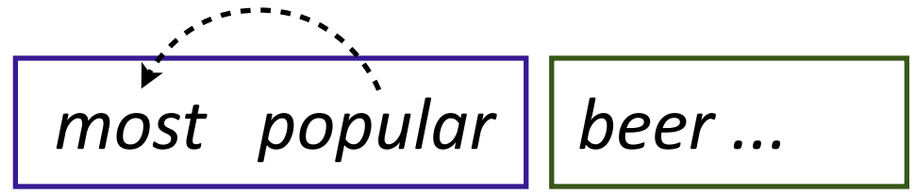


Several transition systems

Each transition system defines **different transition actions**

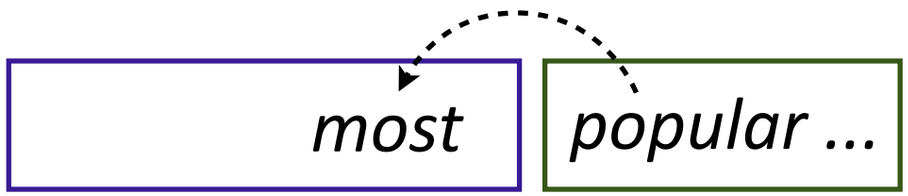
- Arc-standard [Nivre, 2004]

  ⇒ connect two words in the stack



- Arc-eager [Nivre, 2003]

  ⇒ connect words in stack/buffer

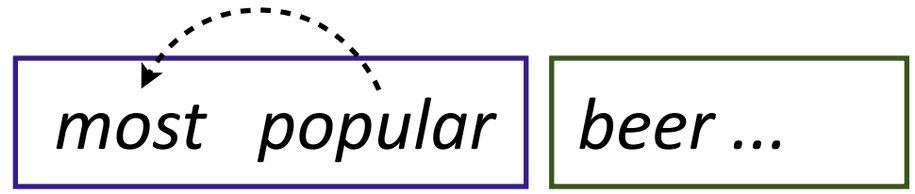


Several transition systems

Each transition system defines **different transition actions**

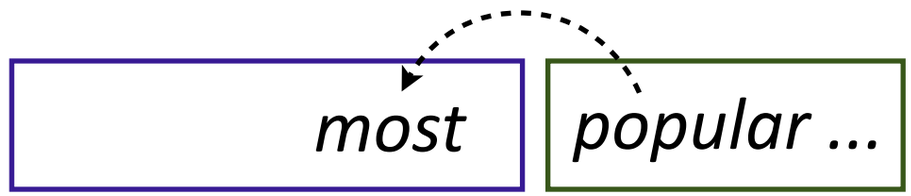
- Arc-standard [Nivre, 2004]

  ⇒ connect two words in the stack



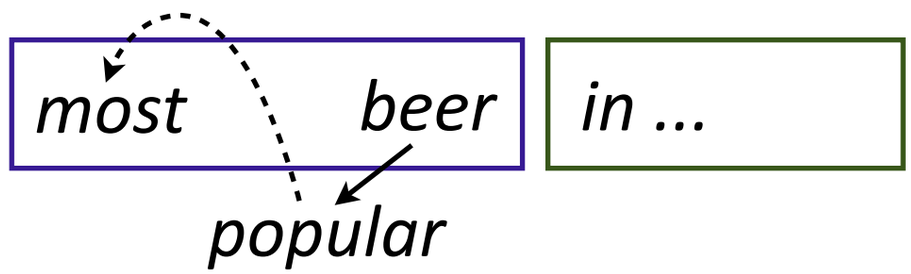
- Arc-eager [Nivre, 2003]

  ⇒ connect words in stack/buffer



- Spine [Kitagawa & Tanaka-Ishii, 2011; Sartorio et al., 2013]

  ⇒ attaches to a token in a spine

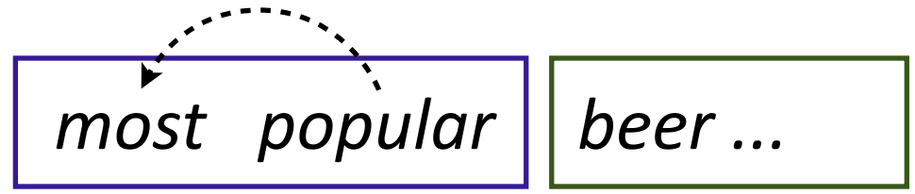


Several transition systems

Each transition system defines **different transition actions**

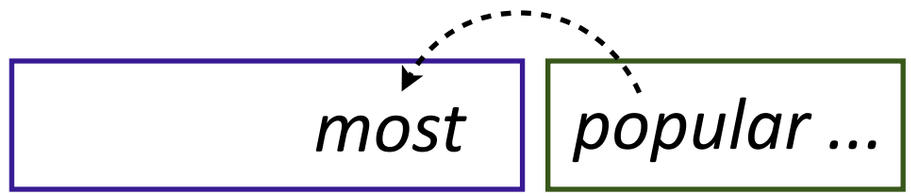
- Arc-standard [Nivre, 2004]

  ⇒ connect two words in the stack



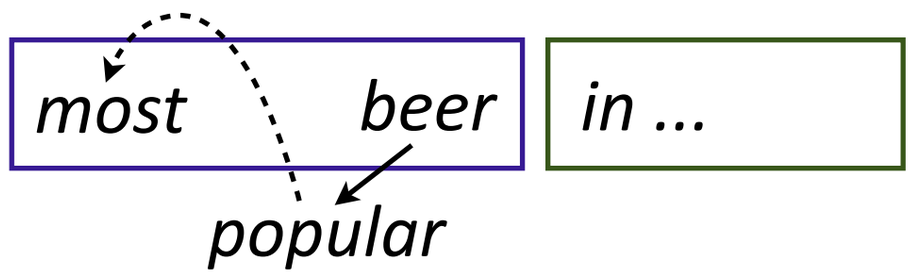
- Arc-eager [Nivre, 2003]

  ⇒ connect words in stack/buffer



- Spine [Kitagawa & Tanaka-Ishii, 2011; Sartorio et al., 2013]

  ⇒ attaches to a token in a spine



Each system has a different property of memory cost

Desired system: memory cost increases only with center-embedding

Arc-standard: Memory property

Arc-standard: Memory property

Arc-standard is not a desirable transition system:

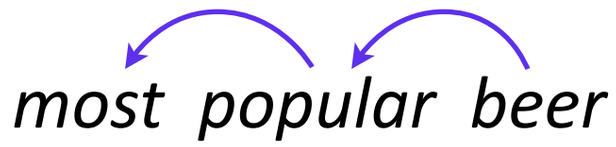
- Different memory cost for left- and right-branchings

Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

left-branching:

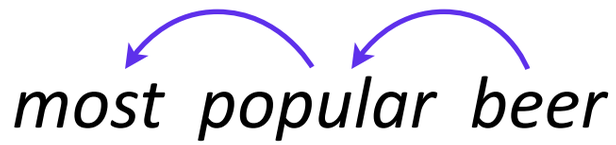


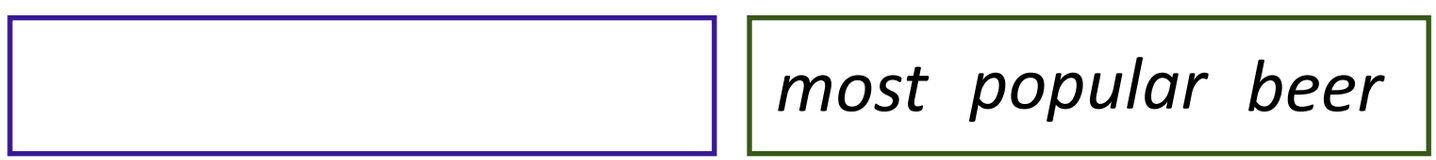
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

left-branching:



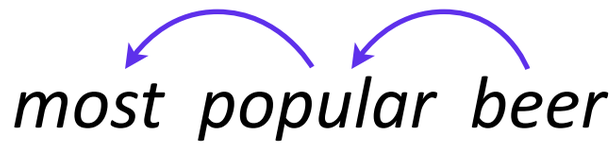


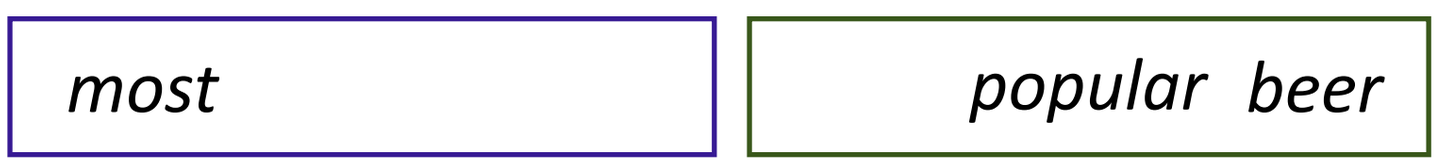
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

left-branching:



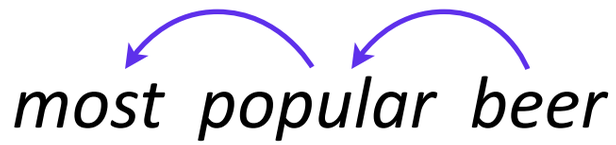


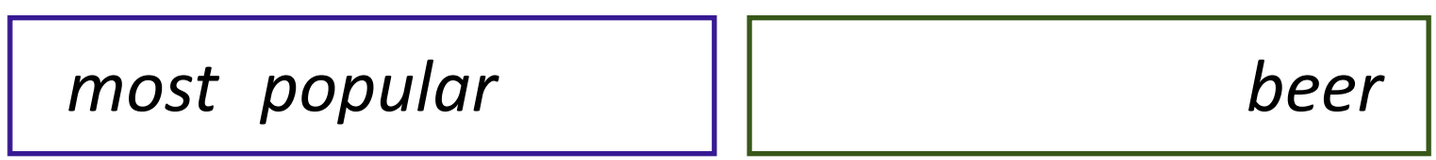
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

left-branching:



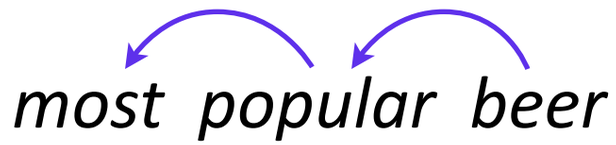


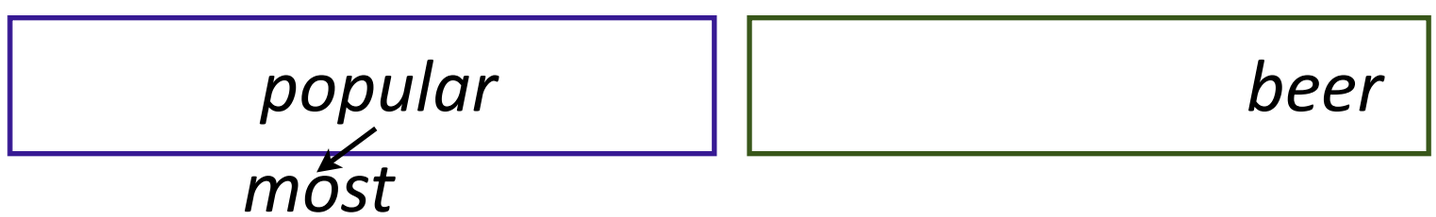
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

left-branching:



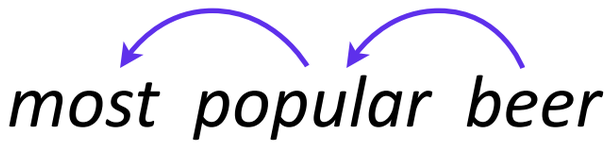


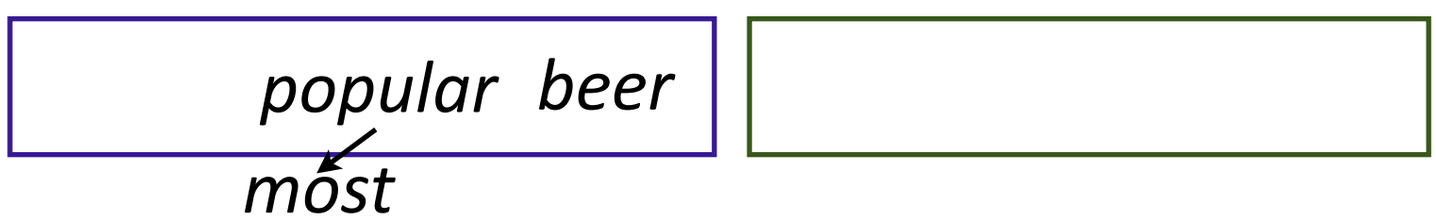
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

left-branching:



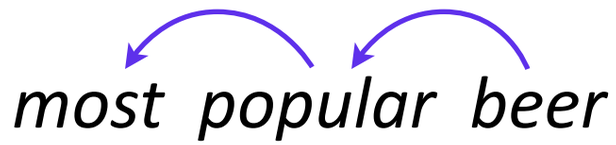


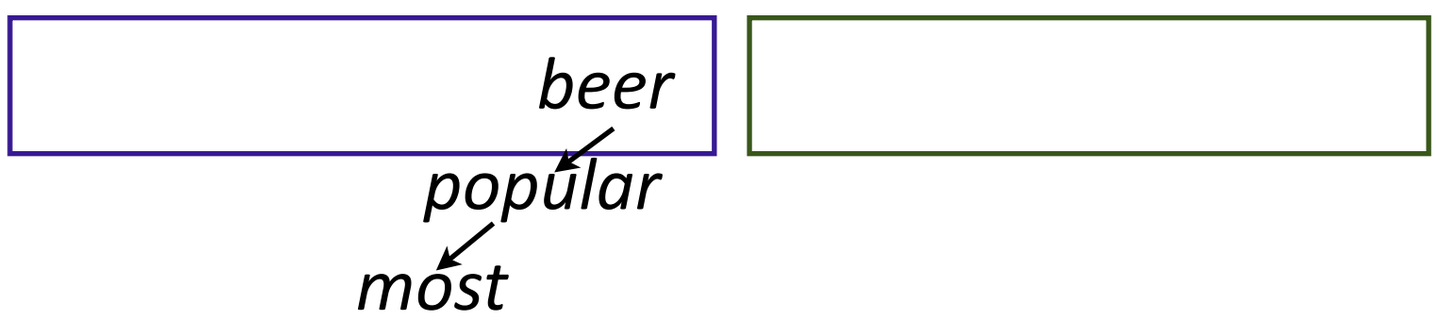
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

left-branching:



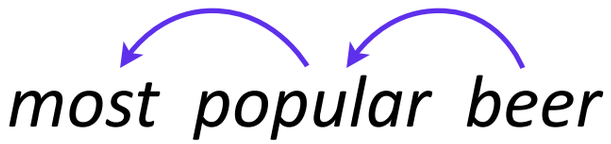


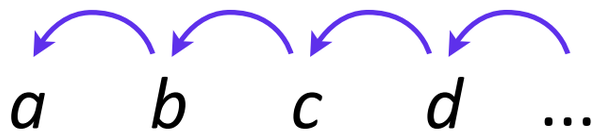
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

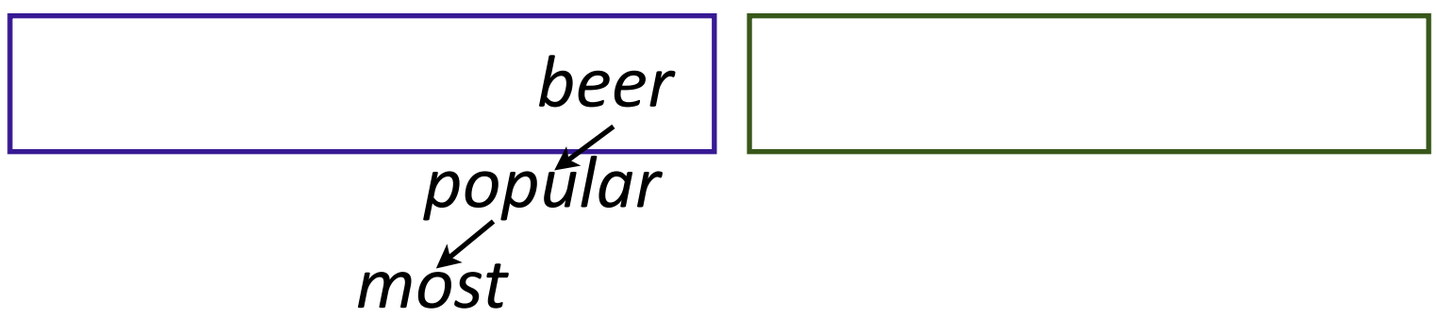
- Different memory cost for left- and right-branchings

left-branching:





Max memory cost is 2 (**constant**)

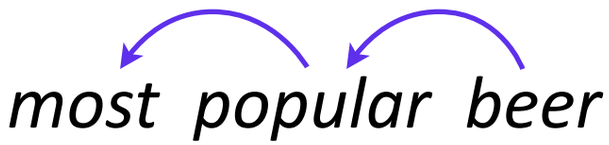


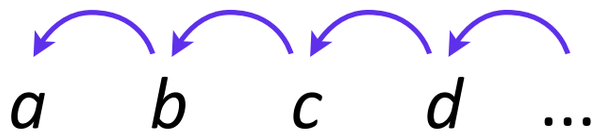
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

left-branching:





Max memory cost is 2 (**constant**)

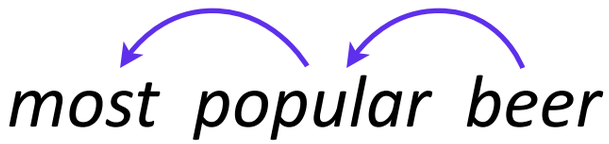
right-branching:

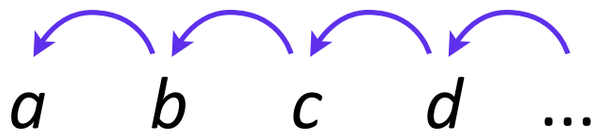
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

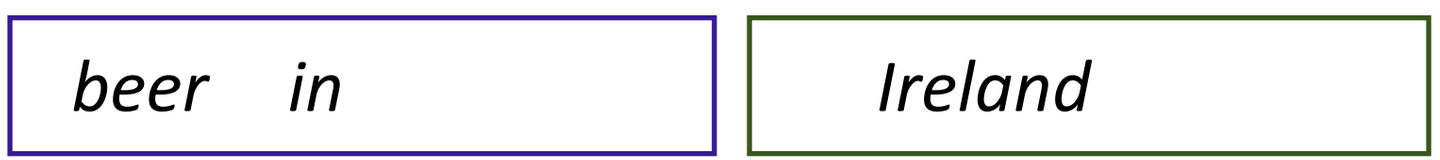
left-branching:





Max memory cost is 2 (**constant**)

right-branching:

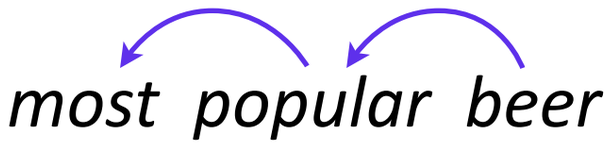


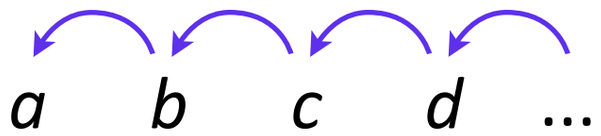
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

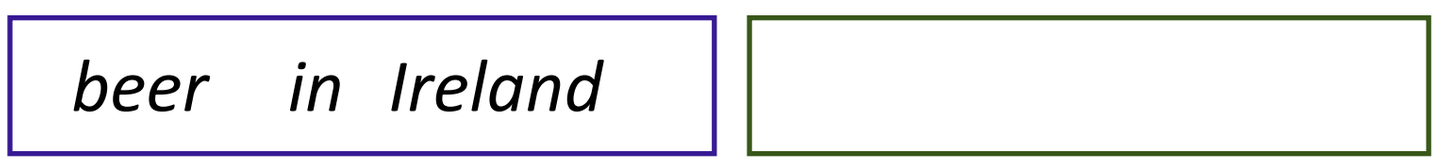
left-branching:





Max memory cost is 2 (**constant**)

right-branching:

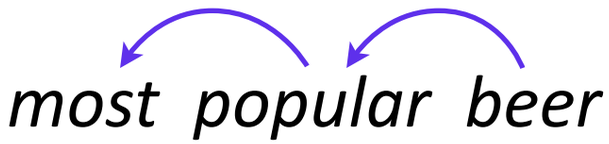


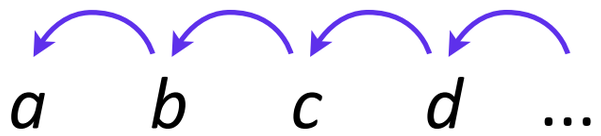
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

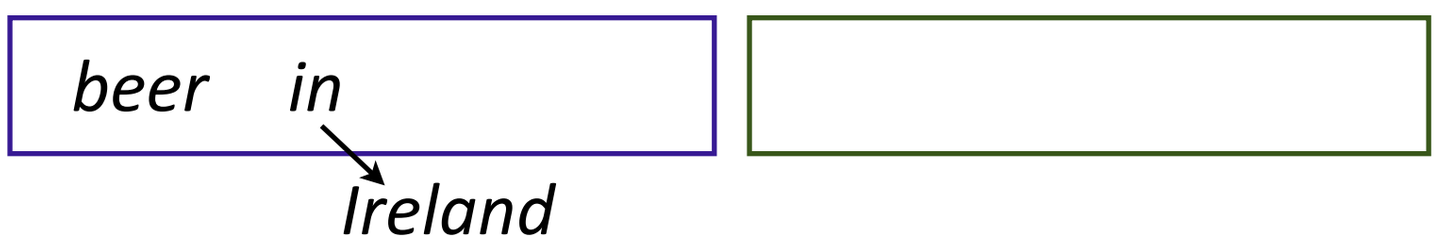
left-branching:





Max memory cost is 2 (**constant**)

right-branching:

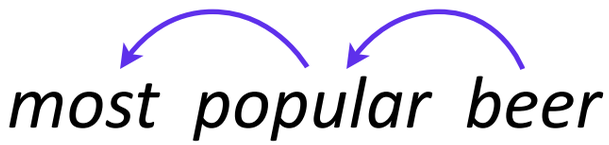


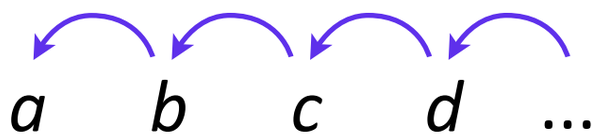
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

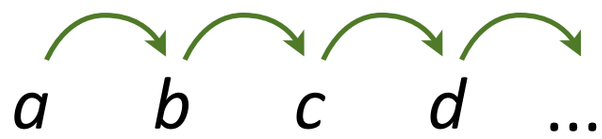
left-branching:



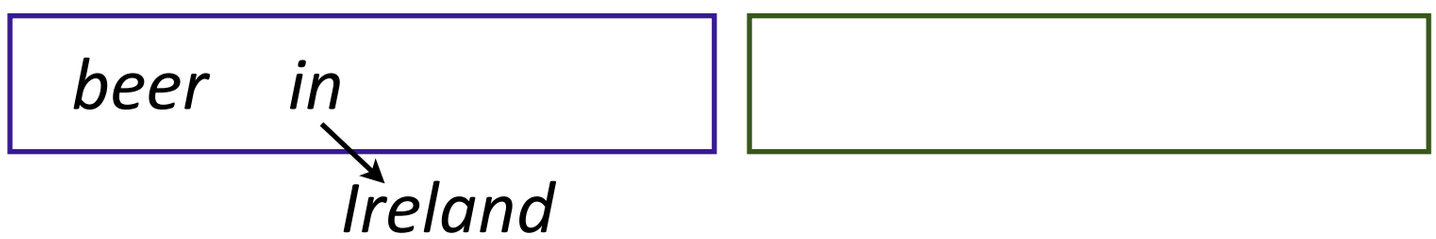


Max memory cost is 2 (**constant**)

right-branching:



Memory cost increases **linearly**!

