Fast(er) Exact Decoding and Global Training for Transition-Based Dependency Parsing via a Minimal Feature Set

Tianze Shi* Liang Huang† Lillian Lee*

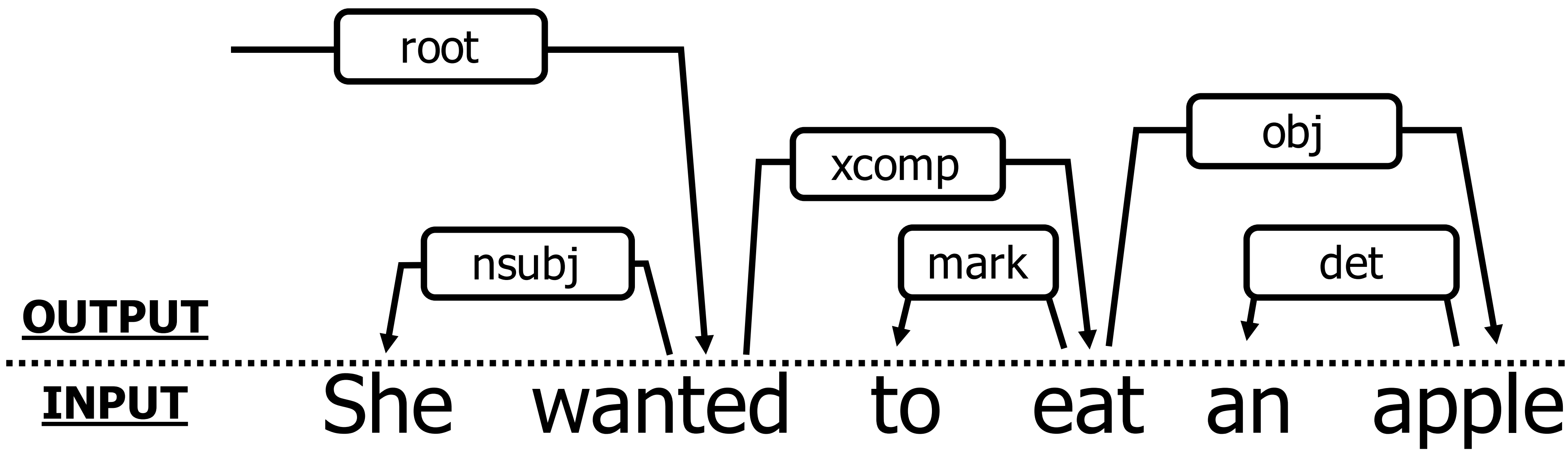
* Cornell University

† Oregon State University

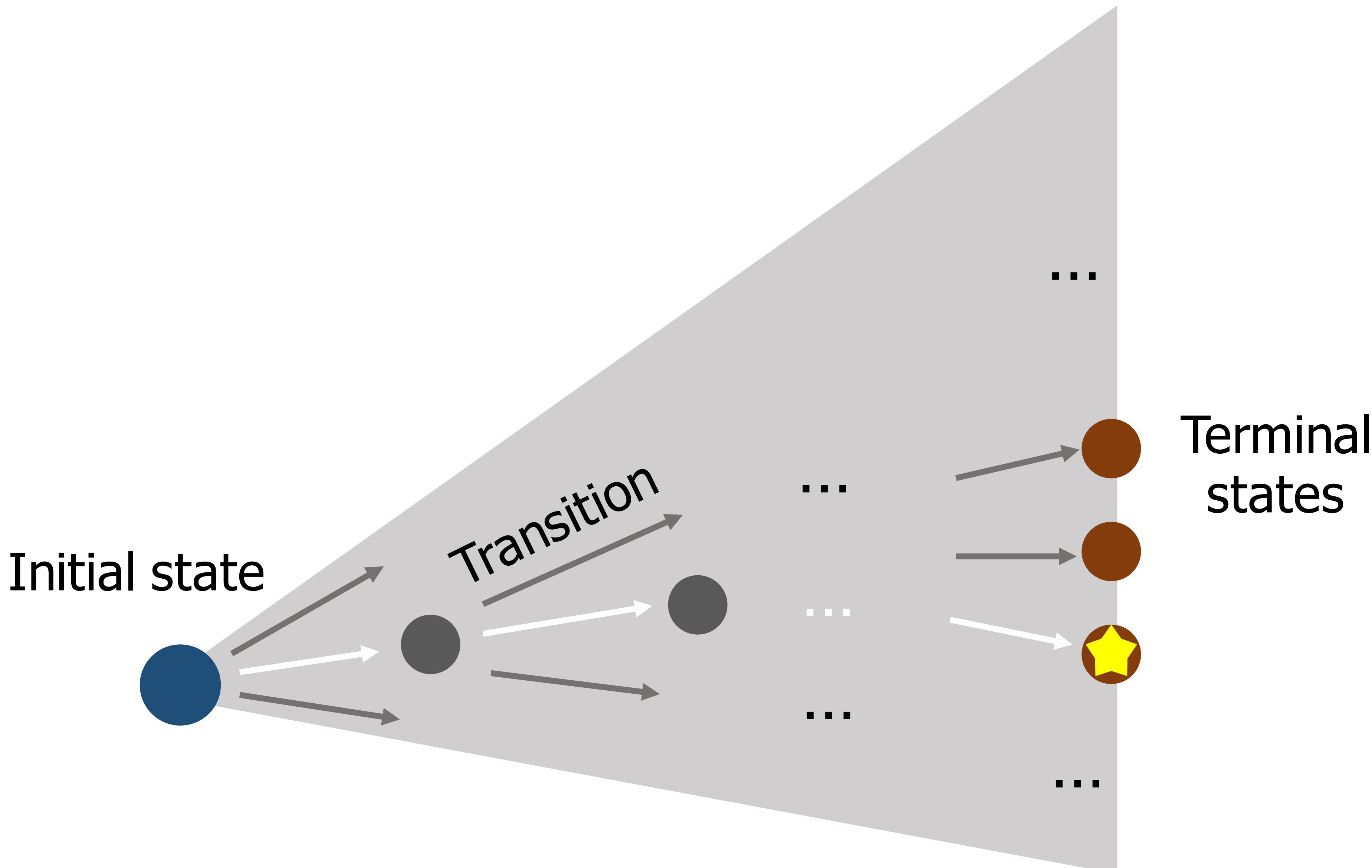
Short Version

- Transition-based dependency parsing has an exponentially-large search space