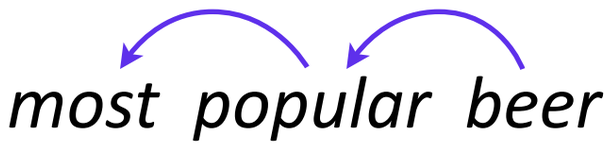


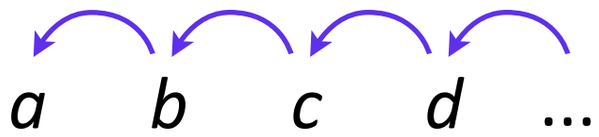
Arc-standard: Memory property

Arc-standard is not a desirable transition system:

- Different memory cost for left- and right-branchings

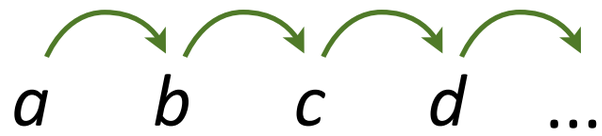
left-branching:





Max memory cost is 2 (**constant**)

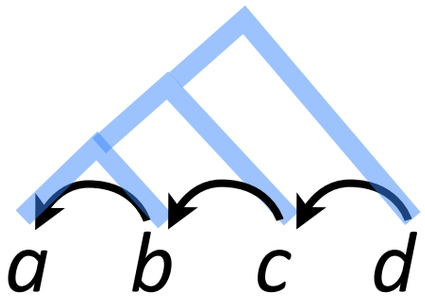
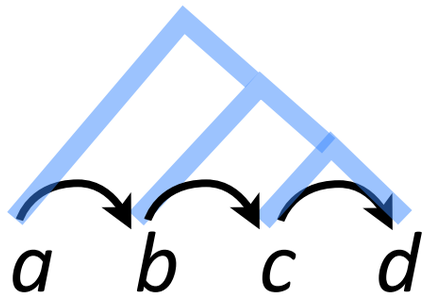
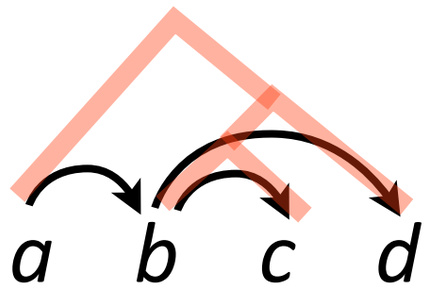
right-branching:



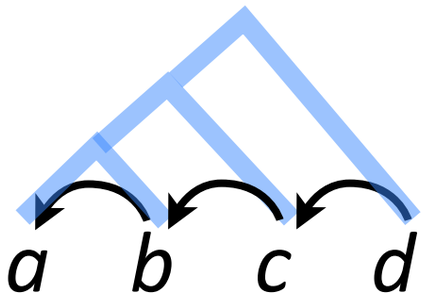
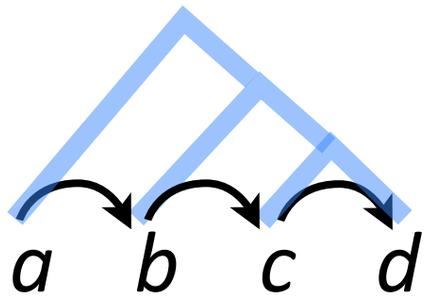
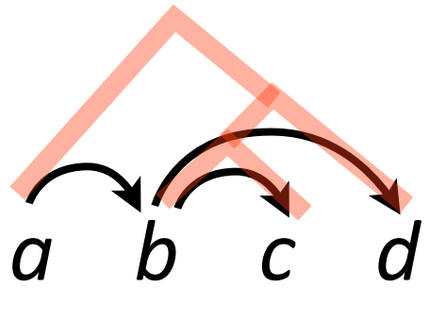
Memory cost increases **linearly**!

- This is because arc-standard is a **bottom-up** algorithm
- For center-embedding: incurs linear cost

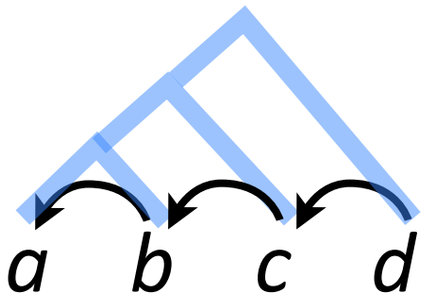
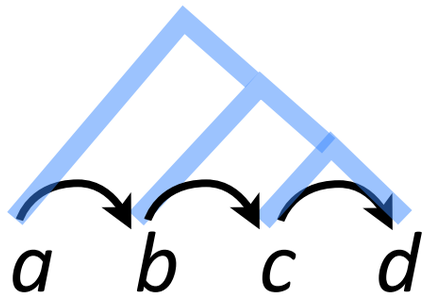
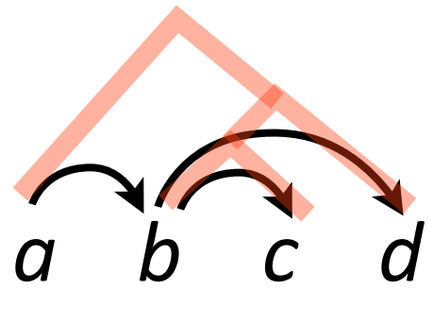
Memory cost comparison

			
Arc-standard	O(1)	O(n)	O(n)
Arc-eager/ Spine	O(1)	O(1~n)	O(1~n)
Human	**O(1)**	**O(1)**	**O(n)**

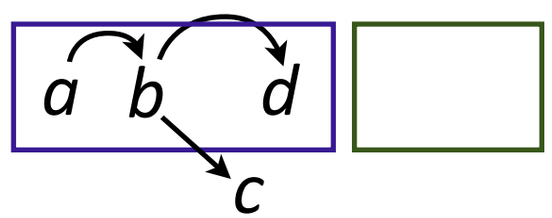
Memory cost comparison

			
Arc-standard	O(1)	O(n)	O(n)
Arc-eager/ Spine	O(1)	O(1~n)	O(1~n)
Human	**O(1)**	**O(1)**	**O(n)**

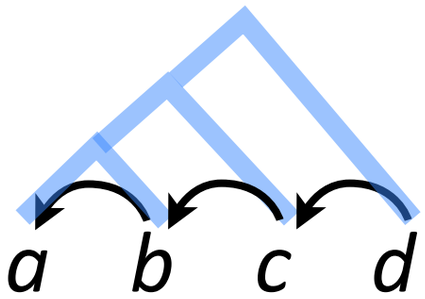
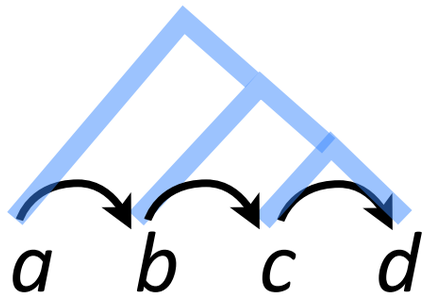
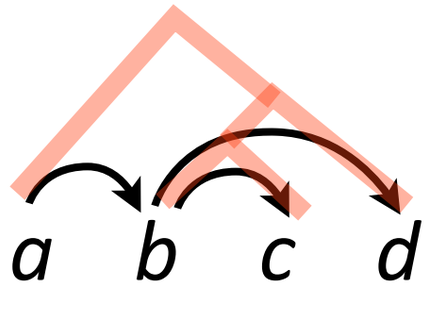
Memory cost comparison

			
Arc-standard	O(1)	O(n) ✗	O(n)
Arc-eager/ Spine	O(1)	O(1~n) ✗	O(1~n) ✗
Human	**O(1)**	**O(1)**	**O(n)**

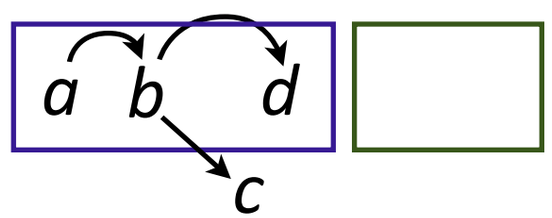
- Arc-eager: incurs no cost as long as all arcs are left-to-right



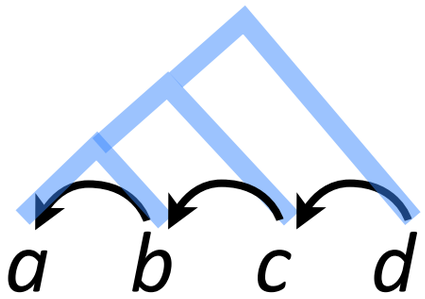
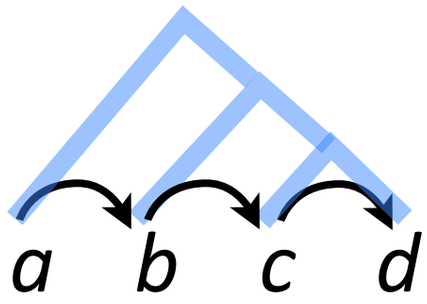
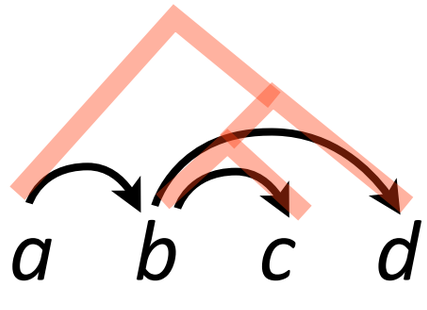
Memory cost comparison

			
Arc-standard	O(1)	O(n) ✗	O(n)
Arc-eager/ Spine	O(1)	O(1~n) ✗	O(1~n) ✗
Human	**O(1)**	**O(1)**	**O(n)**

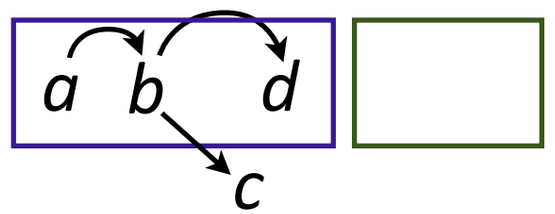
- Arc-eager: incurs no cost as long as all arcs are left-to-right
- Existing systems don't have desired property
  ⇒ We will pursue left-corner transition system



Memory cost comparison

			
Arc-standard	O(1)	O(n)	O(n)
Arc-eager/ Spine	O(1)	O(1~n)	O(1~n)
Left-corner	**O(1)**	**O(1)**	**O(n)**

- Arc-eager: incurs no cost as long as all arcs are left-to-right
- Existing systems don't have desired property
  ⇒ We will pursue left-corner transition system



Outline

Setup

- Psycholinguistic observation
- Left-corner constituency parsing
- Memory cost of existing transition-based dependency parsers

Left-corner dependency parsing

- Dummy node
- Parse example and action definition

Experiments

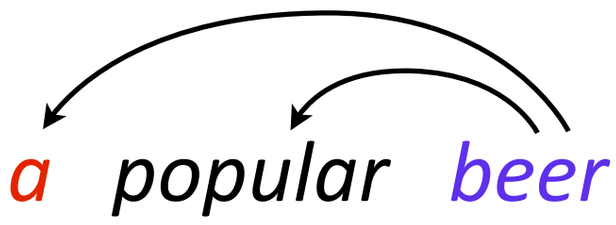
- Memory cost during oracle transitions
- Exploration of syntactic biases

Key idea: transition with prediction

Problem of arc-standard: for right-branchings...

Key idea: transition with prediction

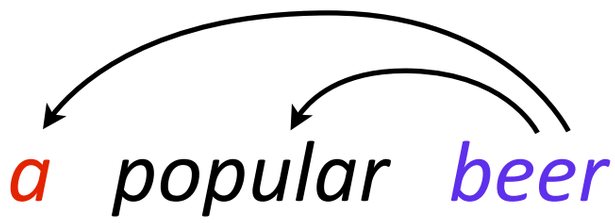
Problem of arc-standard: for right-branchings...



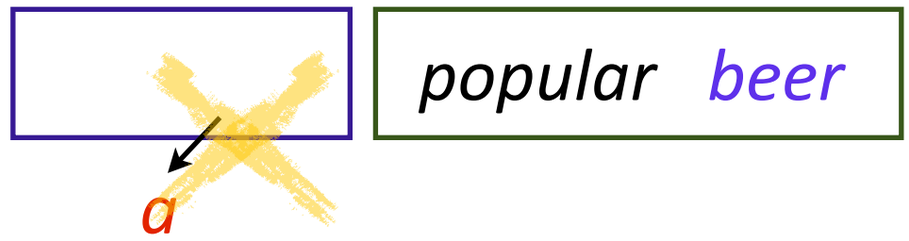
a cannot be reduced before *beer* is shifted

Key idea: transition with prediction

Problem of arc-standard: for right-branchings...

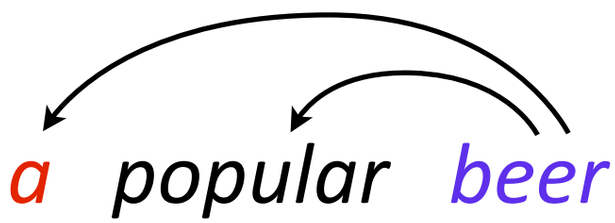


a cannot be reduced before *beer* is shifted

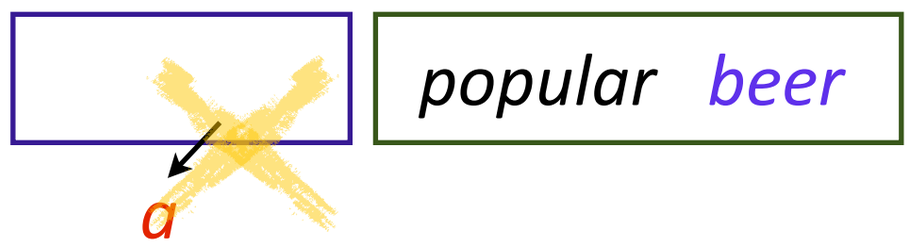


Key idea: transition with prediction

Problem of arc-standard: for right-branchings...



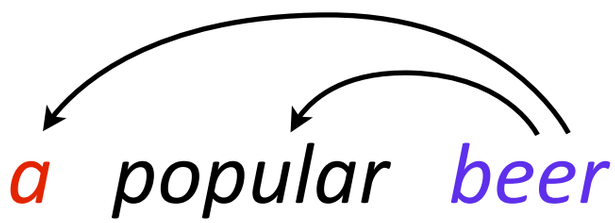
a cannot be reduced before *beer* is shifted



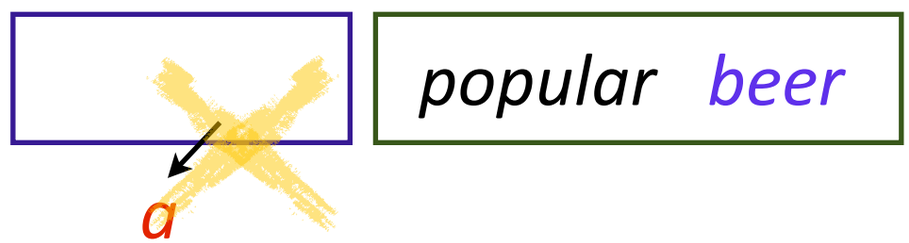
in cannot be reduced before collecting children

Key idea: transition with prediction

Problem of arc-standard: for right-branchings...



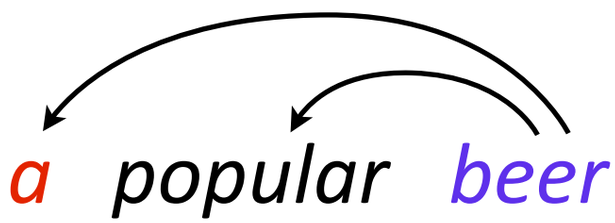
a cannot be reduced before *beer* is shifted



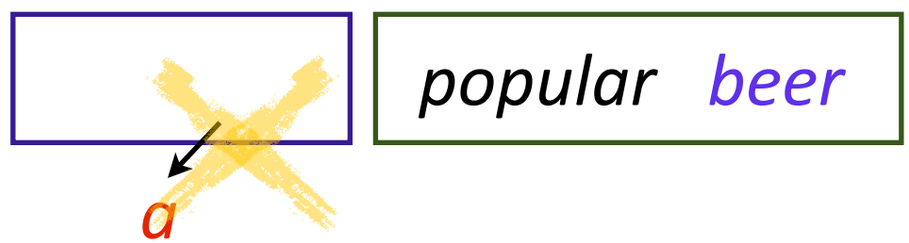
in cannot be reduced before collecting children

Key idea: transition with prediction

Problem of arc-standard: for right-branchings...



a cannot be reduced before *beer* is shifted



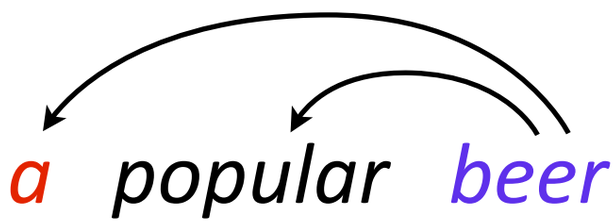
in cannot be reduced before collecting children

Solution: right side of subtree is predicted during transition

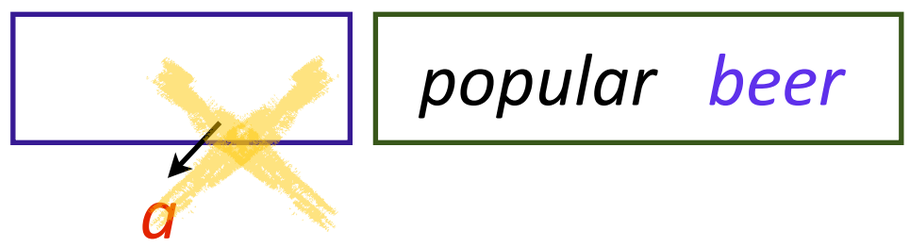
- Dummy node: represents predicted part in a subtree

Key idea: transition with prediction

Problem of arc-standard: for right-branchings...



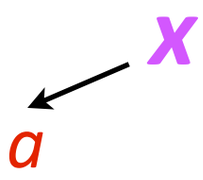
a cannot be reduced before *beer* is shifted



in cannot be reduced before collecting children

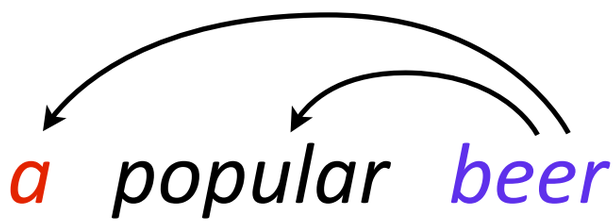
Solution: right side of subtree is predicted during transition

- Dummy node: represents predicted part in a subtree

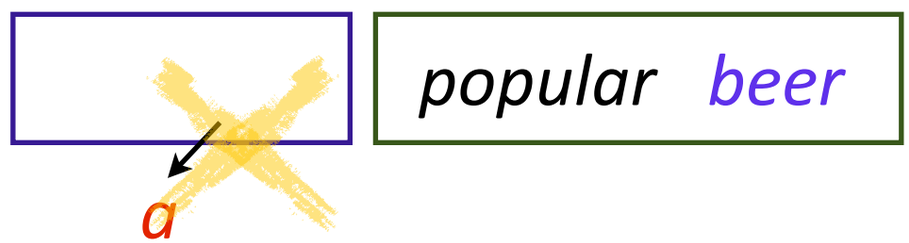


Key idea: transition with prediction

Problem of arc-standard: for right-branchings...



a cannot be reduced before *beer* is shifted



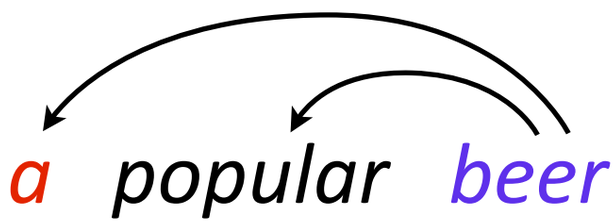
in cannot be reduced before collecting children

Solution: right side of subtree is predicted during transition

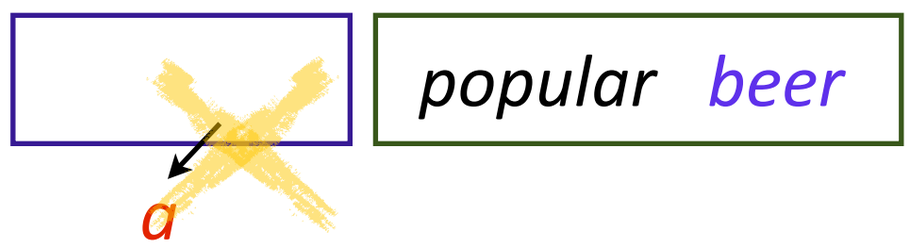
- Dummy node: represents predicted part in a subtree

Key idea: transition with prediction

Problem of arc-standard: for right-branchings...



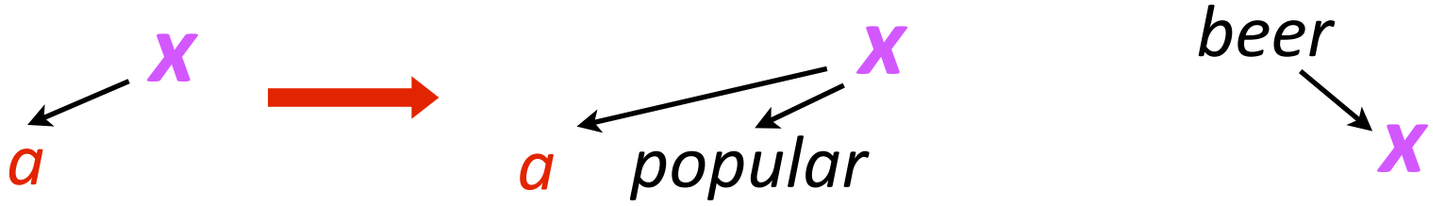
a cannot be reduced before *beer* is shifted



in cannot be reduced before collecting children

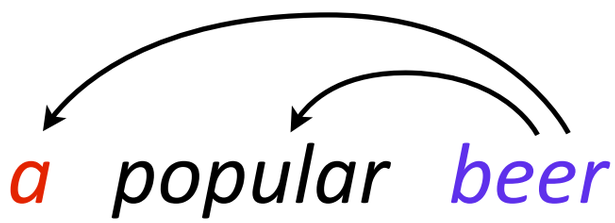
Solution: right side of subtree is predicted during transition

- Dummy node: represents predicted part in a subtree

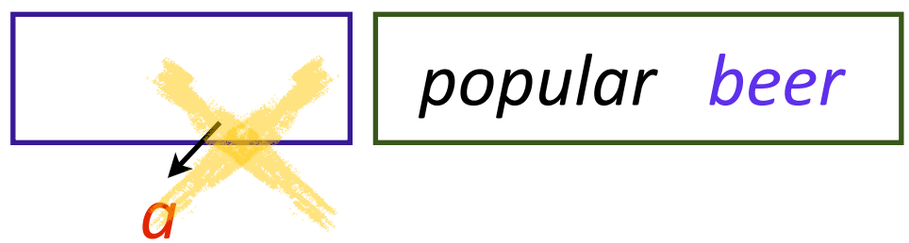


Key idea: transition with prediction

Problem of arc-standard: for right-branchings...



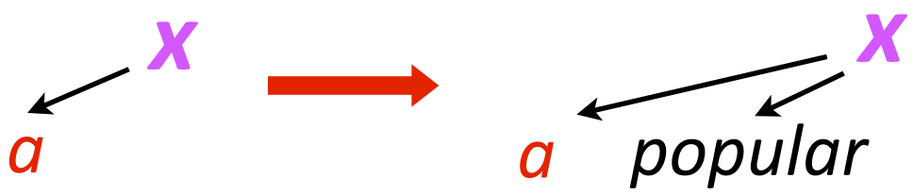
a cannot be reduced before *beer* is shifted

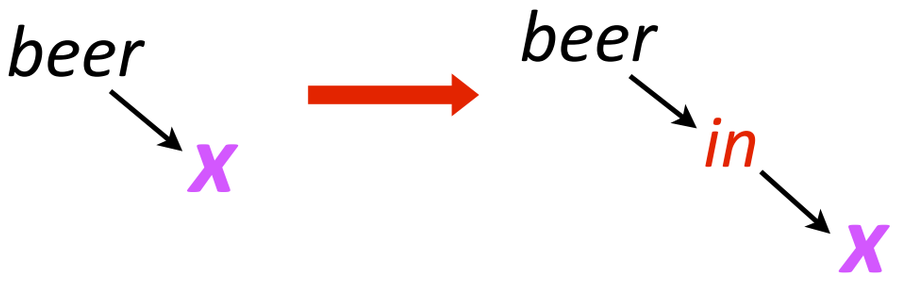


in cannot be reduced before collecting children

Solution: right side of subtree is predicted during transition

- Dummy node: represents predicted part in a subtree





Parsing example

Stack

Buffer

a popular beer in Ireland

Parsing example

Stack

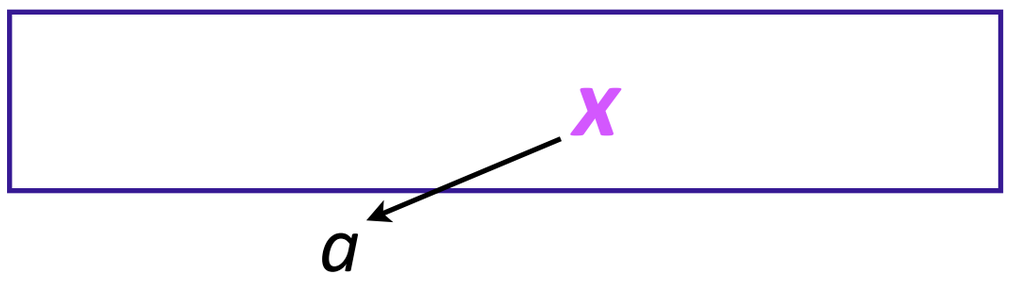
a

Buffer

popular beer in Ireland

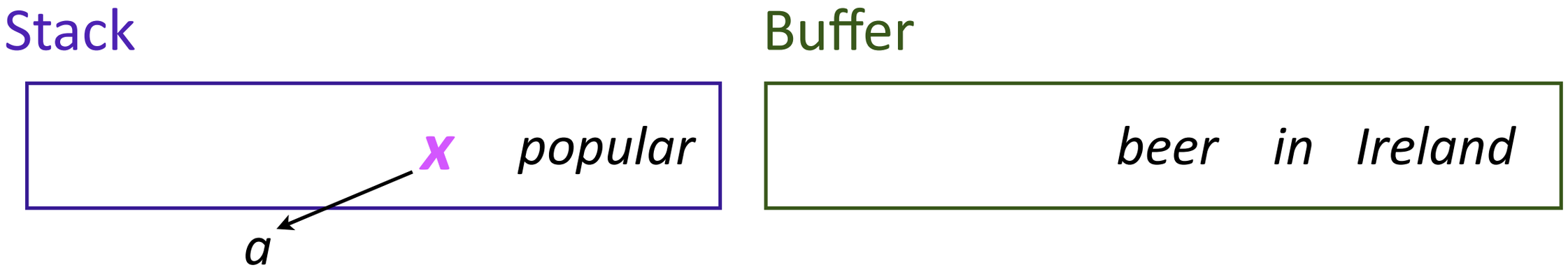
Parsing example

Stack



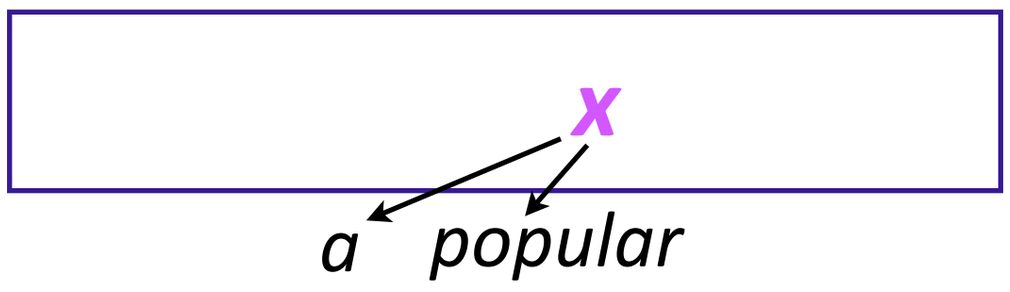
Buffer

Parsing example



Parsing example

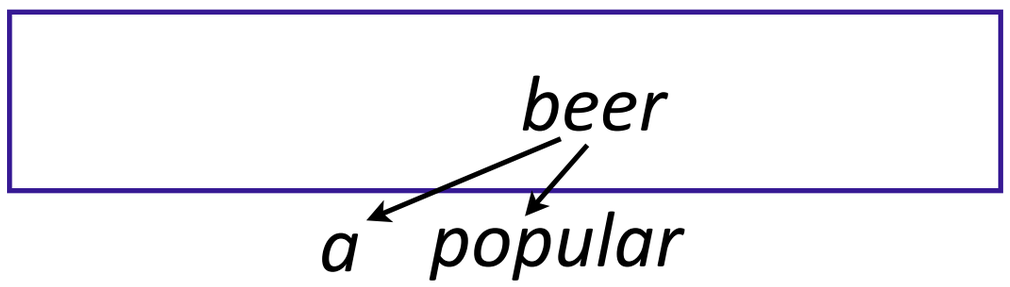
Stack



Buffer

Parsing example

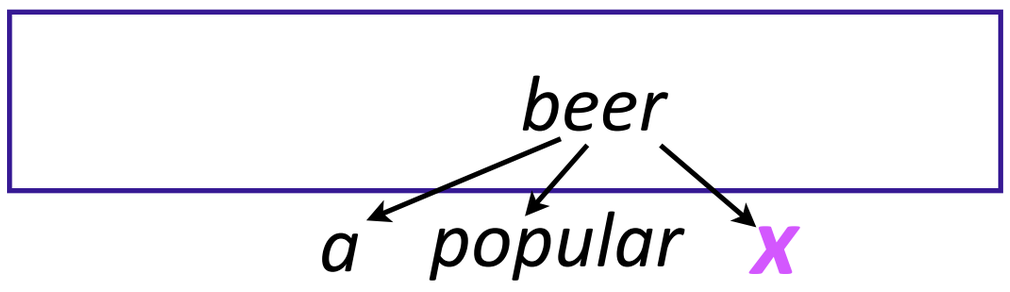
Stack



Buffer

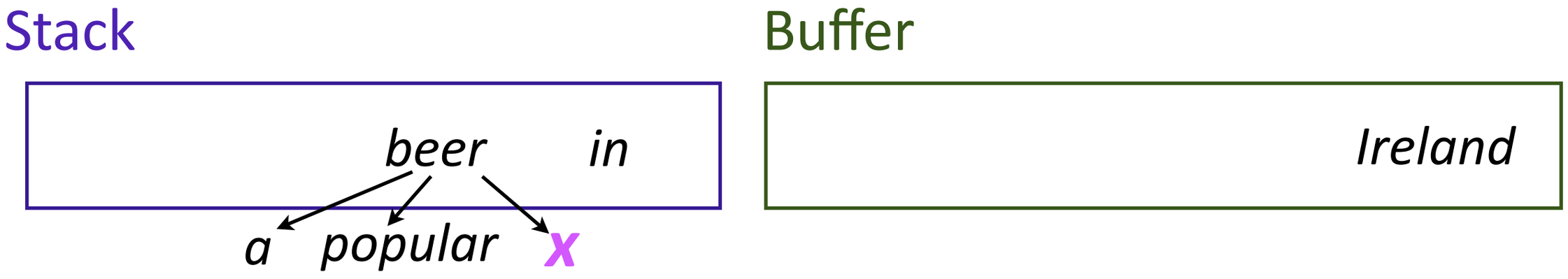
Parsing example

Stack



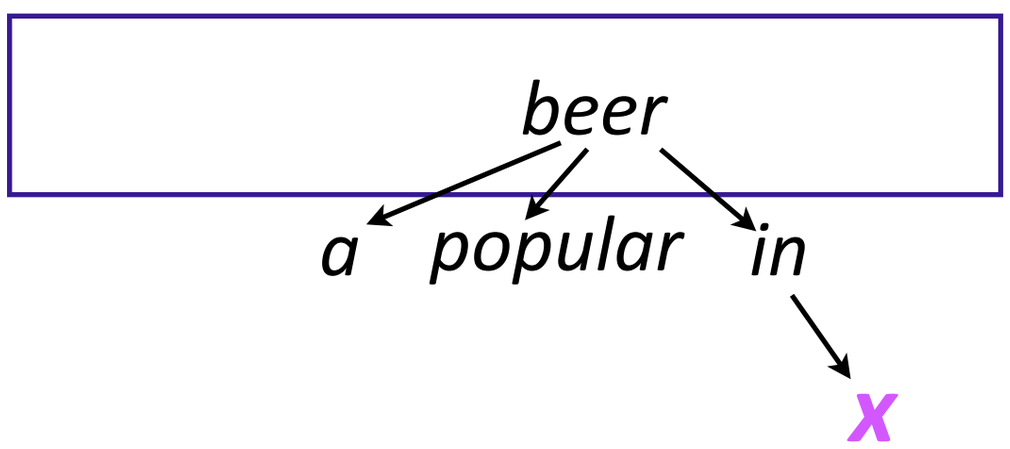
Buffer

Parsing example



Parsing example

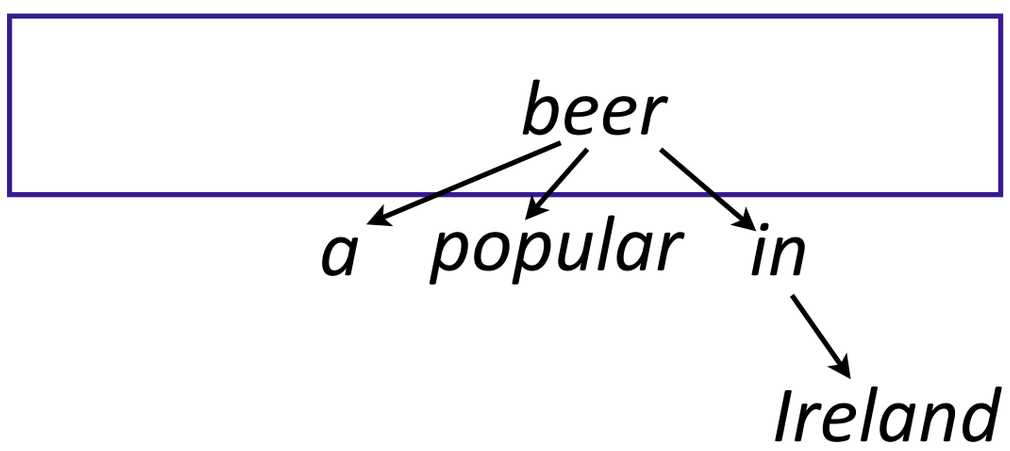
Stack



Buffer

Parsing example

Stack



Buffer

4 kinds of actions

SHIFT

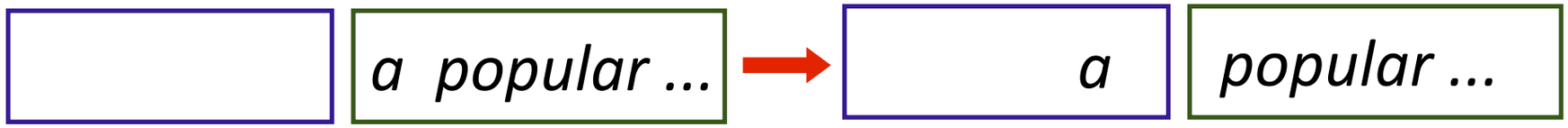
INSERT

PRED (**left**/right)

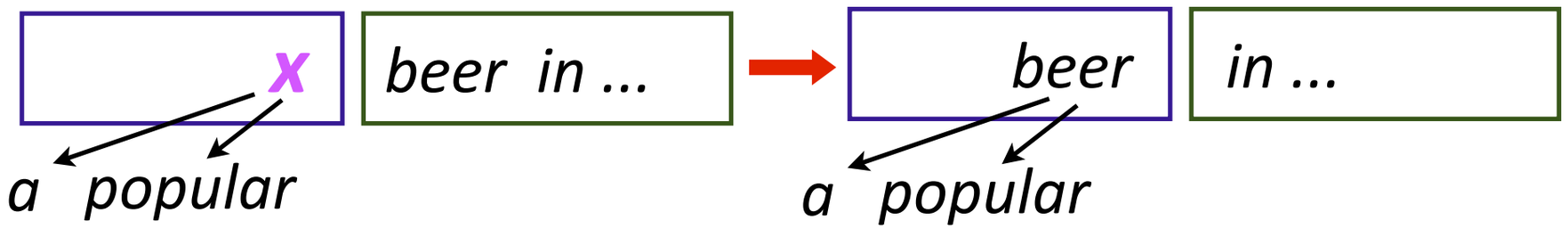
COMP (**left**/right)

4 kinds of actions

SHIFT



INSERT

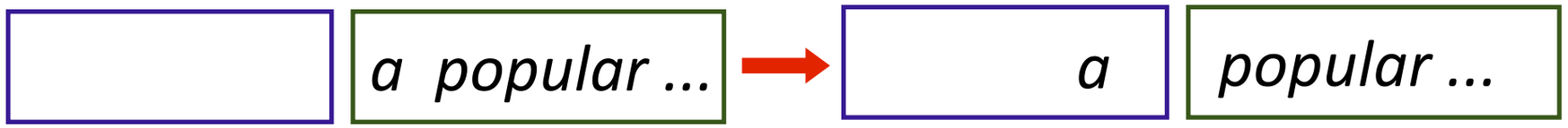


PRED (**left**/right)

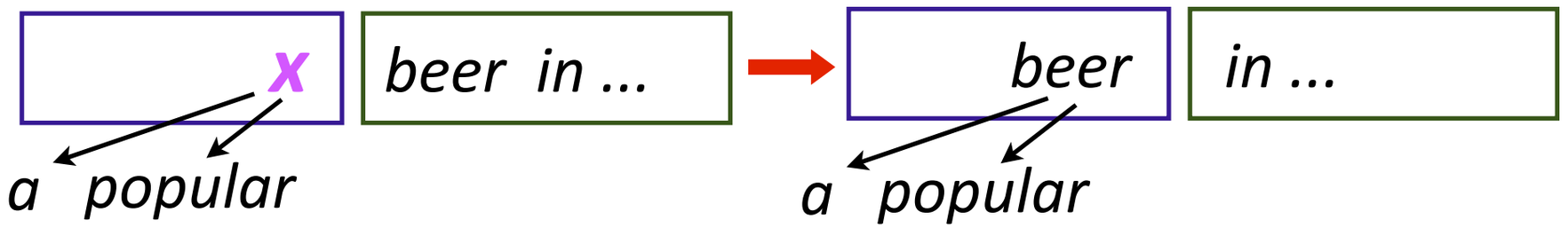
COMP (**left**/right)

4 kinds of actions

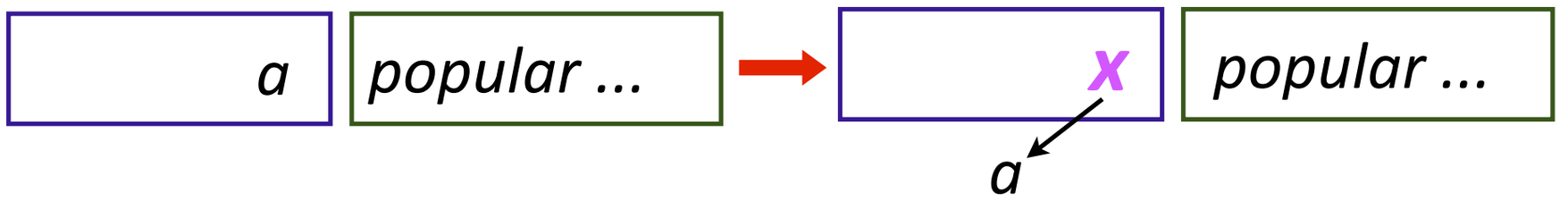
SHIFT



INSERT



PRED (**left**/right)

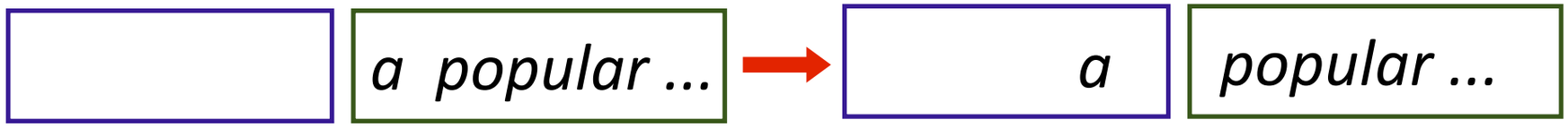


create a new dummy node

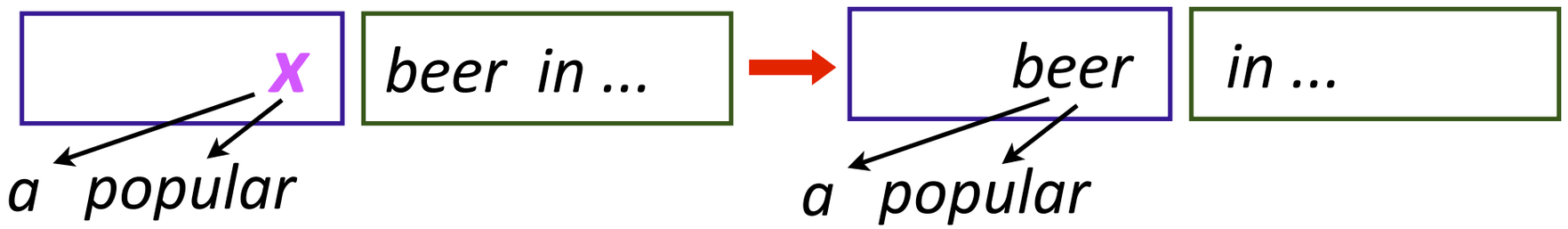
COMP (**left**/right)

4 kinds of actions

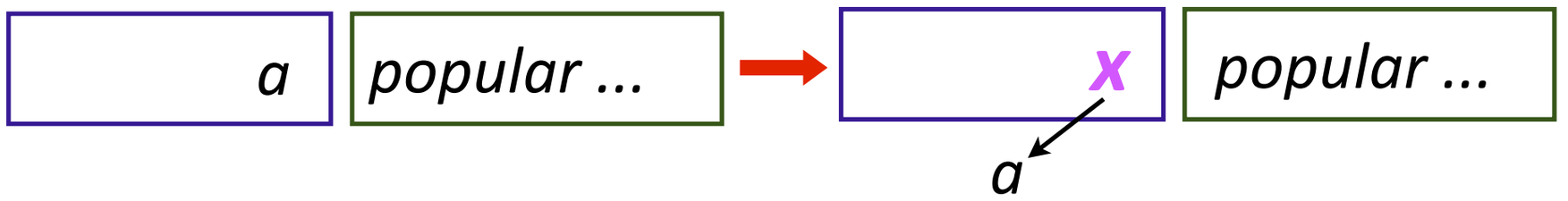
SHIFT



INSERT

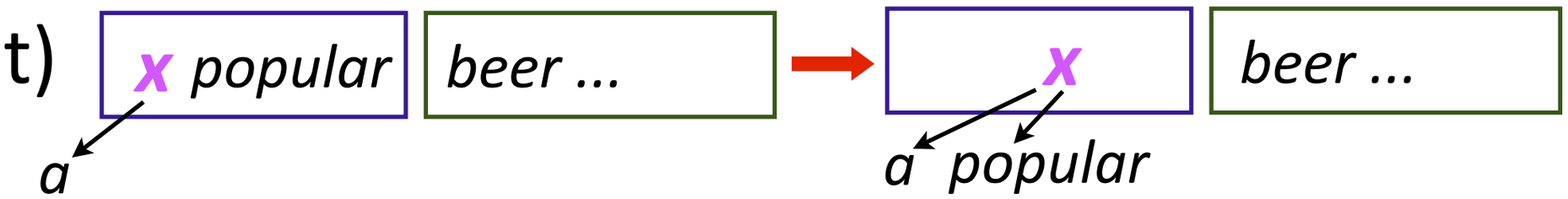


PRED (**left**/right)



create a new dummy node

COMP (**left**/right)

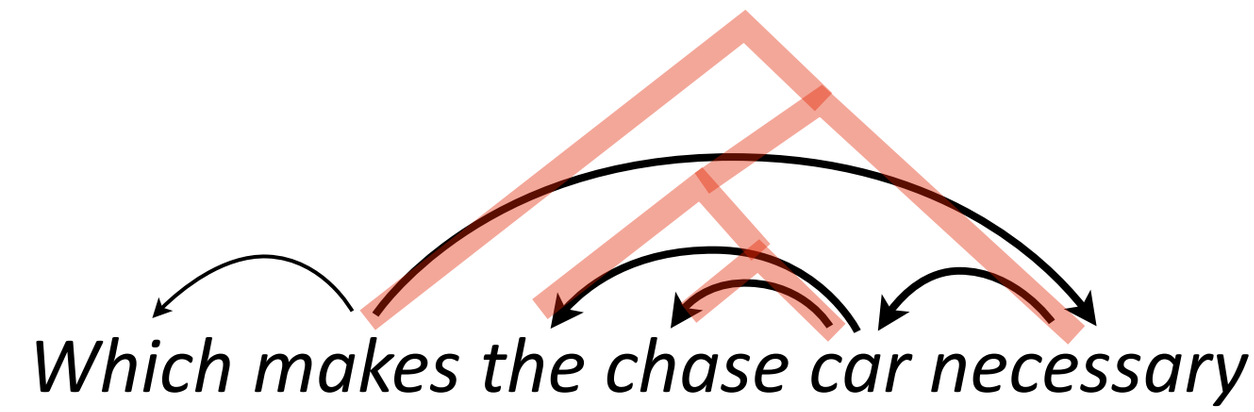


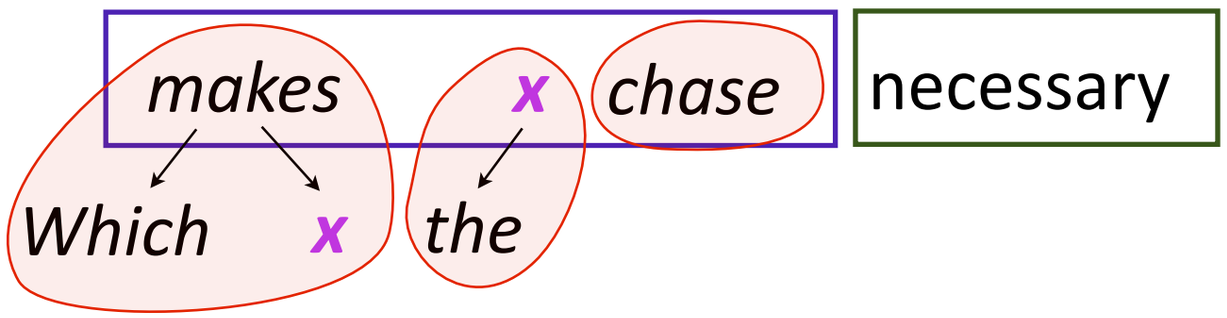
connect two trees via a dummy node

Characteristics

Characteristics

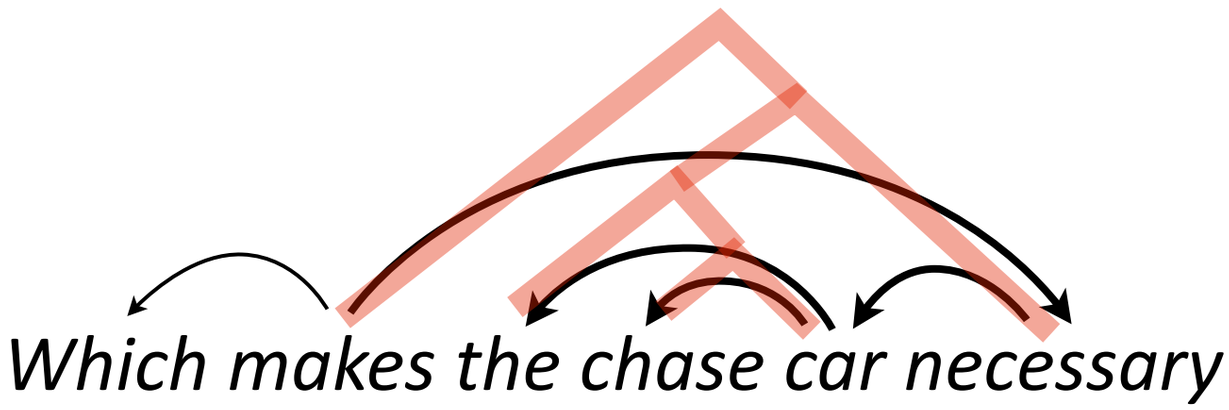
Non-constant cost only with center-embedding:

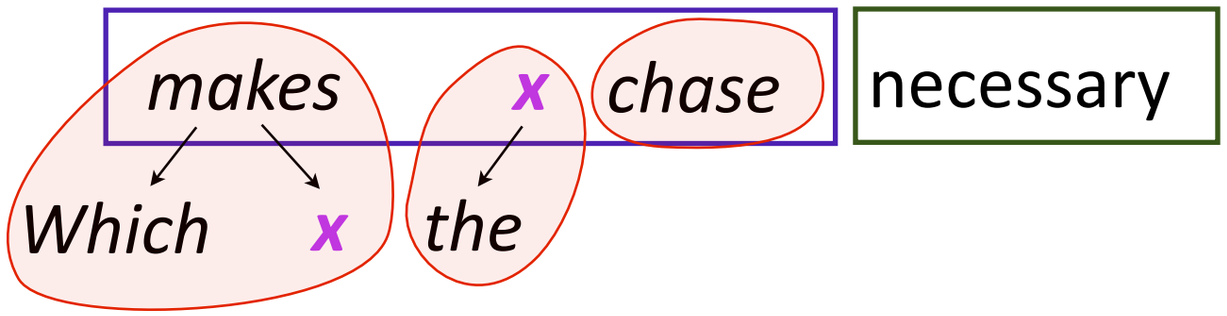




Characteristics

Non-constant cost only with center-embedding:



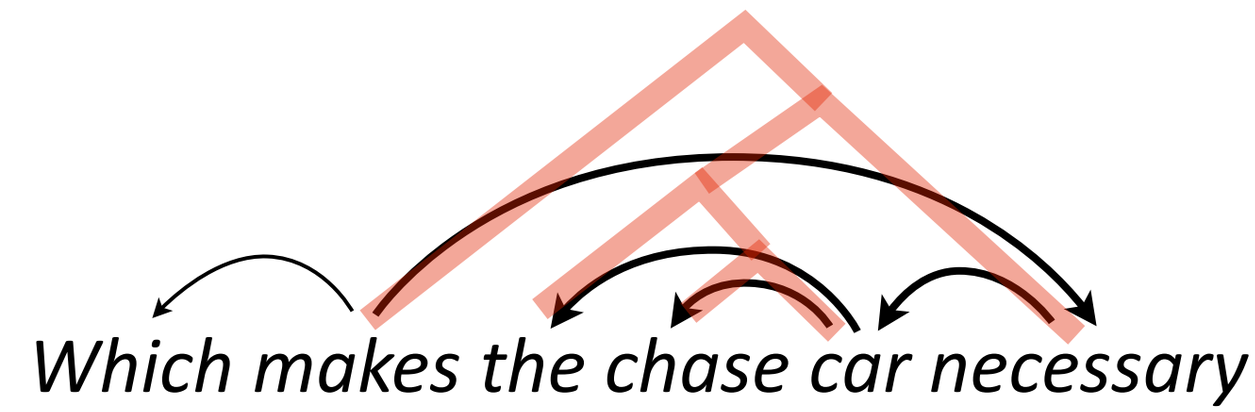


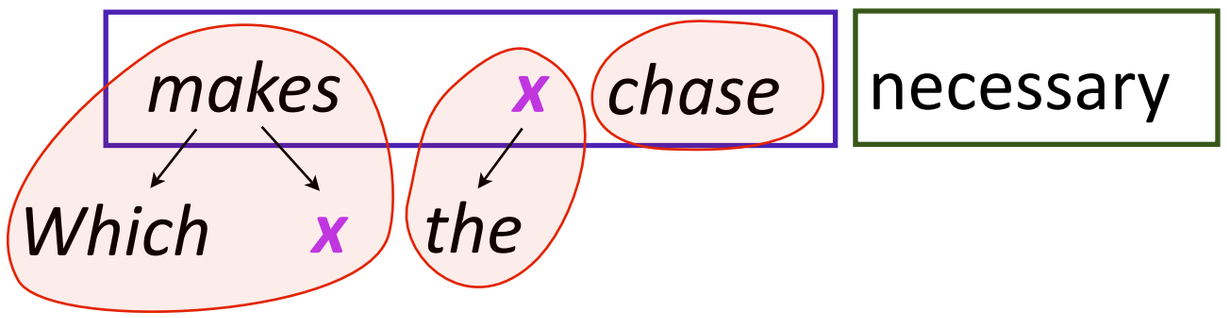
Deep connection to constituency left-corner parsing

- PRED can be interpreted as original **predict** with CNF:

Characteristics

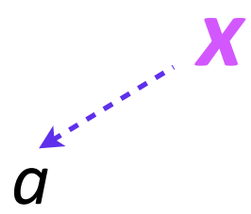
Non-constant cost only with center-embedding:





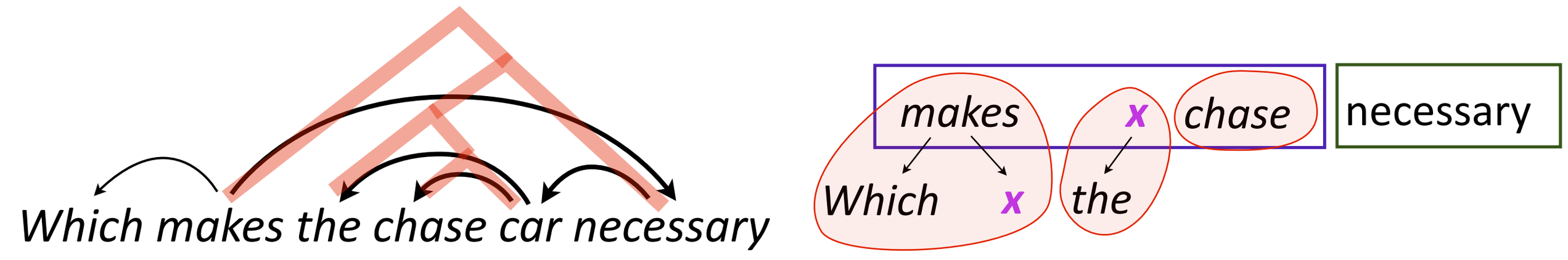
Deep connection to constituency left-corner parsing

- PRED can be interpreted as original **predict** with CNF:



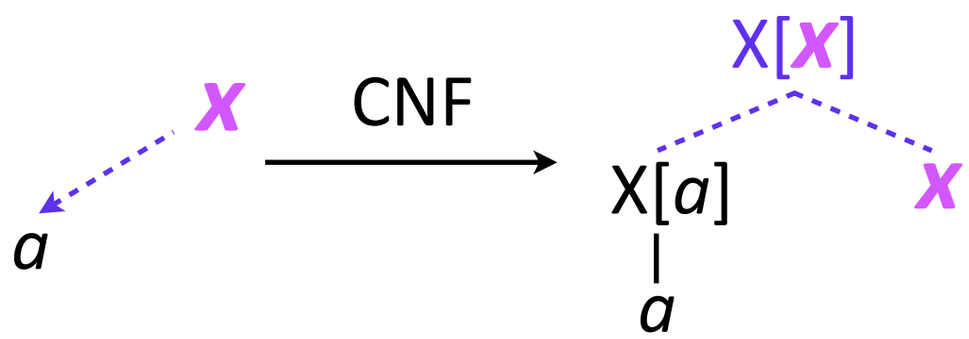
Characteristics

Non-constant cost only with center-embedding:



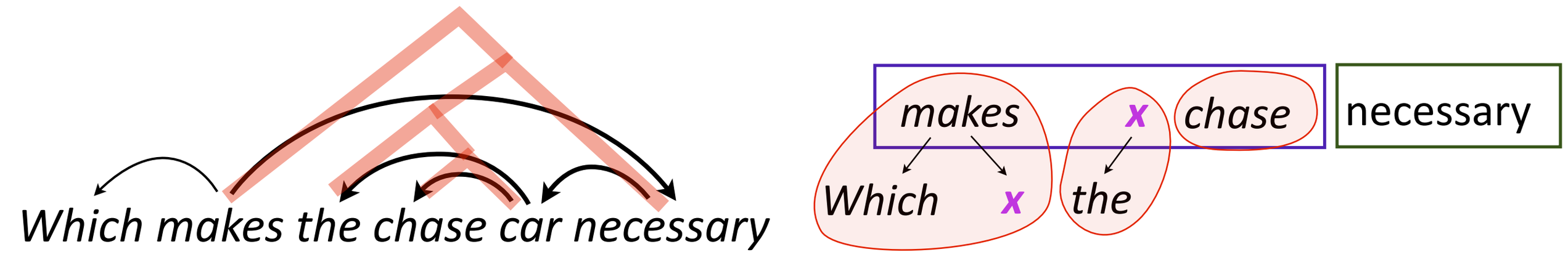
Deep connection to constituency left-corner parsing

- PRED can be interpreted as original **predict** with CNF:



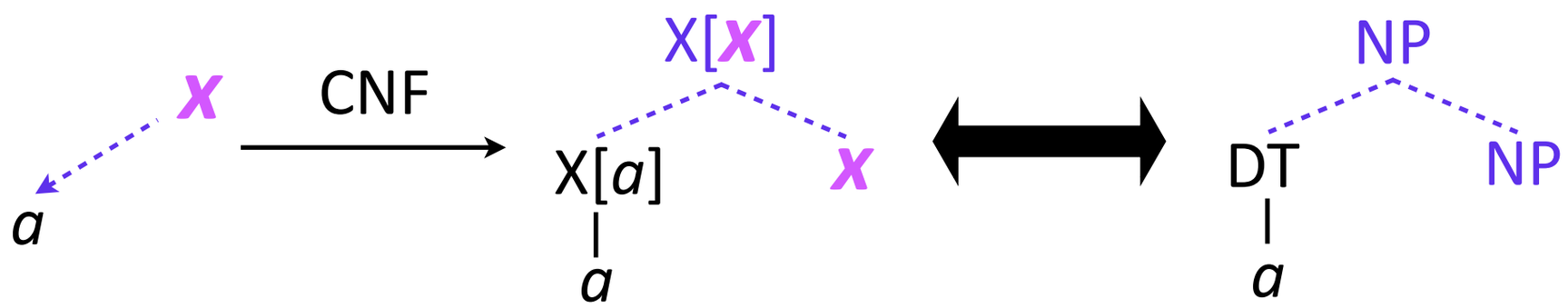
Characteristics

Non-constant cost only with center-embedding:



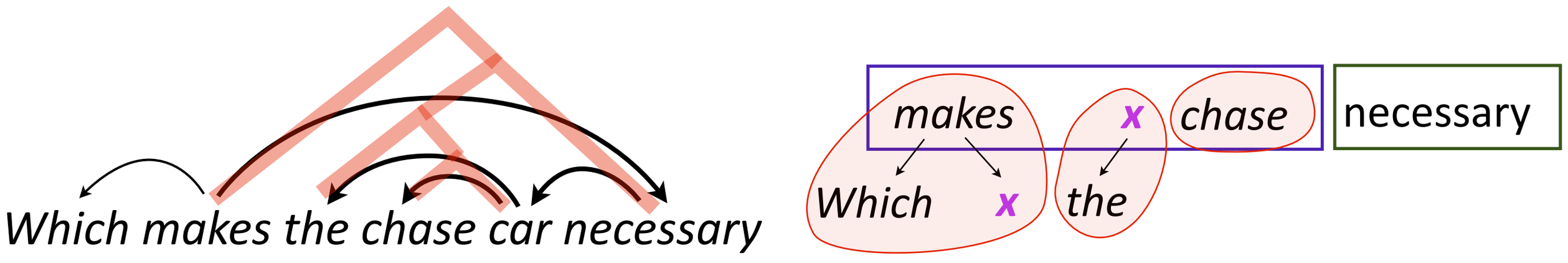
Deep connection to constituency left-corner parsing

- PRED can be interpreted as original **predict** with CNF:



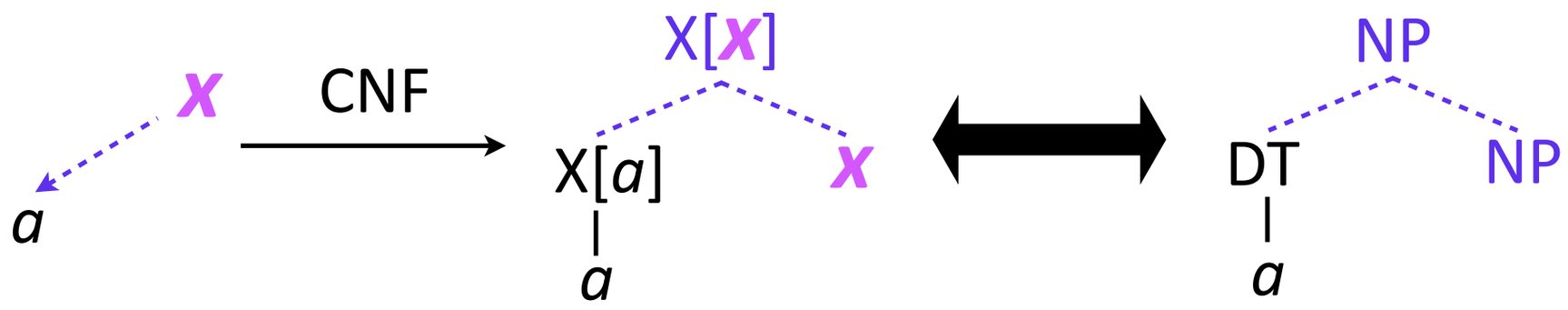
Characteristics

Non-constant cost only with center-embedding:



Deep connection to constituency left-corner parsing

- PRED can be interpreted as original **predict** with CNF:



- COMP is also connected to **composite** in [Resnik, 1992]

Outline

Setup

- Psycholinguistic observation
- Left-corner constituency parsing
- Memory cost of dependency parsers

Left-corner dependency parsing

- Key concept: dummy node
- Parse example and action definition

Experiments

- Memory cost during oracle transitions
- Exploration of syntactic biases

Experimental setting

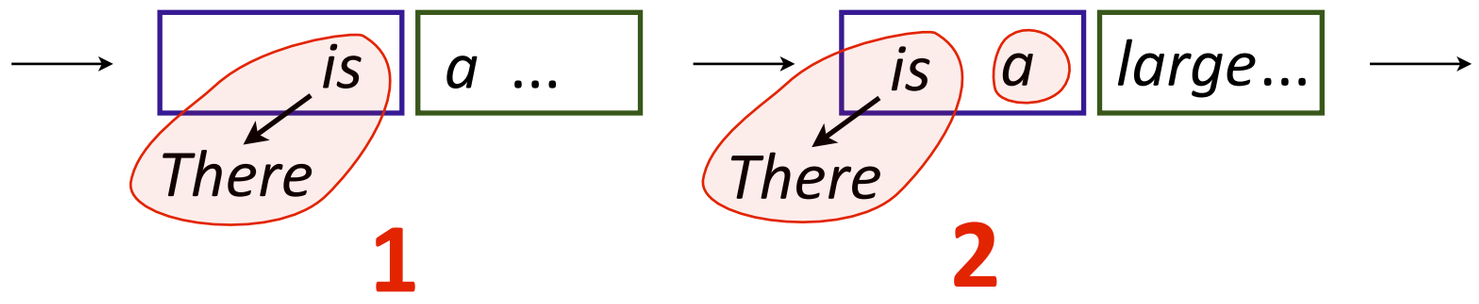
Experimental setting

Dataset: CoNLL 06/07 shared tasks (18 language treebanks)

Experimental setting

Dataset: CoNLL 06/07 shared tasks (18 language treebanks)

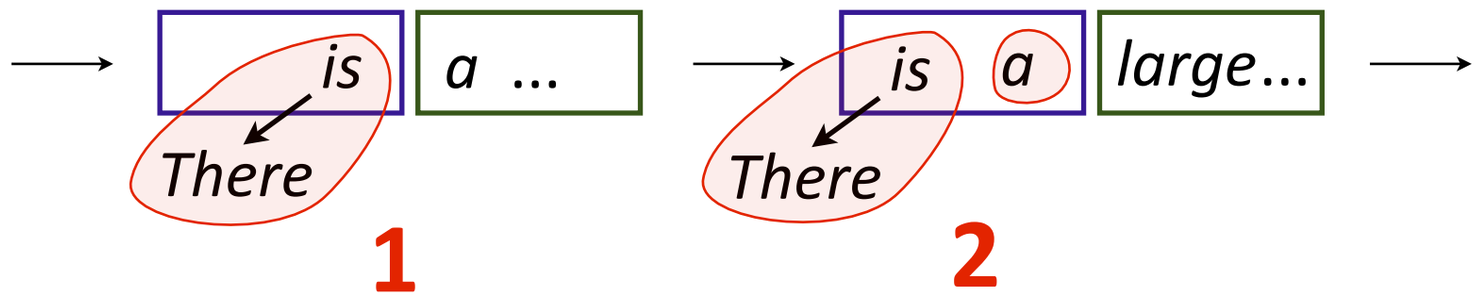
Evaluation: Required memory cost to recognize gold dependency trees in treebanks



Experimental setting

Dataset: CoNLL 06/07 shared tasks (18 language treebanks)

Evaluation: Required memory cost to recognize gold dependency trees in treebanks

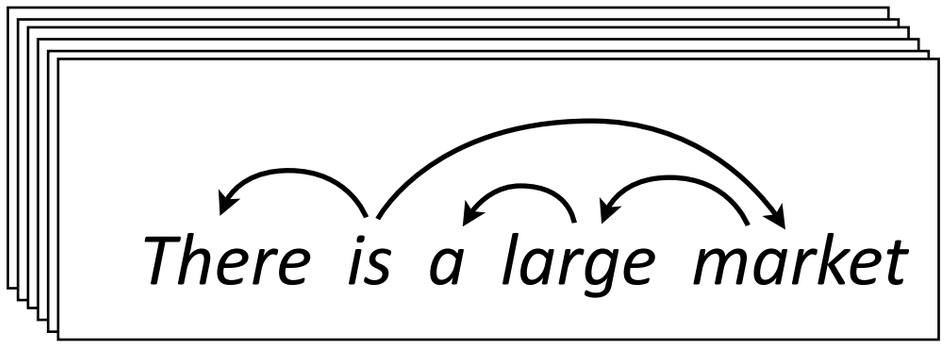


- Our interest: cross-linguistically cognitively plausible parser
- Such parser should use less memory for recognizing sentences

across diverse languages

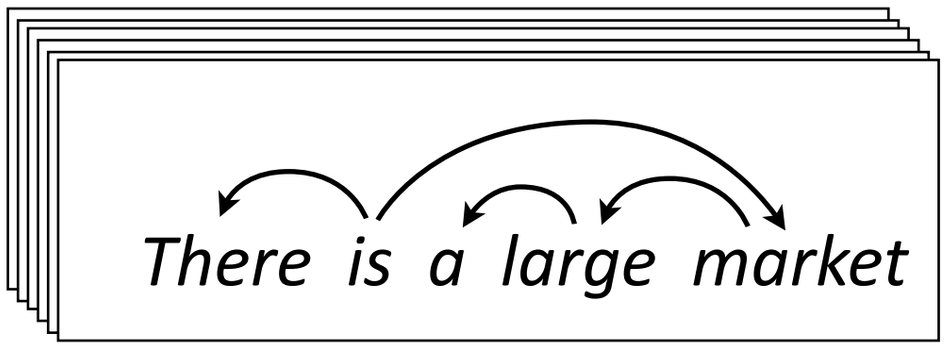
Example: English

Given the dependency treebank:



Example: English

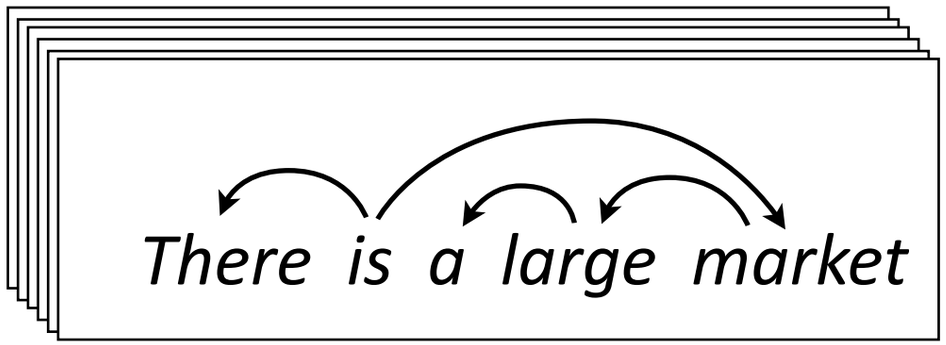
Given the dependency treebank:



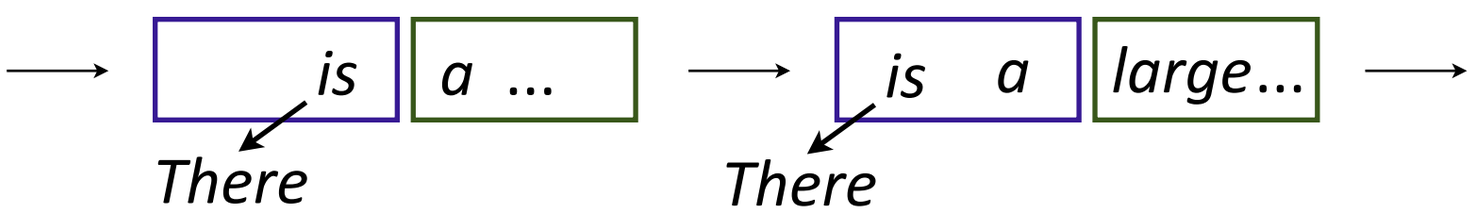
- Select a transition system (here arc-standard)

Example: English

Given the dependency treebank:

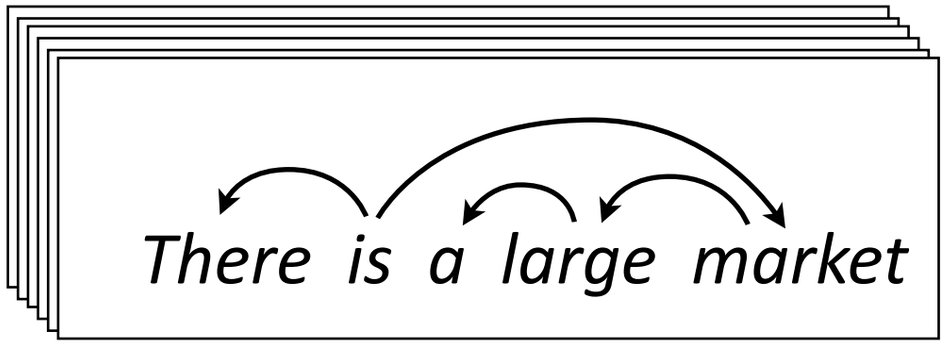


- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)

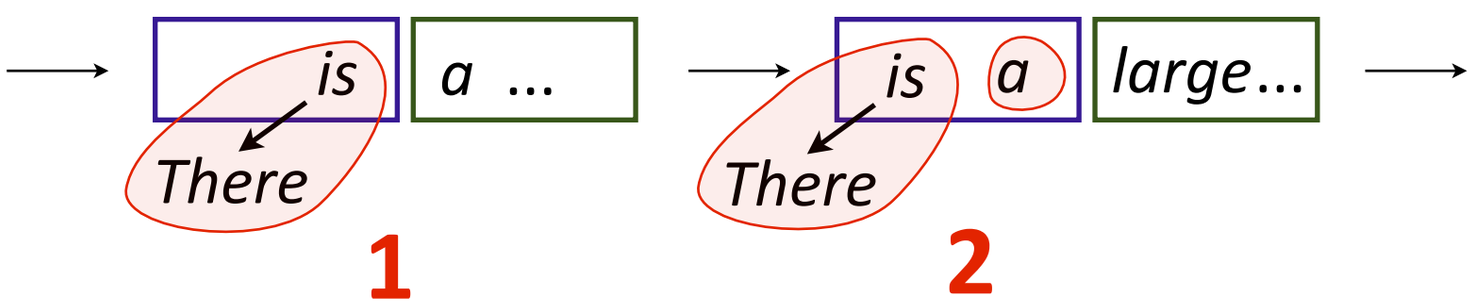


Example: English

Given the dependency treebank:

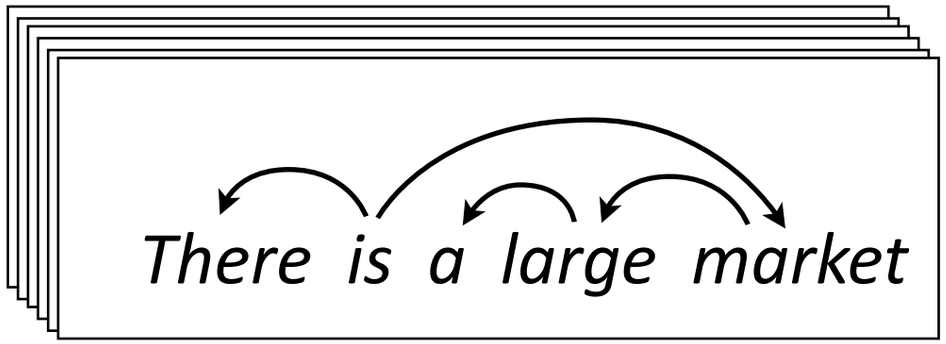


- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)



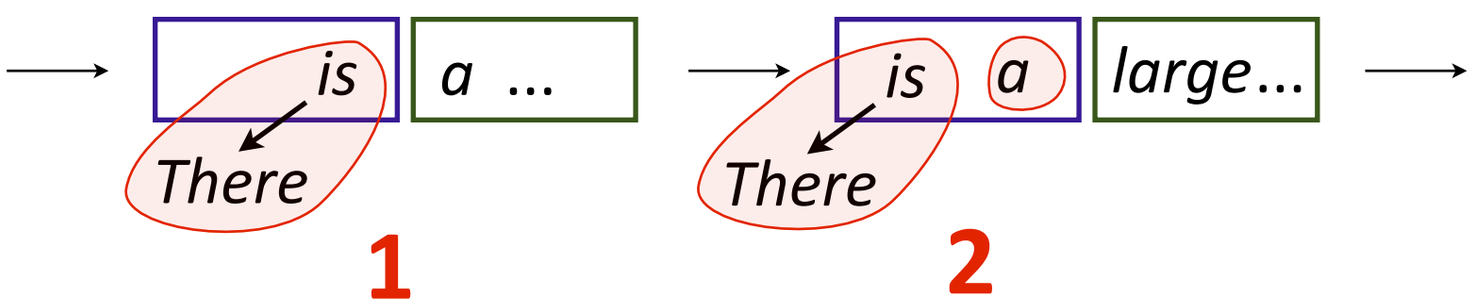
Example: English

Given the dependency treebank:



- Select a transition system
  (here arc-standard)

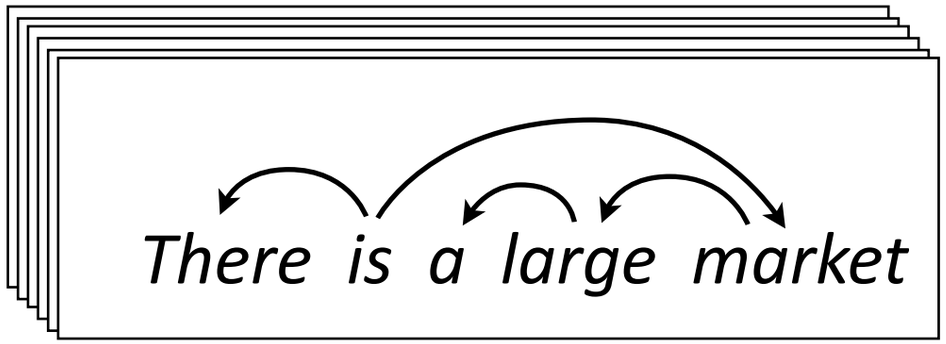
- Simulate oracle transitions
  (recover gold dependency)



How many times each memory cost happens?

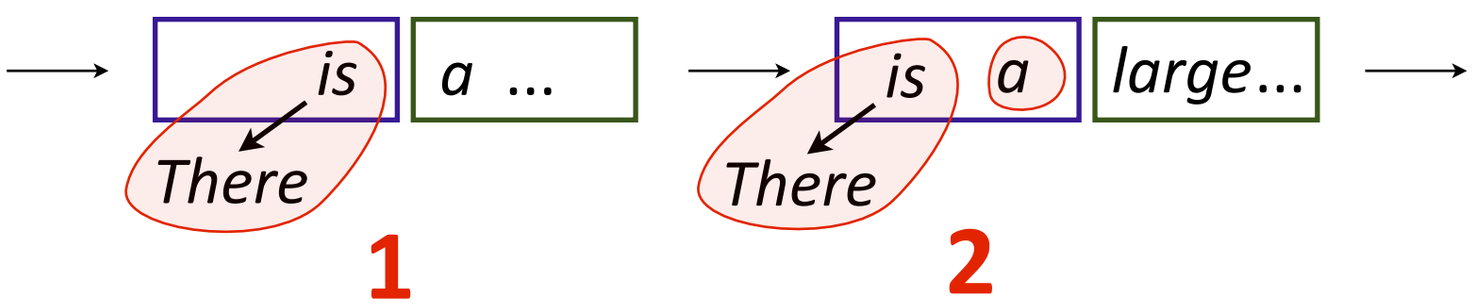
Example: English

Given the dependency treebank:



	# times
1	99,610
2	151,198
3	136,937
...	

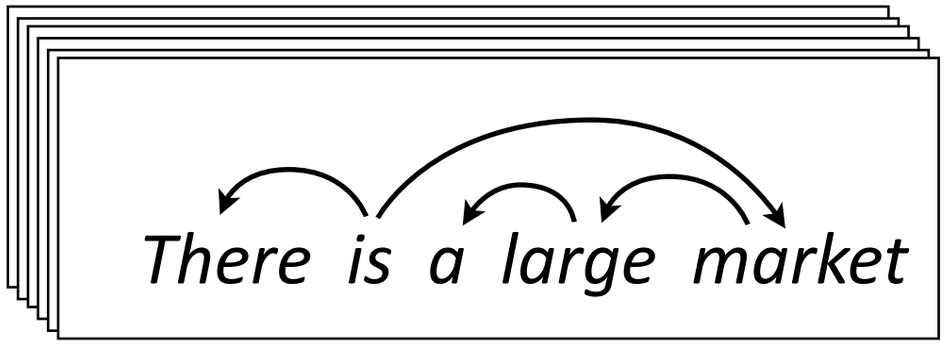
- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)



How many times each memory cost happens?

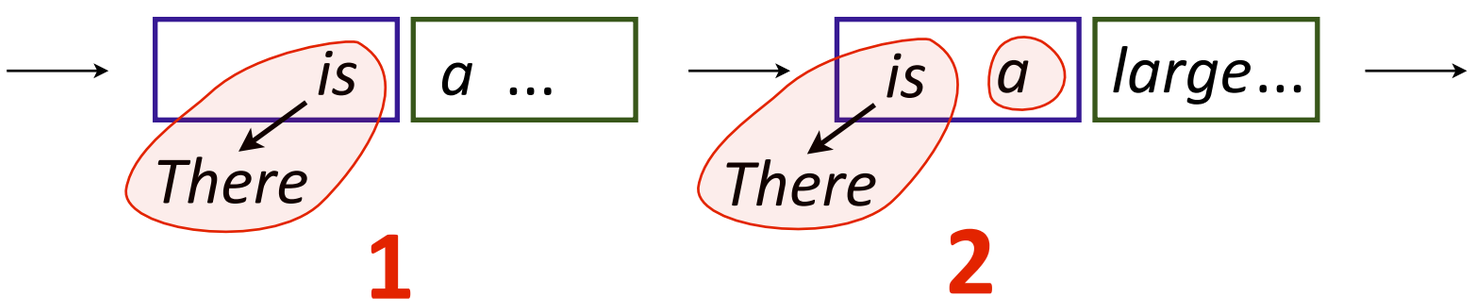
Example: English

Given the dependency treebank:



- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)

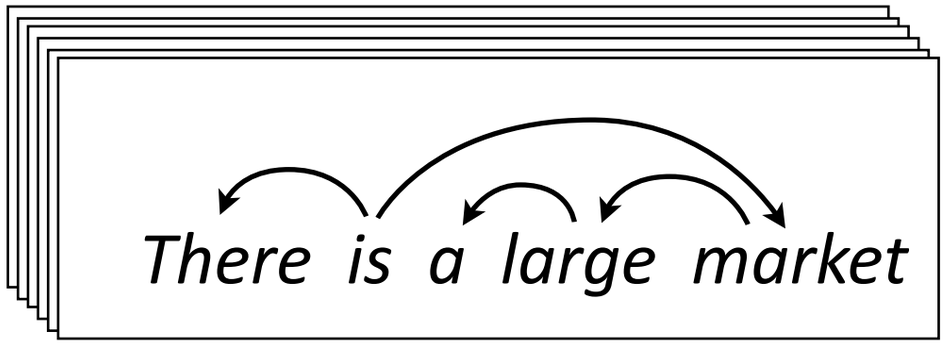
	# times	frequency (cumulative)
1	99,610	
2	151,198	
3	136,937	
...		



How many times each memory cost happens?

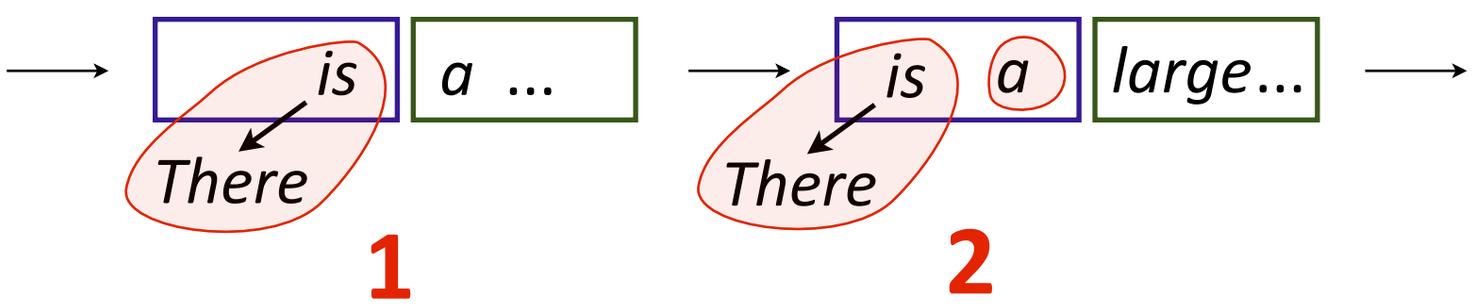
Example: English

Given the dependency treebank:



- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)

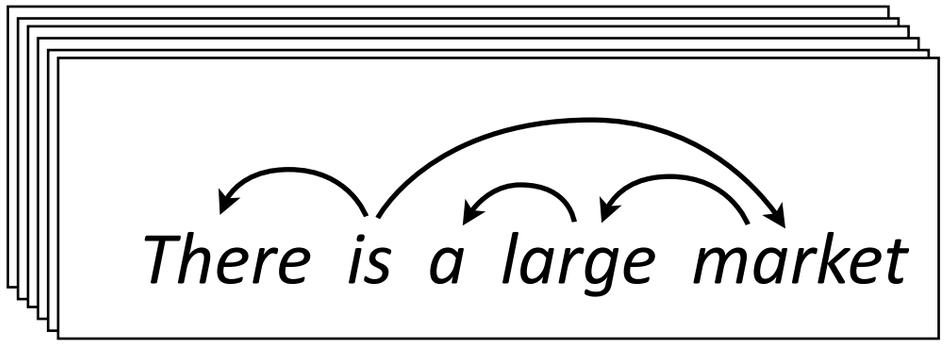
	# times	frequency (cumulative)
1	99,610	0.13 (0.13)
2	151,198	
3	136,937	
...		



How many times each memory cost happens?

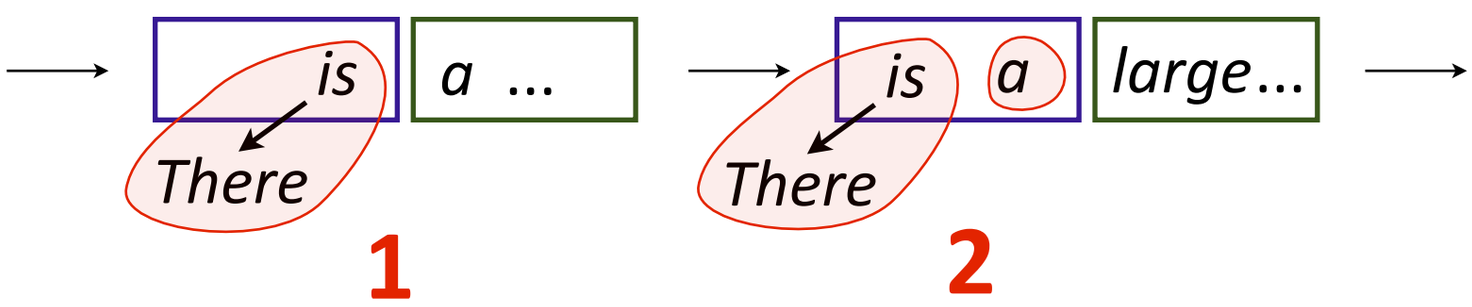
Example: English

Given the dependency treebank:



- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)

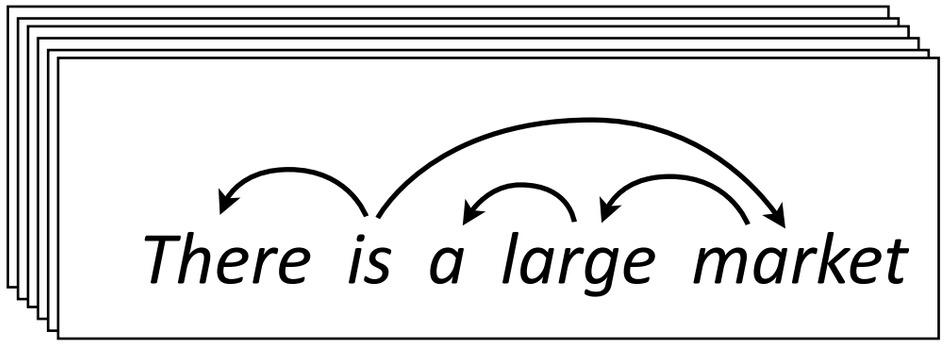
	# times	frequency (cumulative)
1	99,610	0.13 (0.13)
2	151,198	0.20 (0.33)
3	136,937	
...		



How many times each memory cost happens?

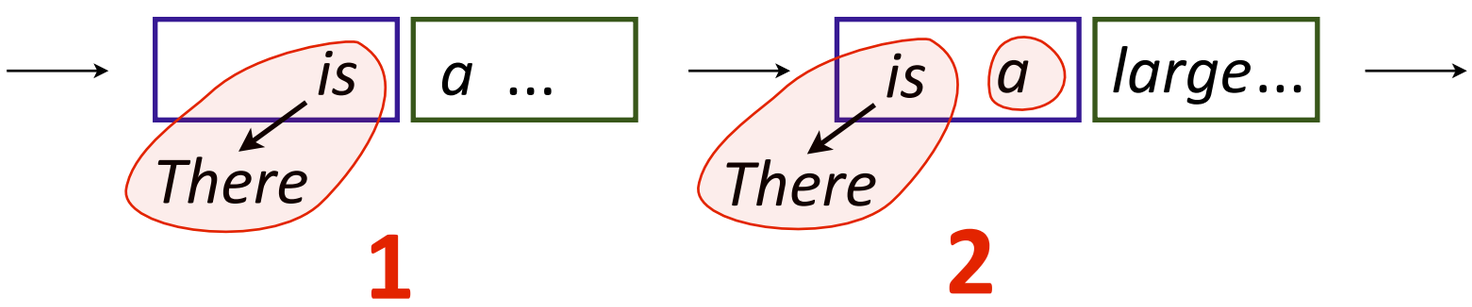
Example: English

Given the dependency treebank:



- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)

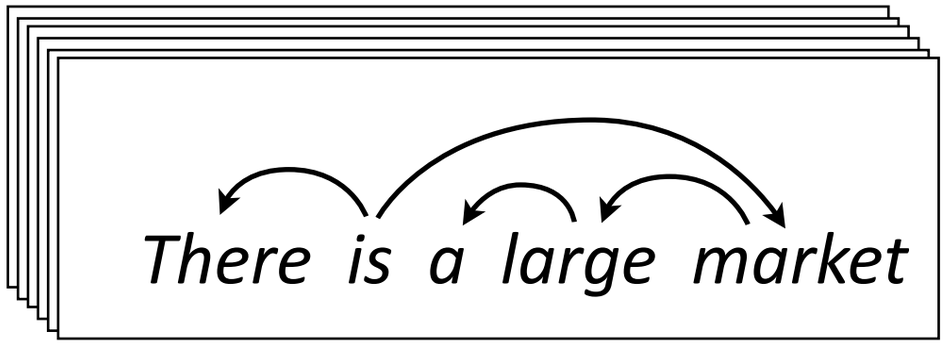
	# times	frequency (cumulative)
1	99,610	0.13 (0.13)
2	151,198	0.20 (0.33)
3	136,937	0.16 (0.49)
...		



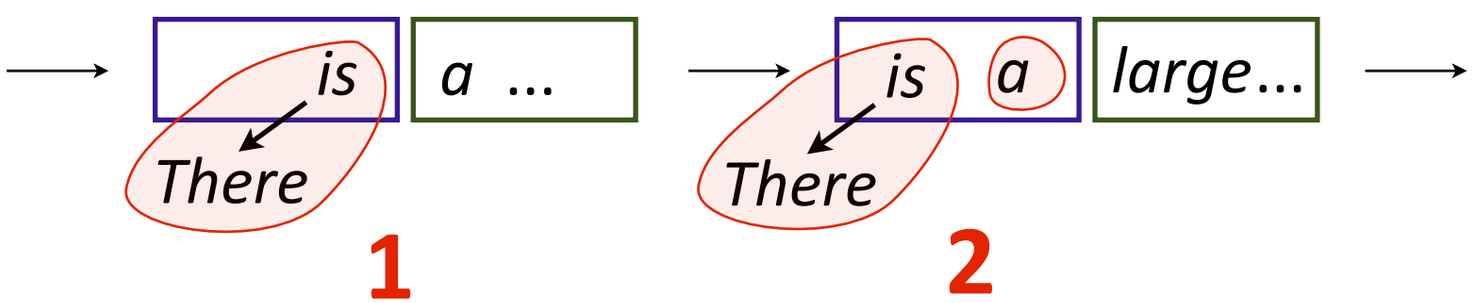
How many times each memory cost happens?

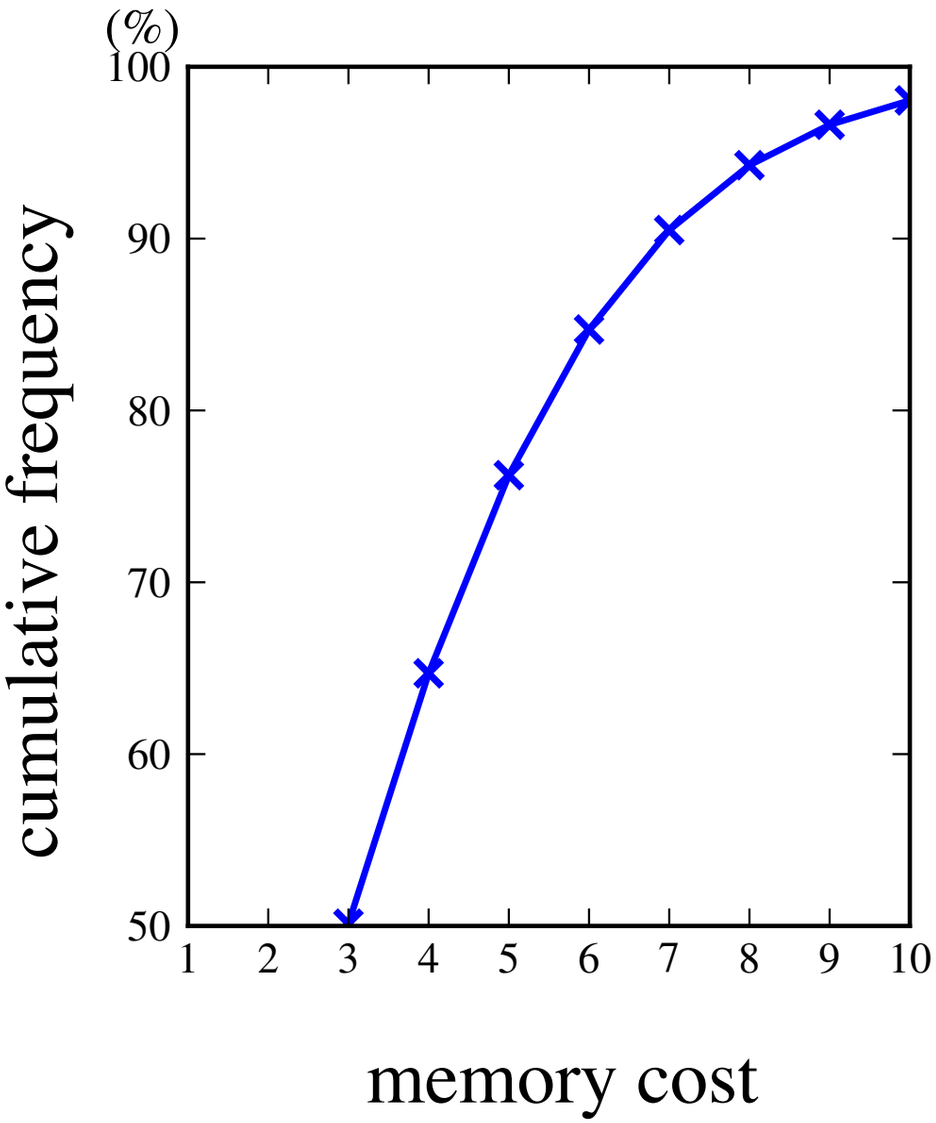
Example: English

Given the dependency treebank:



- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)

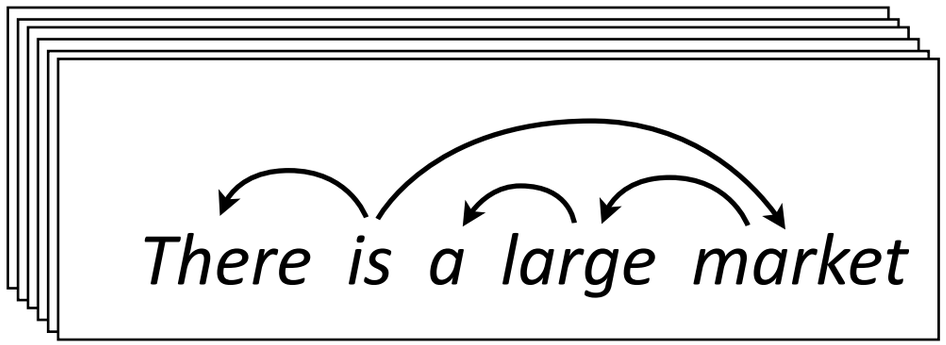




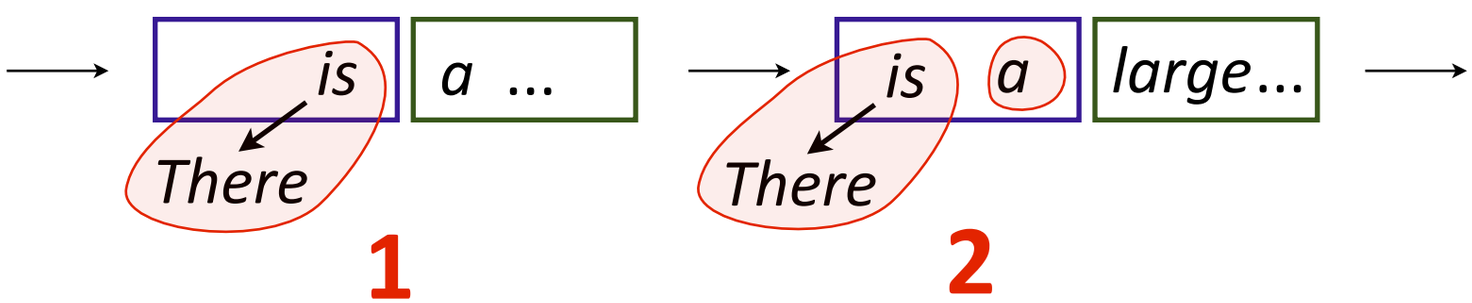
How many times each memory cost happens?

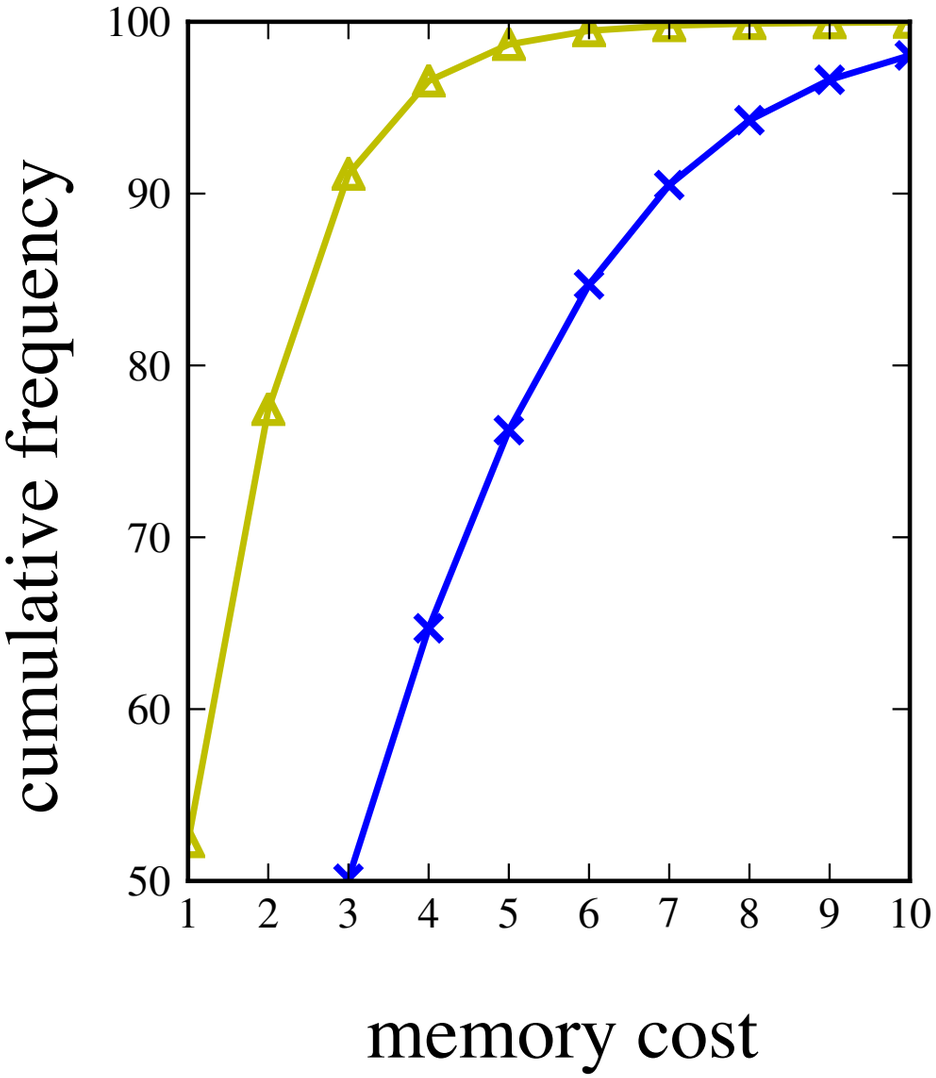
Example: English

Given the dependency treebank:



- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)

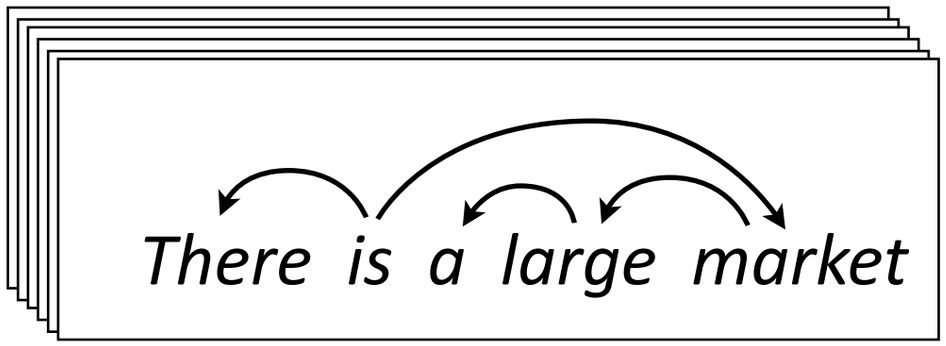




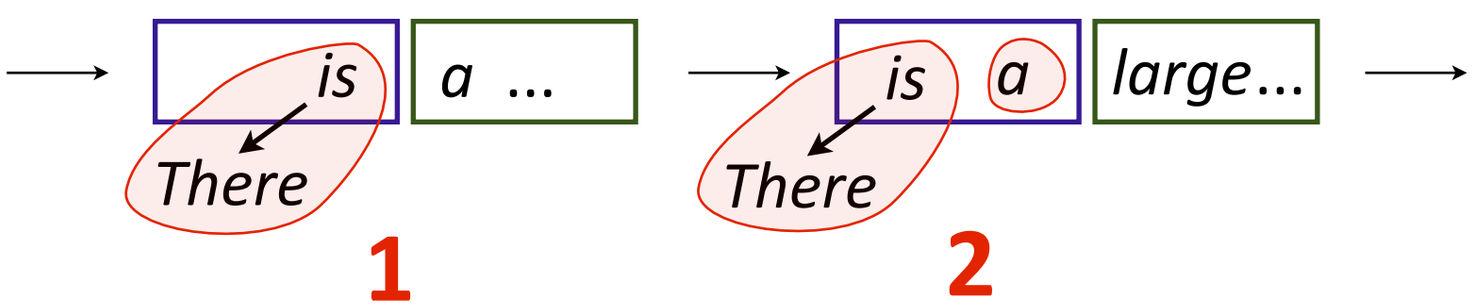
How many times each memory cost happens?

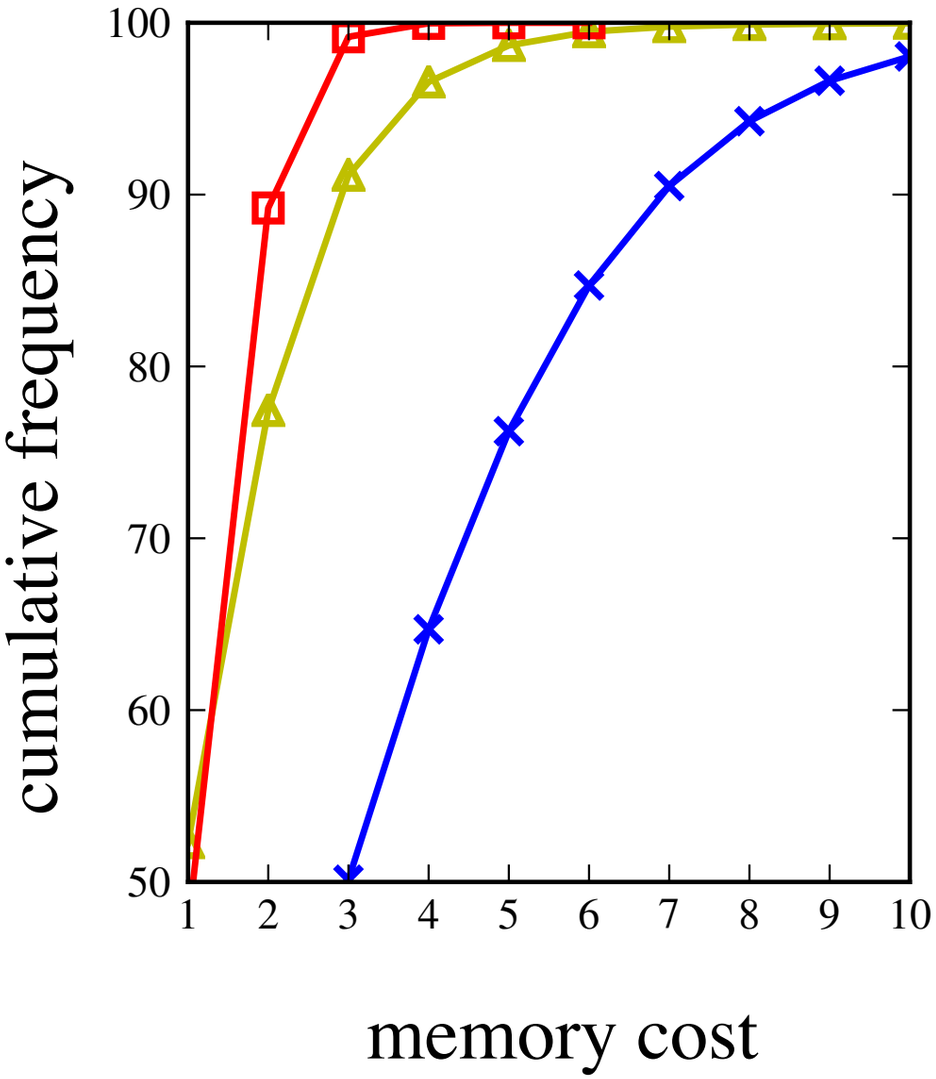
Example: English

Given the dependency treebank:



- Select a transition system (here arc-standard)
- Simulate oracle transitions (recover gold dependency)

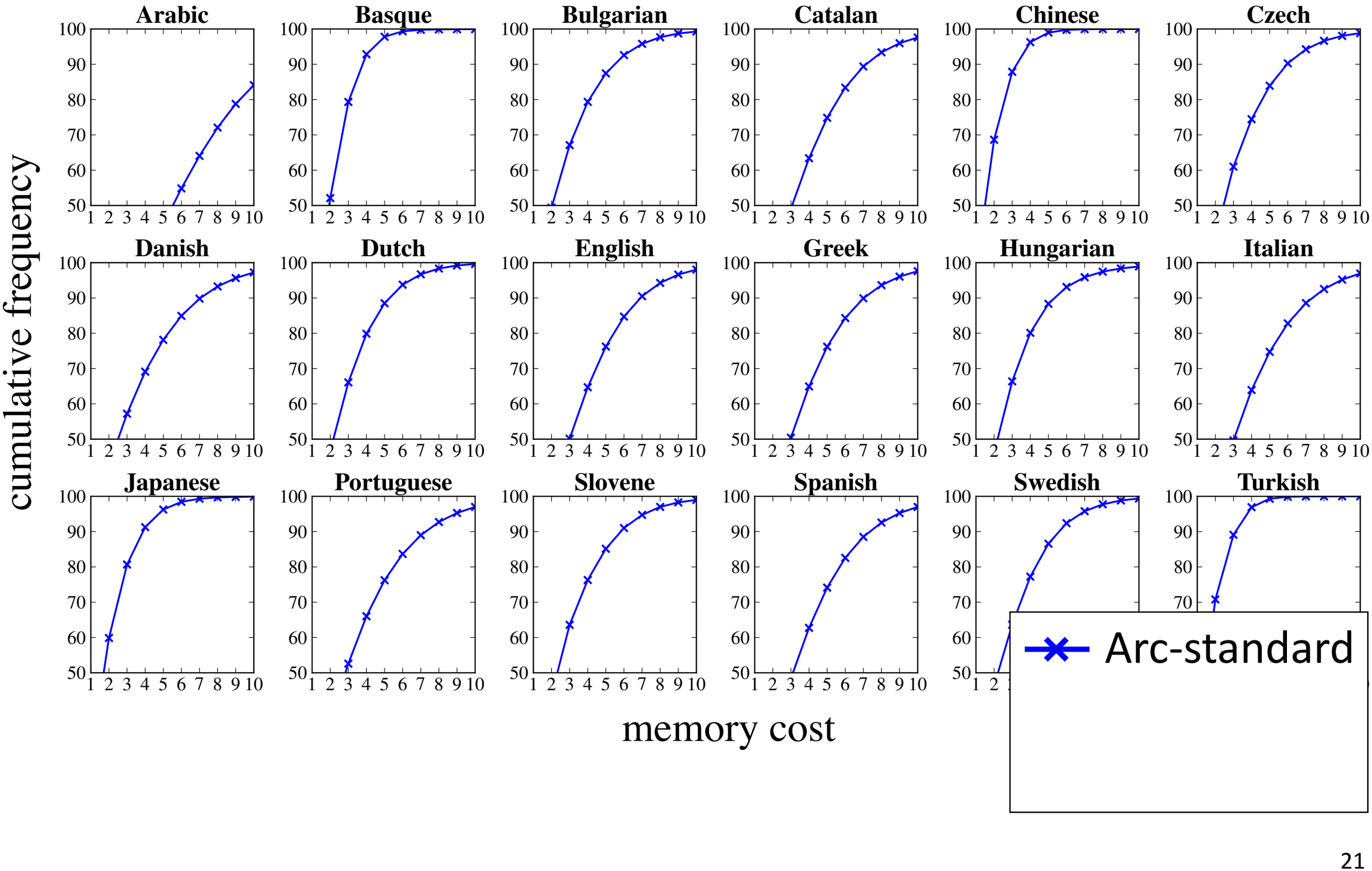




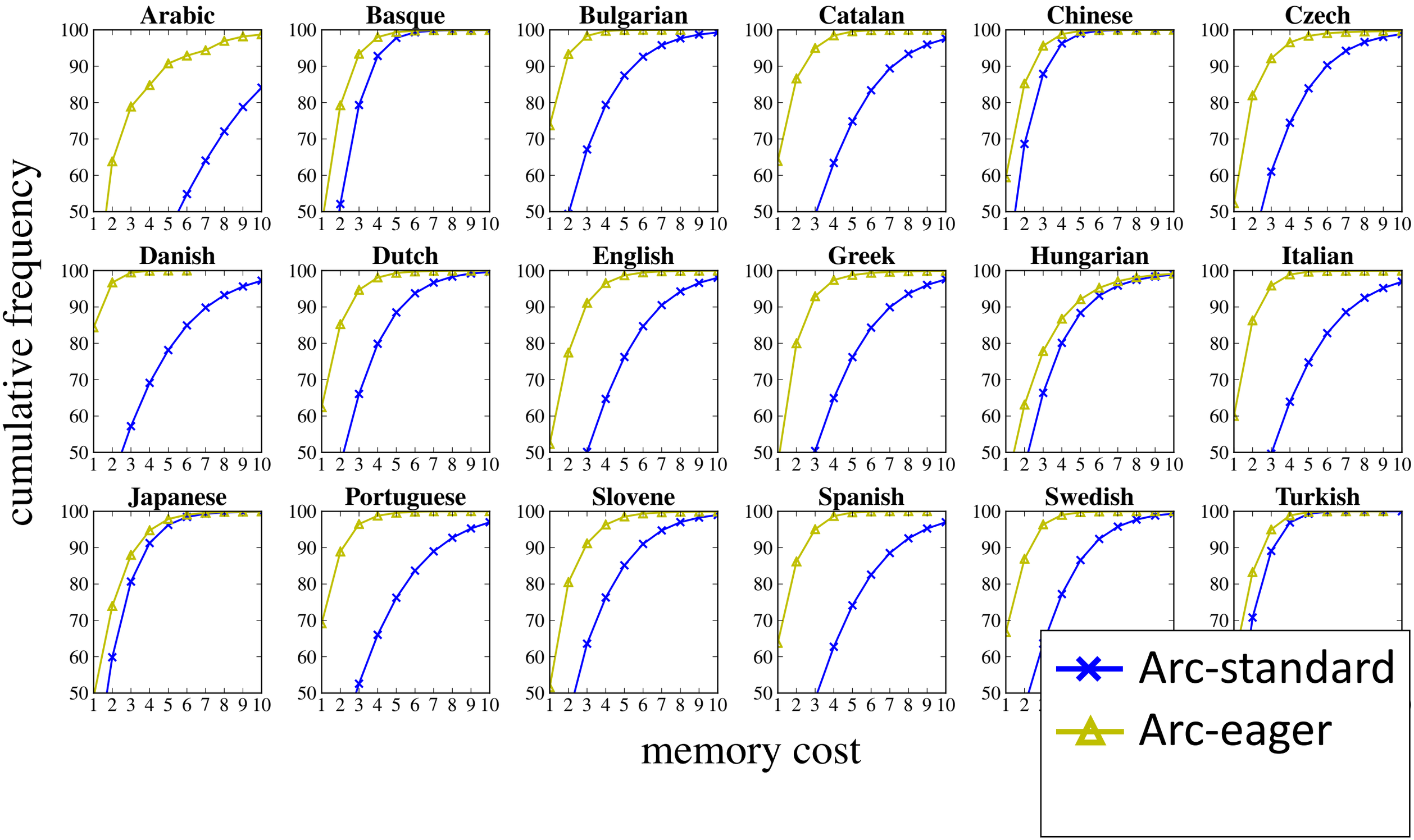
How many times each memory cost happens?

Cross-lingual comparison: Result

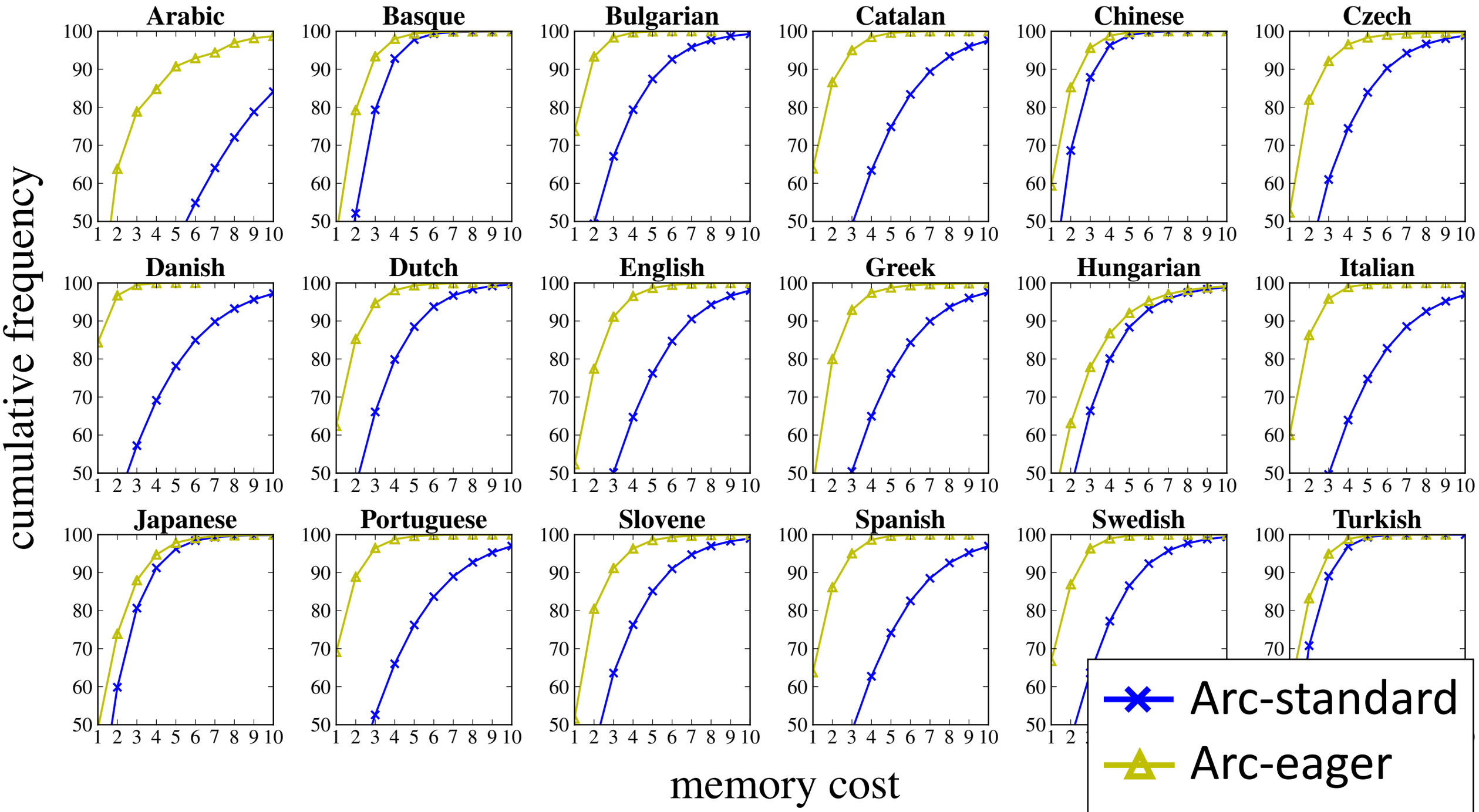
Cross-lingual comparison: Result



Cross-lingual comparison: Result

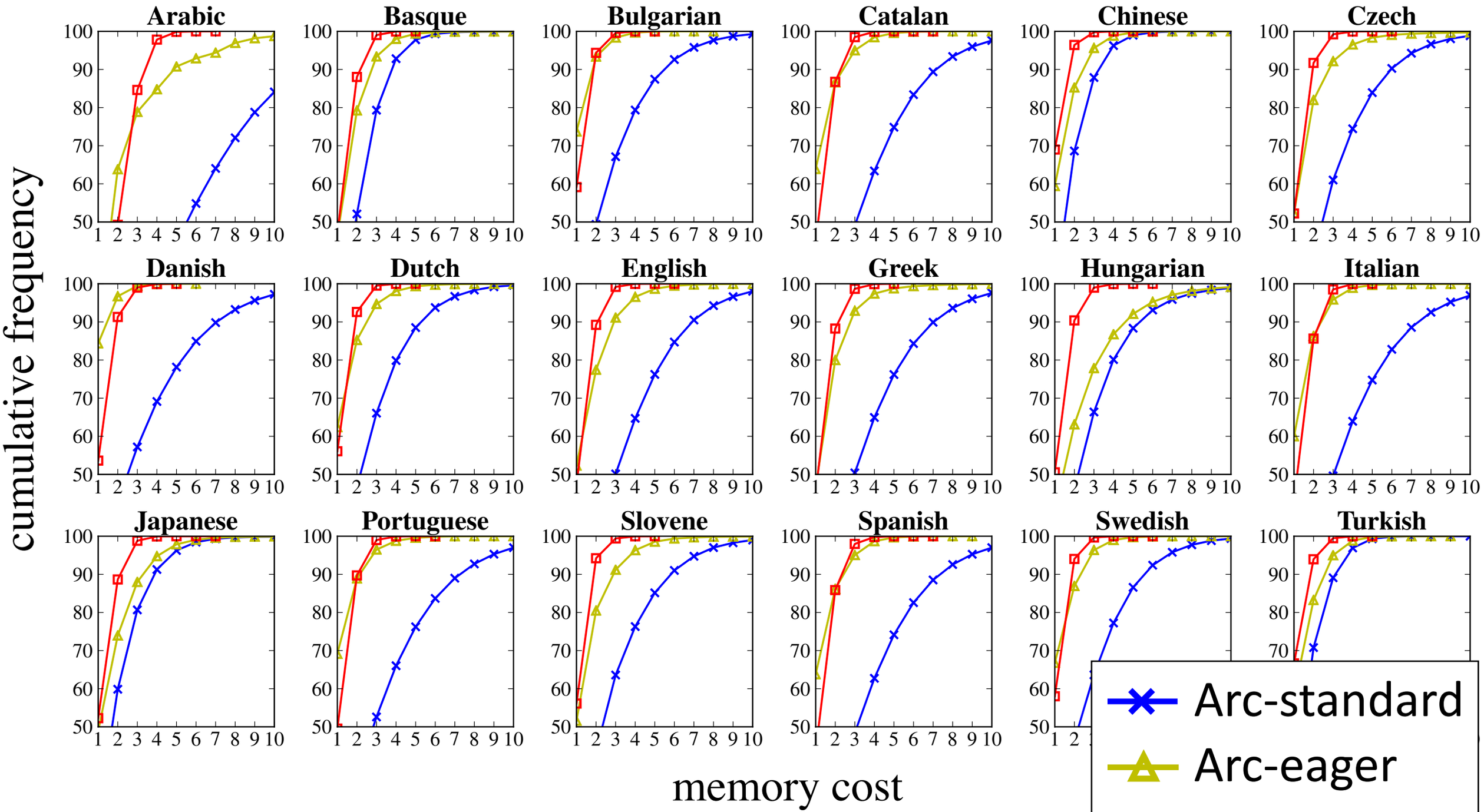


Cross-lingual comparison: Result



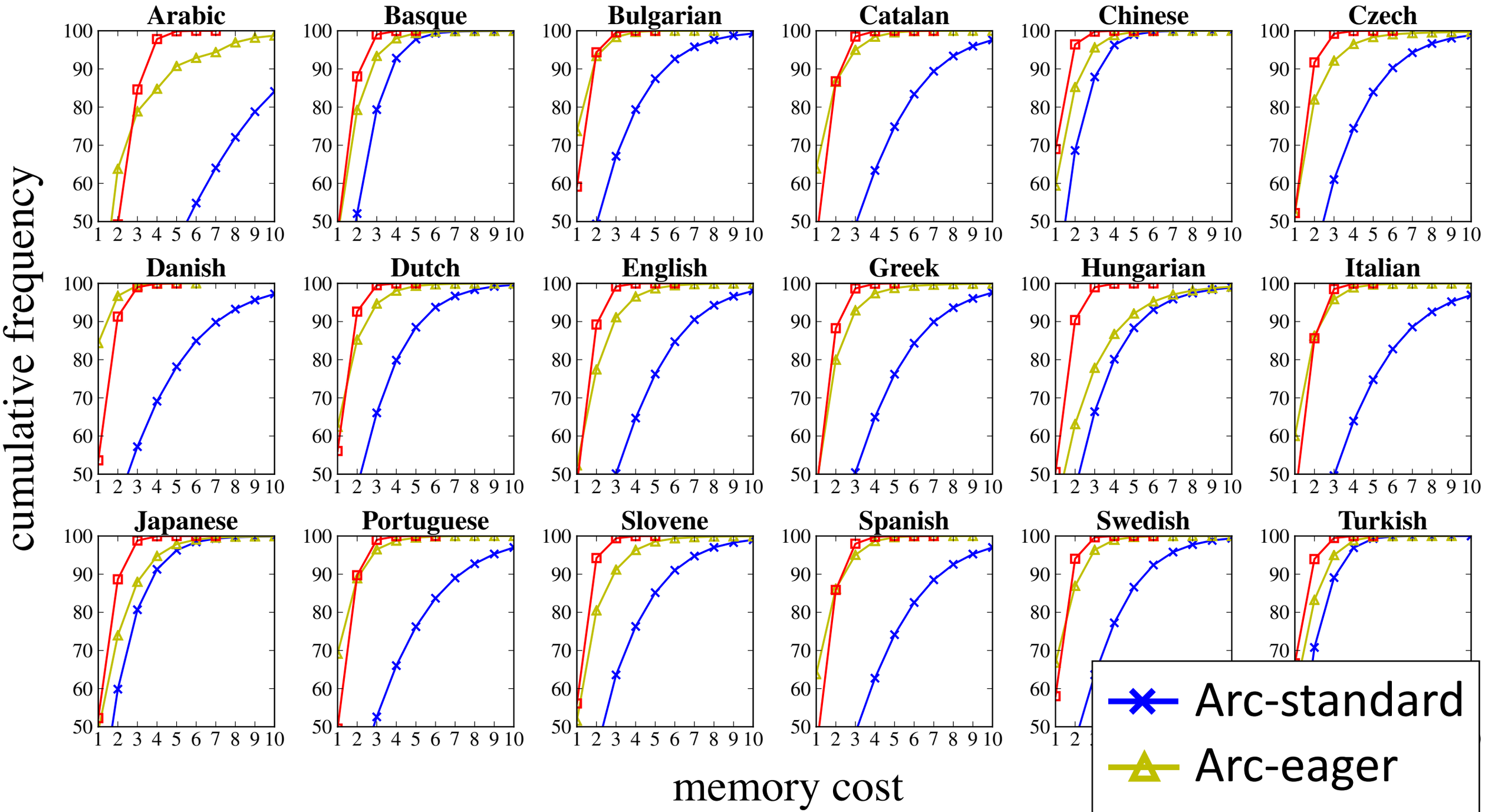
- Arc-eager requires less memory than arc-standard

Cross-lingual comparison: Result



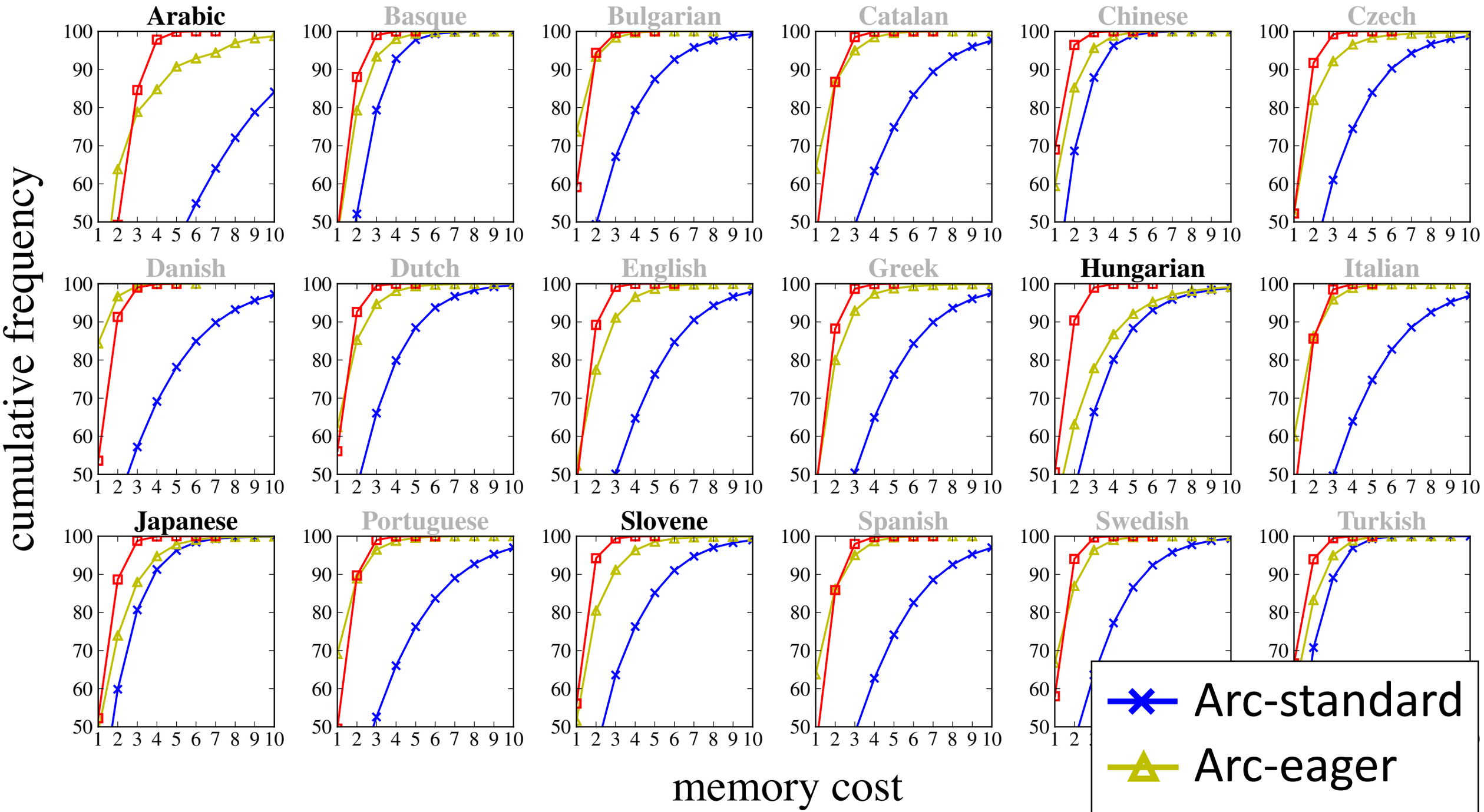
- Arc-eager requires less memory than arc-standard

Cross-lingual comparison: Result



- Arc-eager requires less memory than arc-standard
- Left-corner: except Arabic, 98% of states ⇒ cost < 4

Cross-lingual comparison: Result

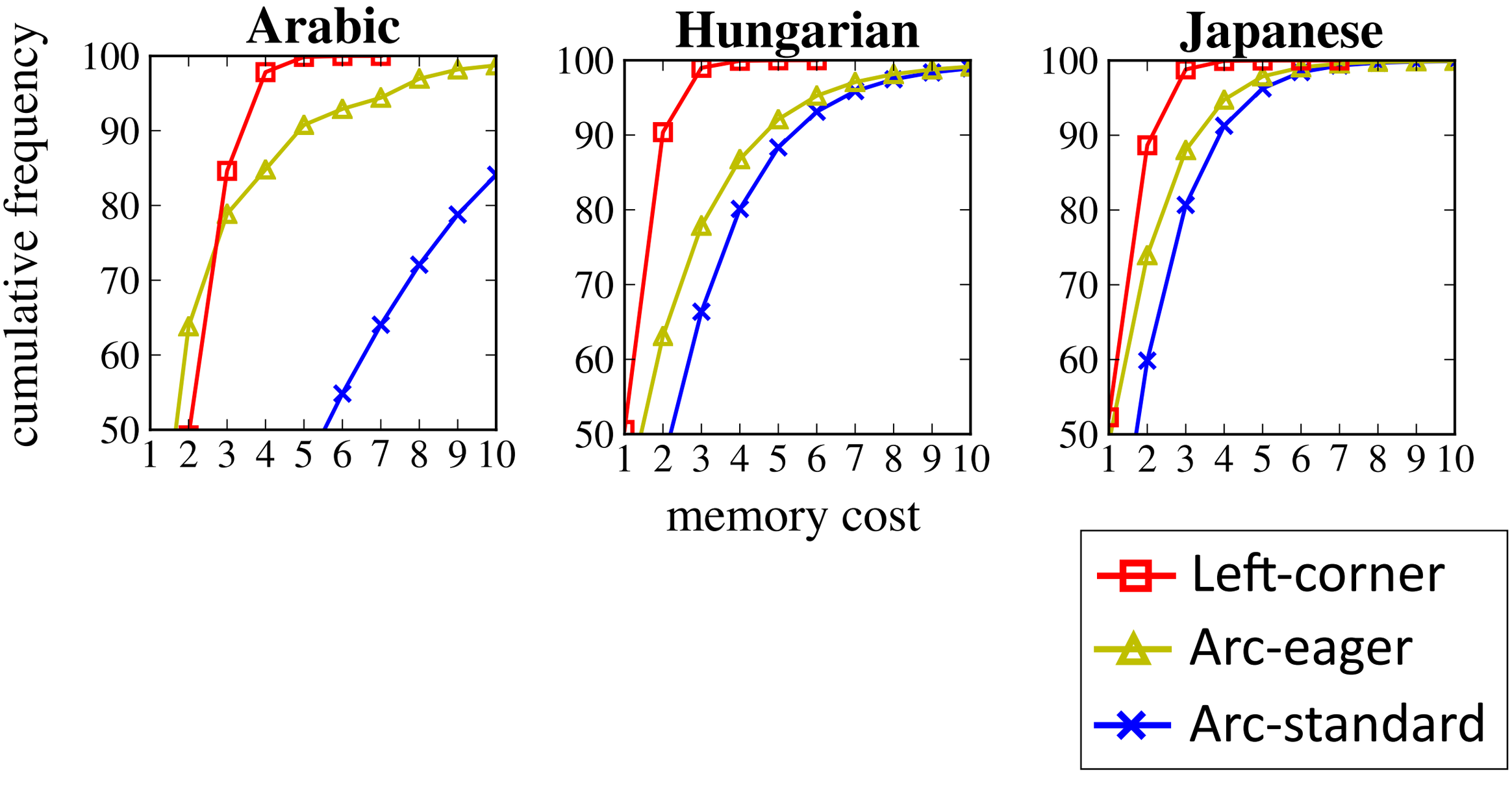


- Arc-eager requires less memory than arc-standard
- Left-corner: except Arabic, 98% of states ⇒ cost < 4

Universality

Arc-eager performs poorly in some languages

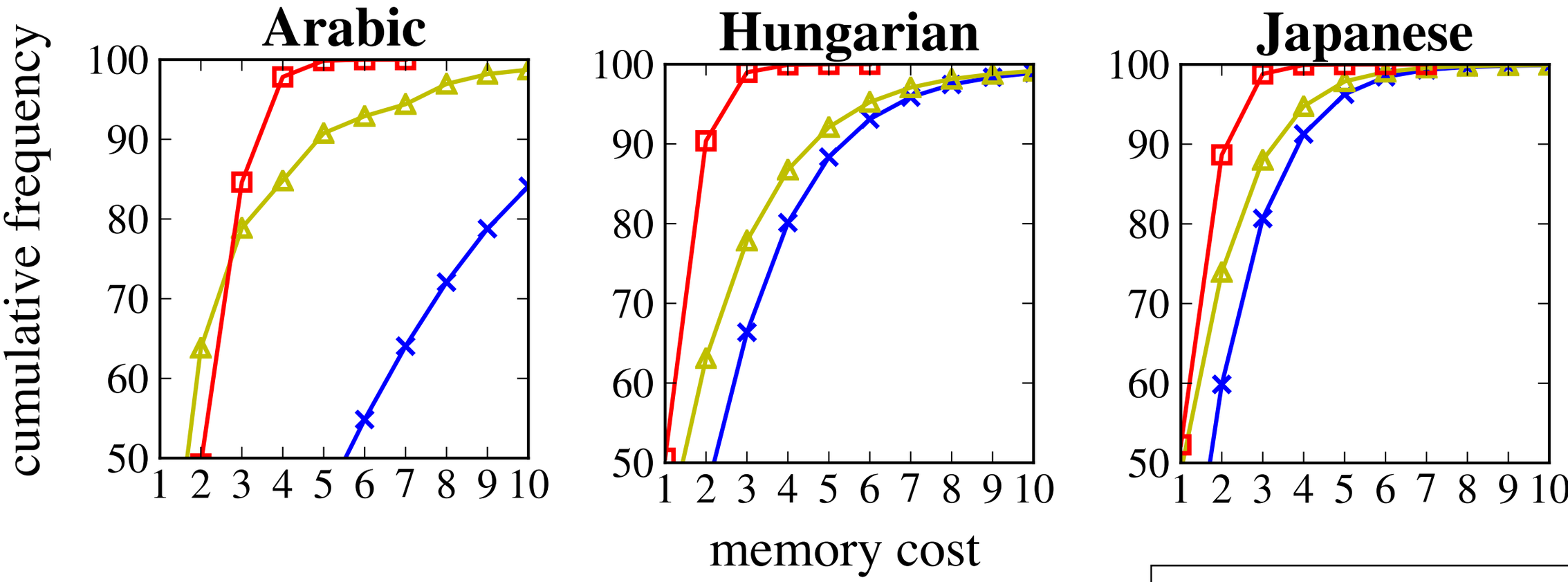
Left-corner's behavior is more consistent across languages



Universality

Arc-eager performs poorly in some languages

Left-corner's behavior is more consistent across languages



Indicate left-corner is cross-linguistically cognitively plausible in terms of memory cost

Exploring syntactic bias

So far...

- We defined left-corner dependency parser on transition system
- This parser uses less memory for sentences in diverse languages

Exploring syntactic bias

So far...

- We defined left-corner dependency parser on transition system
- This parser uses less memory for sentences in diverse languages

Question: Does this parser require less memory **only for grammatically correct sentences**?

Exploring syntactic bias

So far...

- We defined left-corner dependency parser on transition system
- This parser uses less memory for sentences in diverse languages

Question: Does this parser require less memory **only for grammatically correct sentences**?

⟶ Has a grammar of language evolved to avoid center-embedded dependency?

Exploring syntactic bias

So far...

- We defined left-corner dependency parser on transition system
- This parser uses less memory for sentences in diverse languages

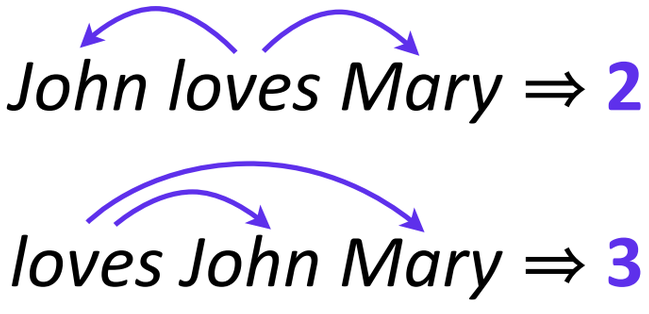
Question: Does this parser require less memory **only for grammatically correct sentences**?

⟶ Has a grammar of language evolved to avoid center-embedded dependency?

Previous study: Dependency length minimization

[Gildea & Temperley, 2007; Park & Levy, 2009]

- Dependency trees may have evolved to minimize the sum of dependency lengths



Exploring syntactic bias

So far...

- We defined left-corner dependency parser on transition system
- This parser uses less memory for sentences in diverse languages

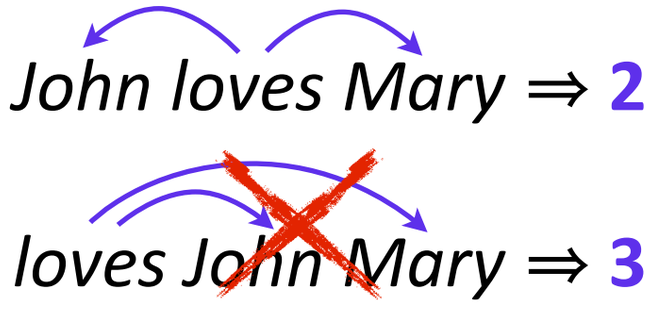
Question: Does this parser require less memory **only for grammatically correct sentences**?

⟶ Has a grammar of language evolved to avoid center-embedded dependency?

Previous study: Dependency length minimization

[Gildea & Temperley, 2007; Park & Levy, 2009]

- Dependency trees may have evolved to minimize the sum of dependency lengths



vs. ungrammatical treebank: Setting

Question: Does this parser require less memory **only for grammatically correct sentences**?

Method: Following [Gildea and Temperley, 2007]

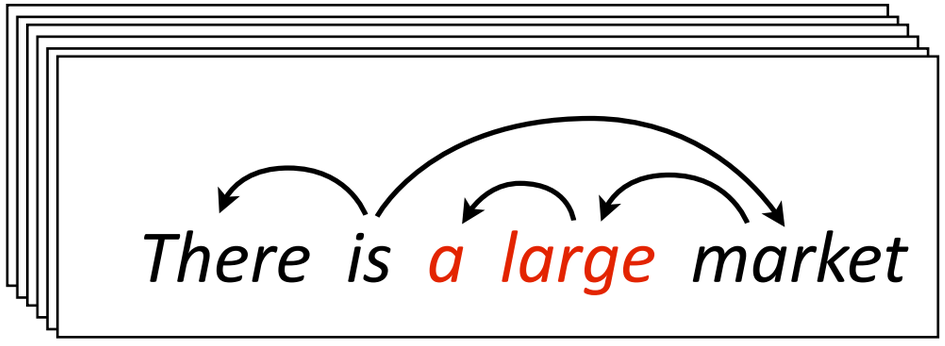
⇒ We prepare treebanks with noise, then run our parser

vs. ungrammatical treebank: Setting

Question: Does this parser require less memory **only for grammatically correct sentences**?

Method: Following [Gildea and Temperley, 2007]

⇒ We prepare treebanks with noise, then run our parser



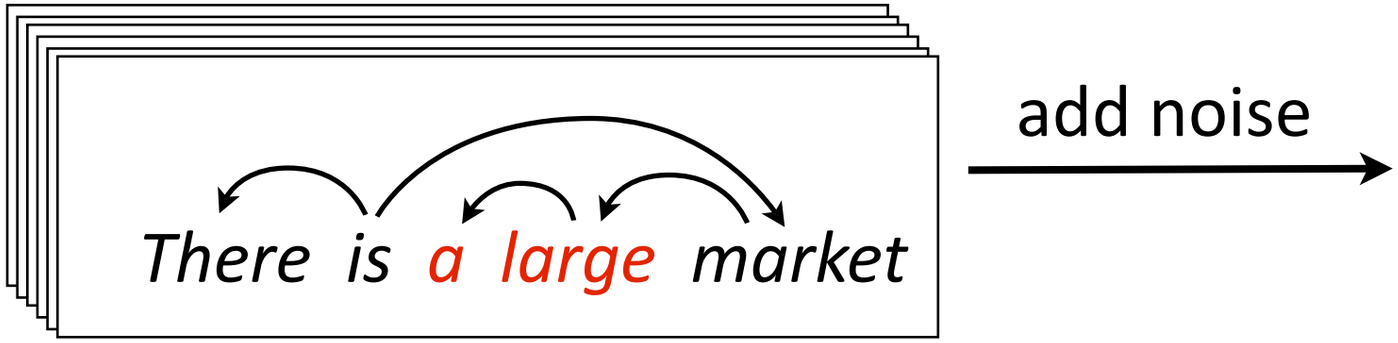
original treebank

vs. ungrammatical treebank: Setting

Question: Does this parser require less memory **only for grammatically correct sentences**?

Method: Following [Gildea and Temperley, 2007]

⇒ We prepare treebanks with noise, then run our parser



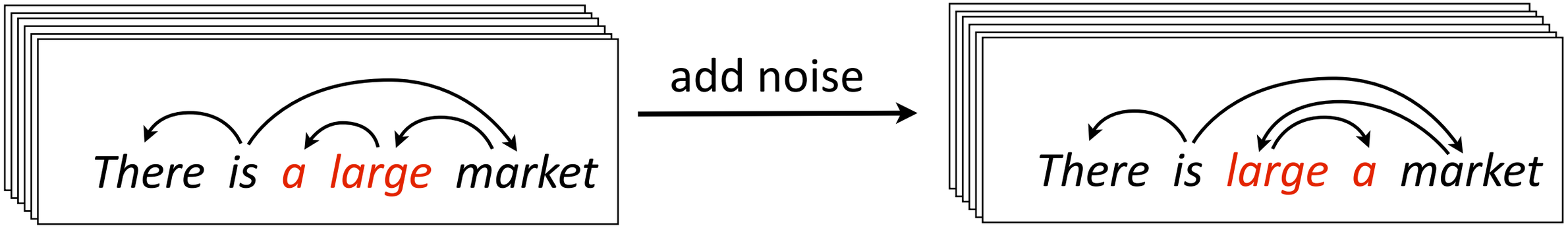
original treebank

vs. ungrammatical treebank: Setting

Question: Does this parser require less memory **only for grammatically correct sentences**?

Method: Following [Gildea and Temperley, 2007]

⇒ We prepare treebanks with noise, then run our parser

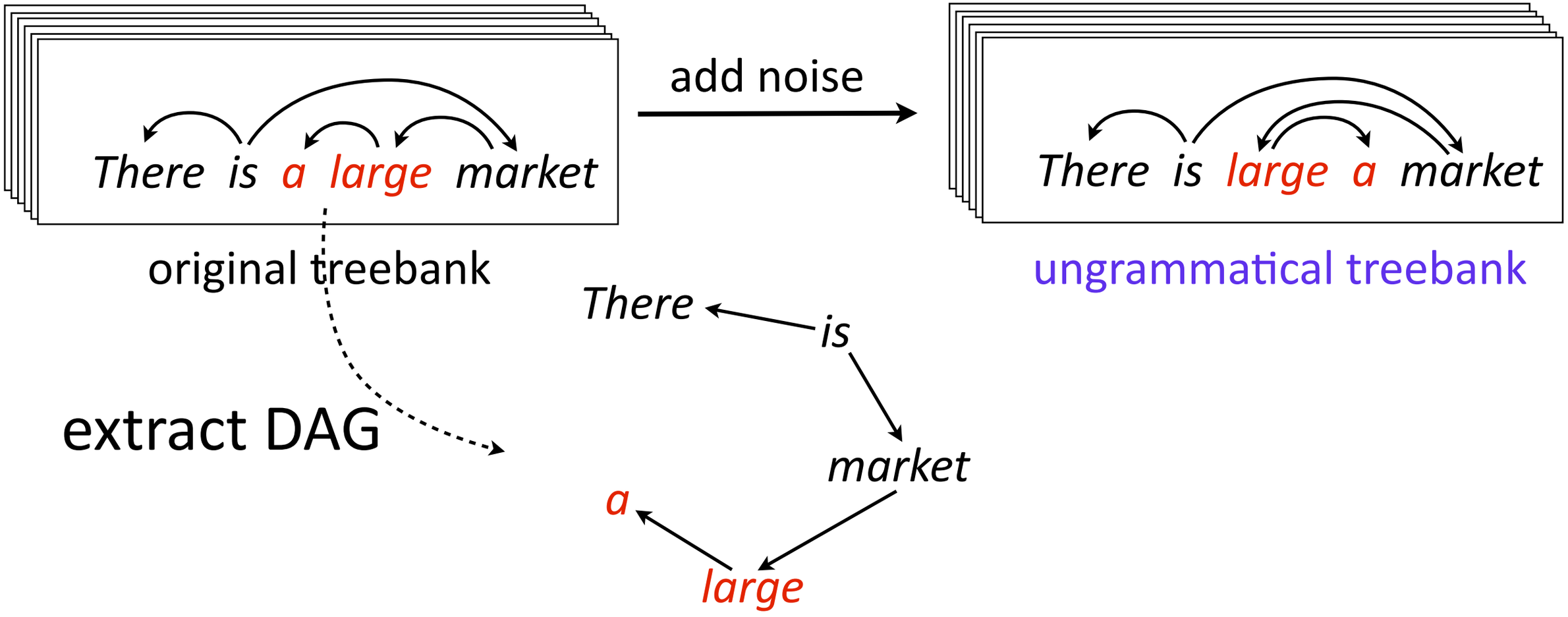


vs. ungrammatical treebank: Setting

Question: Does this parser require less memory **only for grammatically correct sentences**?

Method: Following [Gildea and Temperley, 2007]

⇒ We prepare treebanks with noise, then run our parser

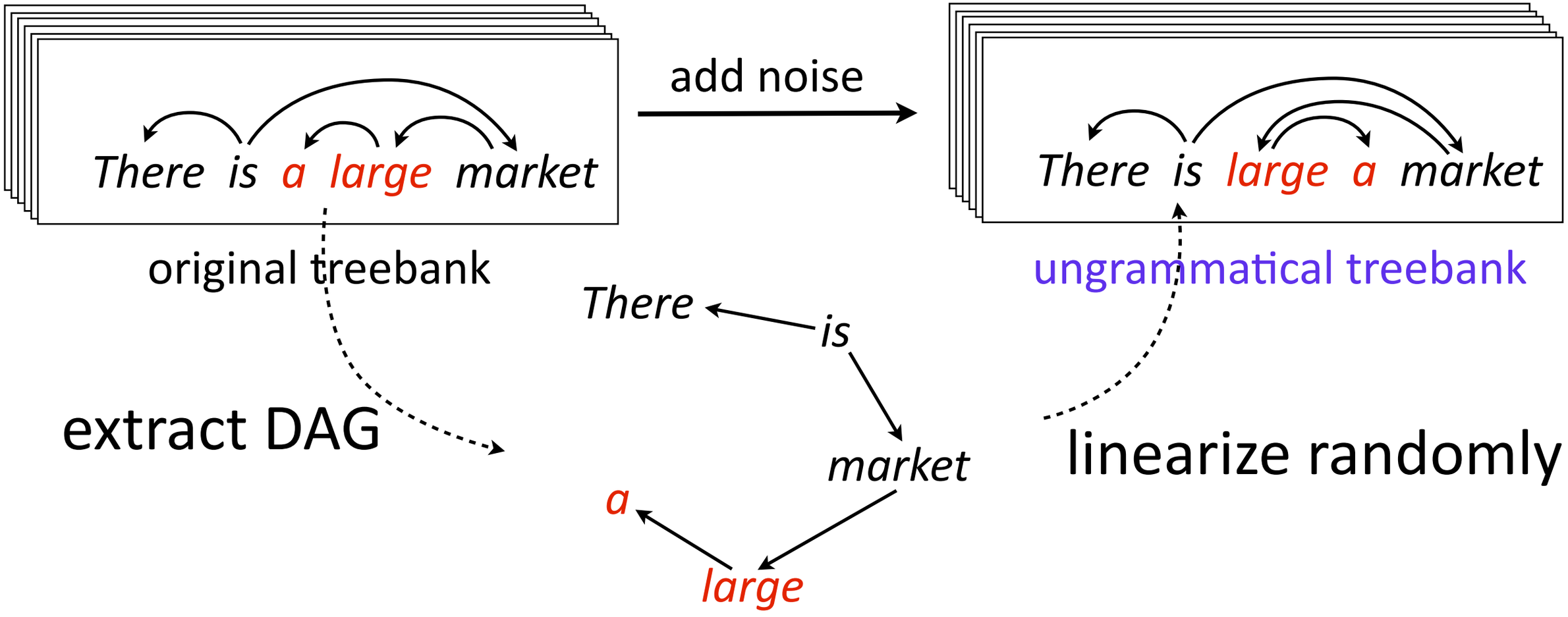


vs. ungrammatical treebank: Setting

Question: Does this parser require less memory **only for grammatically correct sentences**?

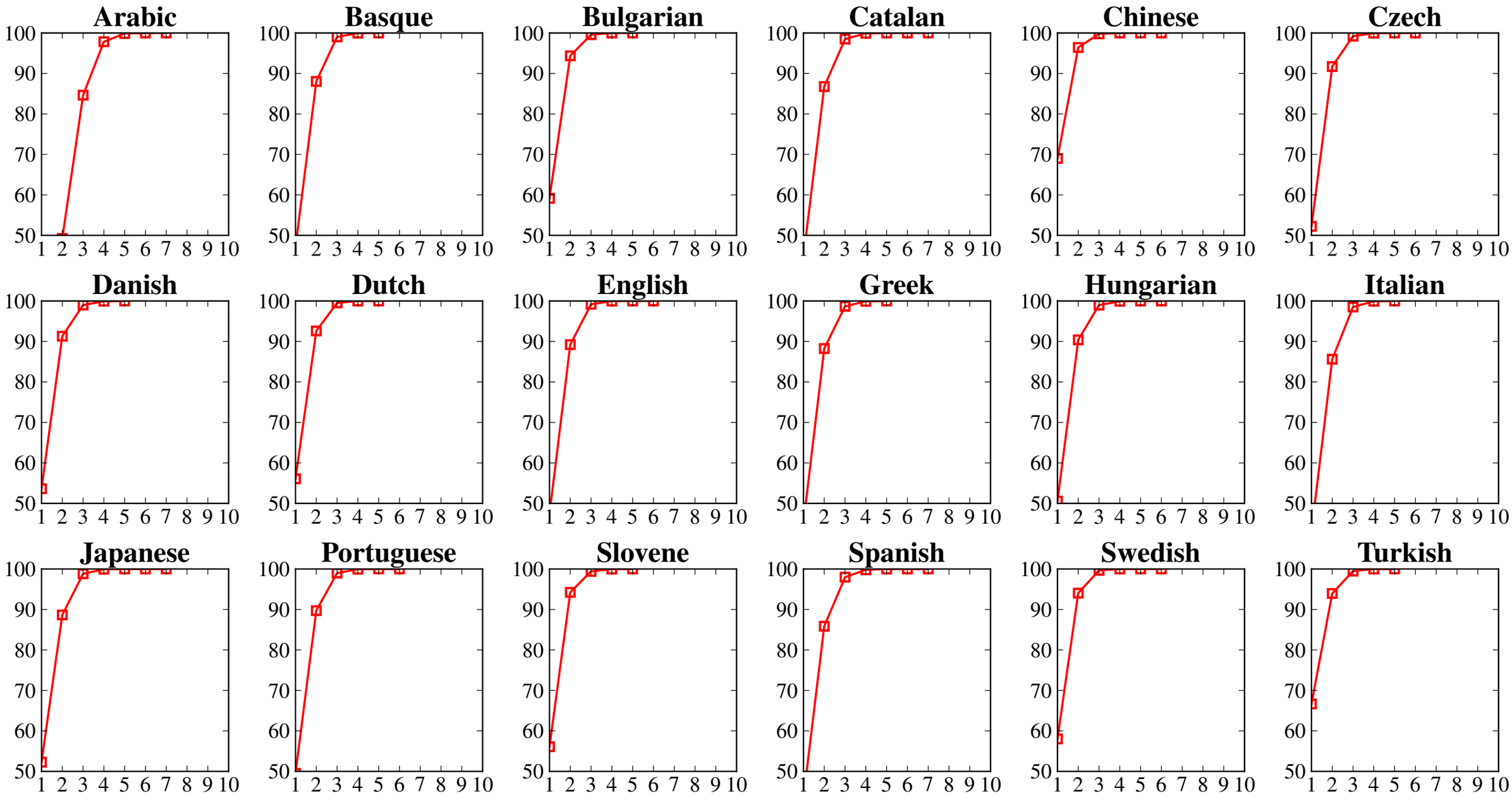
Method: Following [Gildea and Temperley, 2007]

⇒ We prepare treebanks with noise, then run our parser



vs. ungrammatical treebank: Result

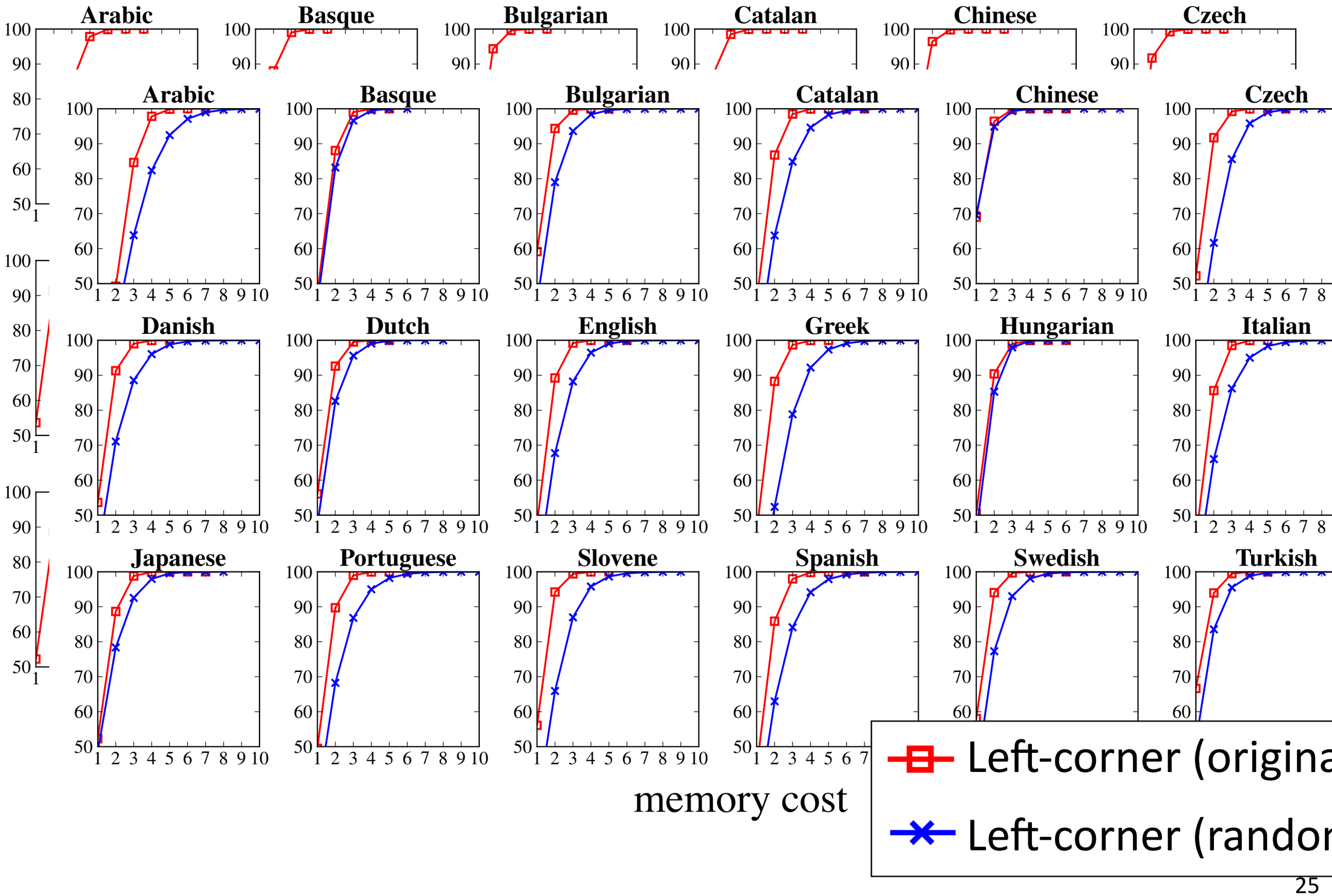
vs. ungrammatical treebank: Result



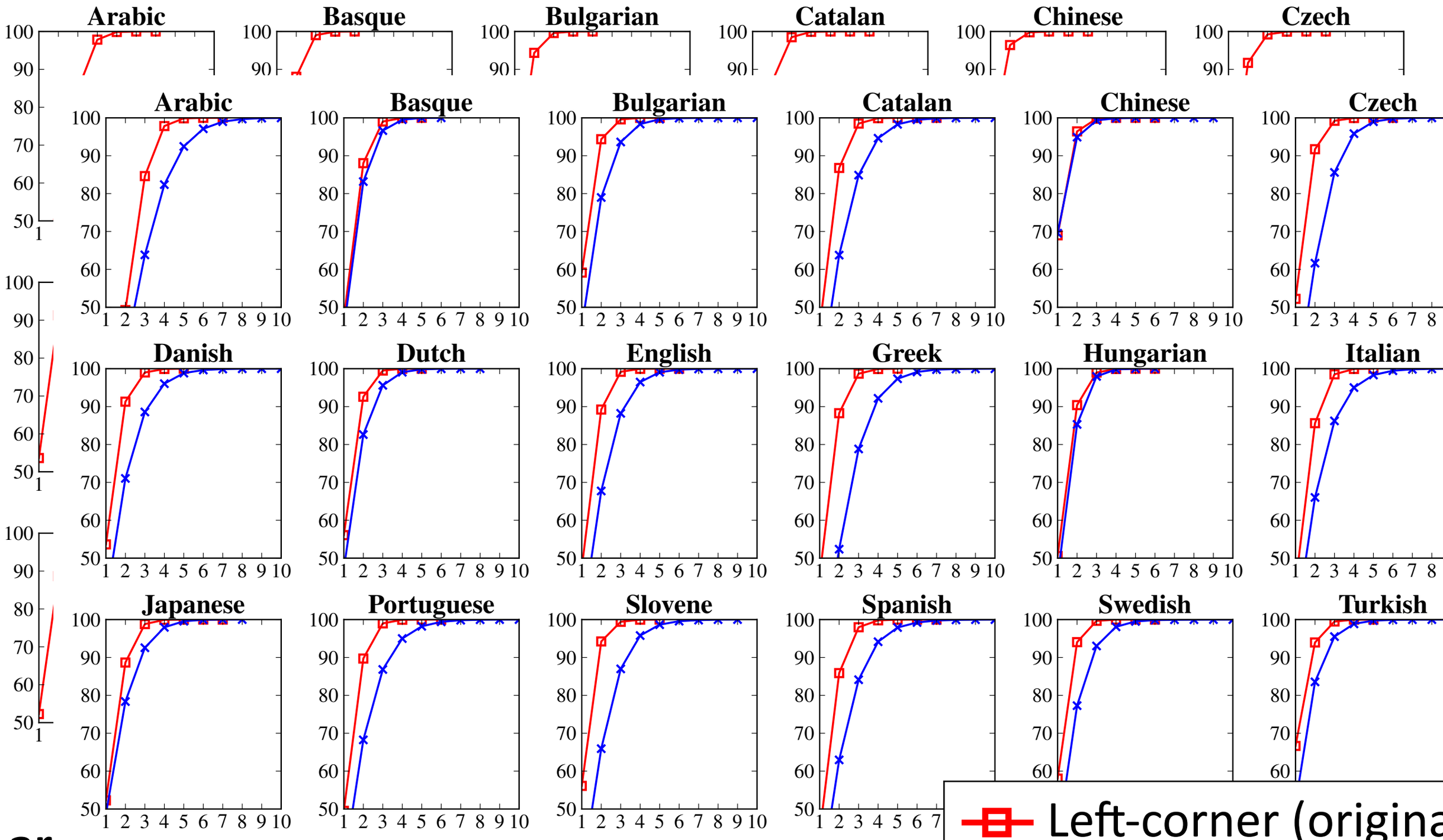
memory cost

Left-corner (origina

vs. ungrammatical treebank: Result



vs. ungrammatical treebank: Result



- **Large differences in most languages**
- Small differences in Basque, Chinese, and Hur

Conclusion

Dependency parsing:

- New transition system with left-corner strategy

Conclusion

Dependency parsing:

- New transition system with left-corner strategy

Psycholinguistics:

- First cross-lingual study of left-corner parser's behavior
- Consistent lesser memory cost indicates its cross-linguistic cognitive plausibility

Conclusion

Dependency parsing:

- New transition system with left-corner strategy

Psycholinguistics:

- First cross-lingual study of left-corner parser's behavior
- Consistent lesser memory cost indicates its cross-linguistic cognitive plausibility

Syntactic bias:

- Bias to avoid center-embedding may be language universal

Future work

Future work

Supervised learning:

- Automatic parsing evaluation
- Memory and accuracy tradeoff?

Future work

Supervised learning:

- Automatic parsing evaluation
- Memory and accuracy tradeoff?

Unsupervised learning [Klein & Manning, 2004]:

- Exploiting syntactic bias is crucial [Mareček & Žabokrtský, 2012]
- Unsupervised transition parsing with this parser? [Cohen et al., 2011]

Future work

Supervised learning:

- Automatic parsing evaluation
- Memory and accuracy tradeoff?

Unsupervised learning [Klein & Manning, 2004]:

- Exploiting syntactic bias is crucial [Mareček & Žabokrtský, 2012]
- Unsupervised transition parsing with this parser? [Cohen et al., 2011]

Other extensions:

- Left-corner parsing for non-projective structure with additional actions [Nivre, 2009; etc]

Future work

Supervised learning:

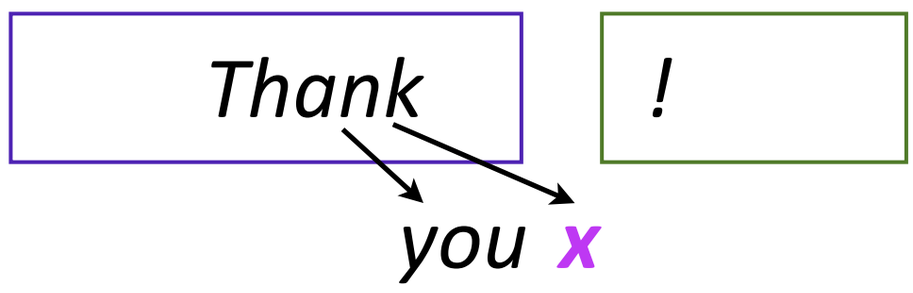
- Automatic parsing evaluation
- Memory and accuracy tradeoff?

Unsupervised learning [Klein & Manning, 2004]:

- Exploiting syntactic bias is crucial [Mareček & Žabokrtský, 2012]
- Unsupervised transition parsing with this parser? [Cohen et al., 2011]

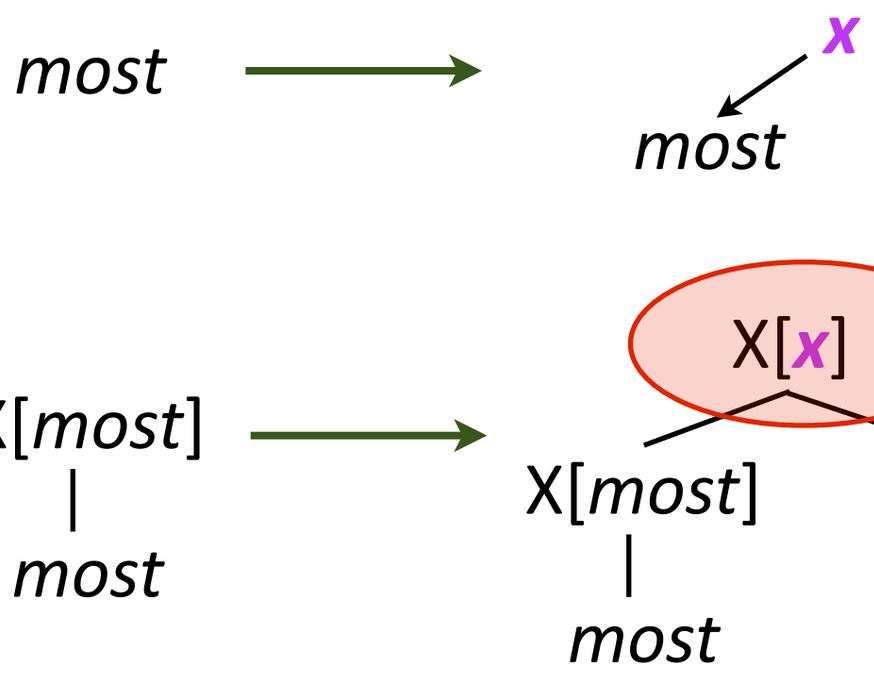
Other extensions:

- Left-corner parsing for non-projective structure with additional actions [Nivre, 2009; etc]

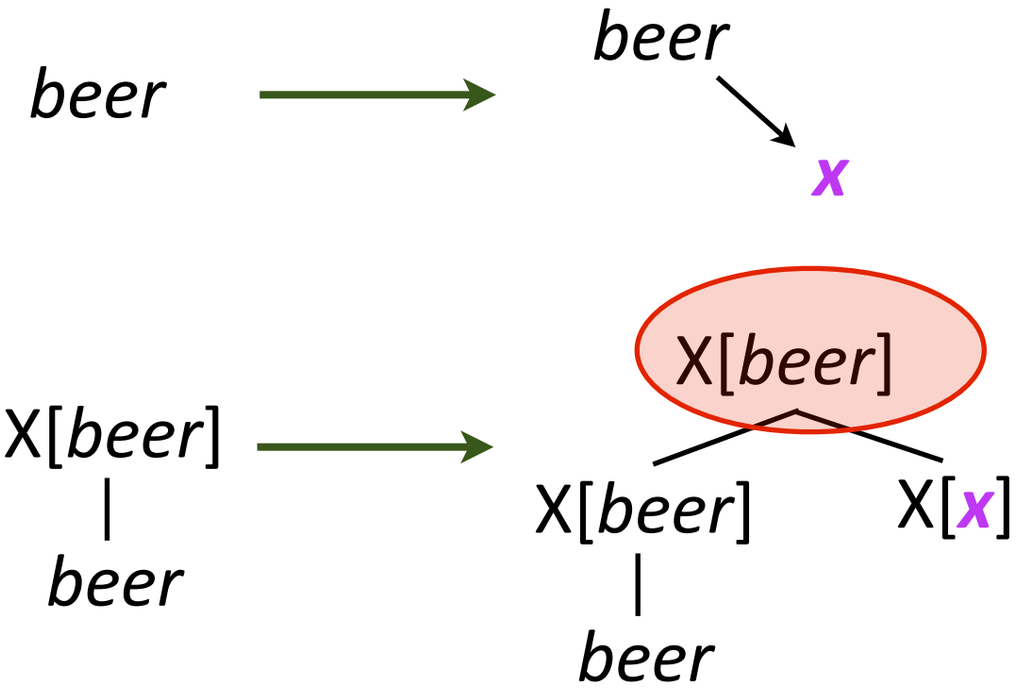


Prediction and CNF interpretation

LEFT-PRED:



RIGHT-PRED:

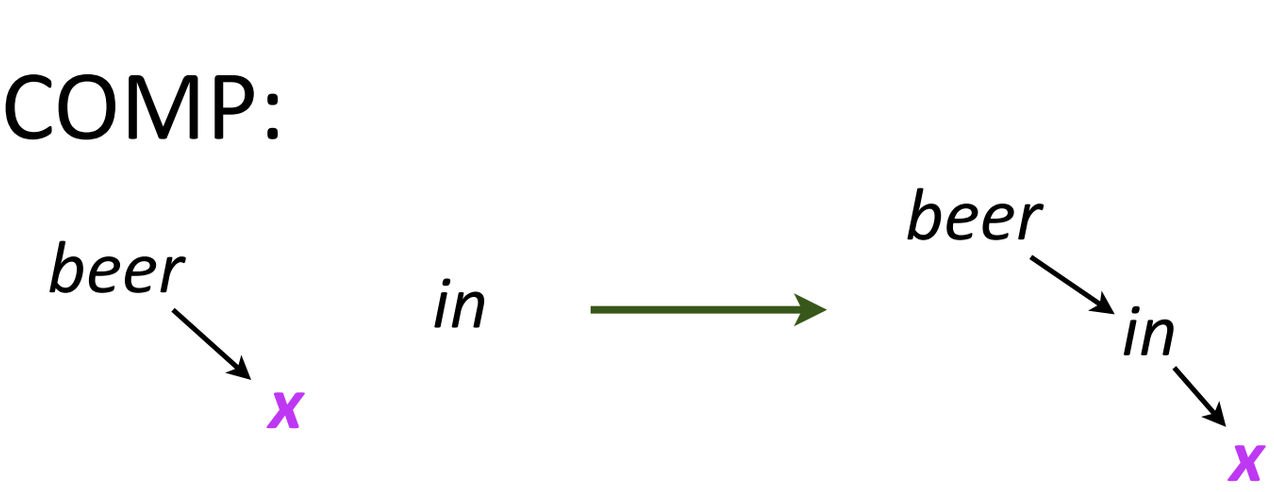


The same form as **prediction** in original left-corner parser

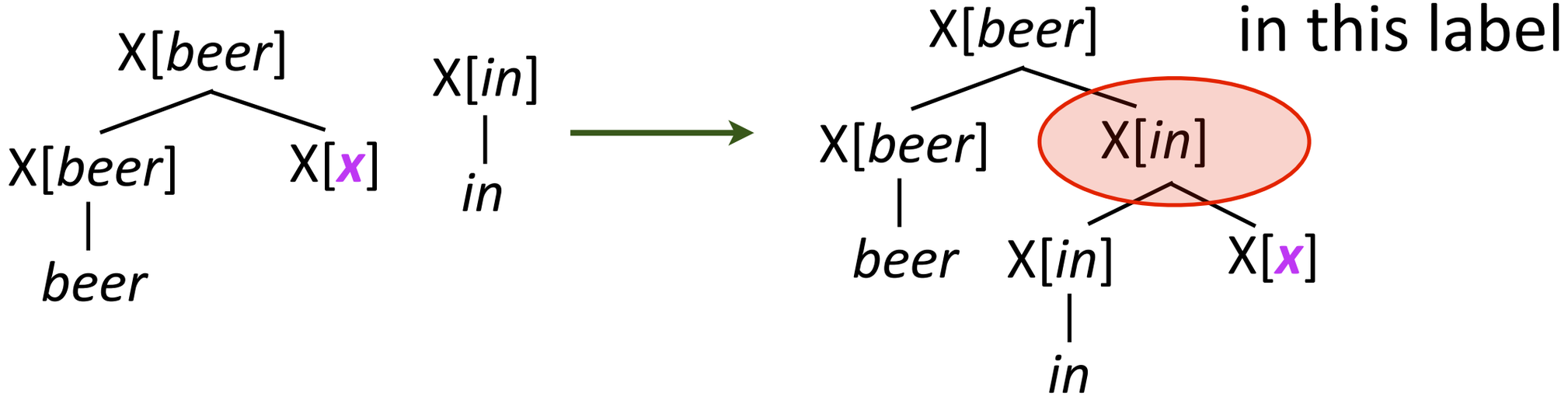
Difference: predicted parent label
(Which child is the head child?)

Composition and CNF interpretation

RIGHT-COMP:



LEFT-COMP differs in this label



The same form as original **composition** [Resnik, 1992]

Important result: Our parser implicitly recognizes a CNF tree as the original left-corner parser would do