- $O(n^3)$ exact solutions exist
- In practice, however, we needed rich features $\Longrightarrow O(n^6)$
- (This work) with bi-LSTMs, now we can do $O(n^3)$!
- And we get state-of-the-art results

Dependency Parsing



Transition-based Dependency Parsing

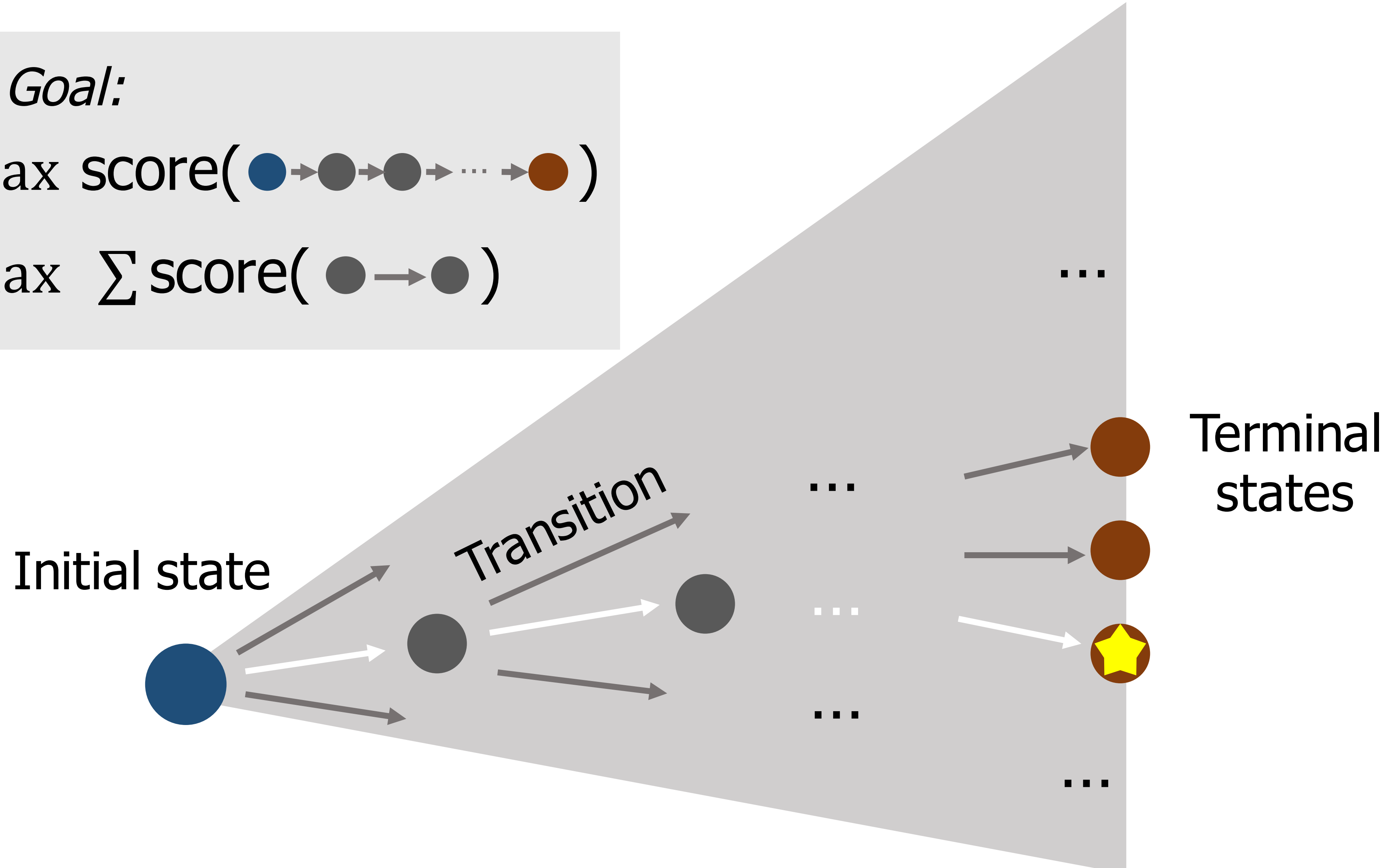


Transition-based Dependency Parsing

Goal:

$\max$ score(● → ● → ● → ⋯ → ●)

$= \max \sum$ score(● → ●)



Exact Decoding

- Dynamic programming (Huang and Sagae, 2010; Kuhlmann, Gómez-Rodríguez and Satta, 2011)

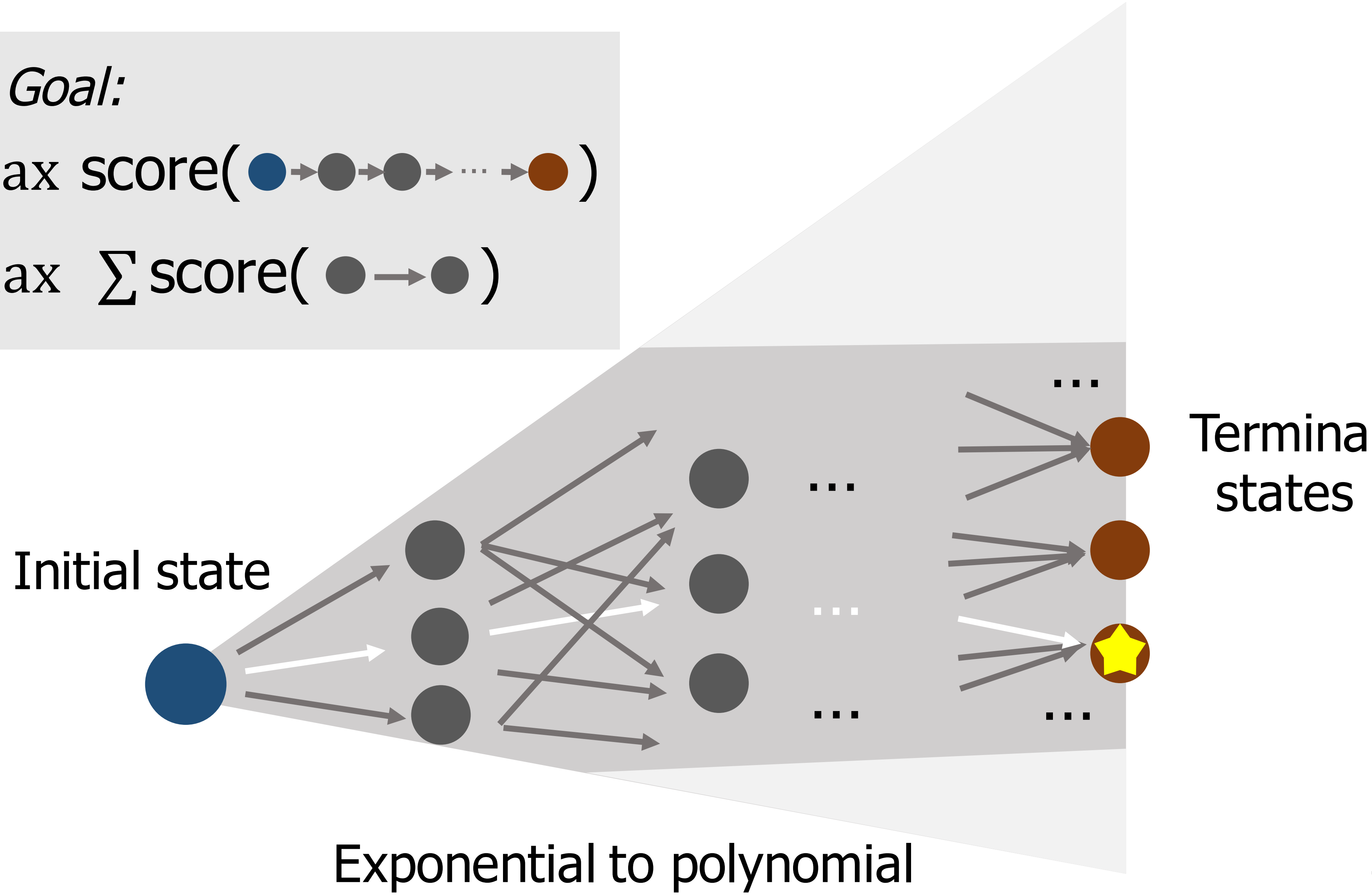
  … since transition (sub-)sequences can be reused

Exact Decoding

Goal:

$$\max \text{score}(\bullet \to \bullet \to \bullet \to \cdots \to \bullet)$$

$$= \max \sum \text{score}(\bullet \to \bullet)$$



Transition Systems

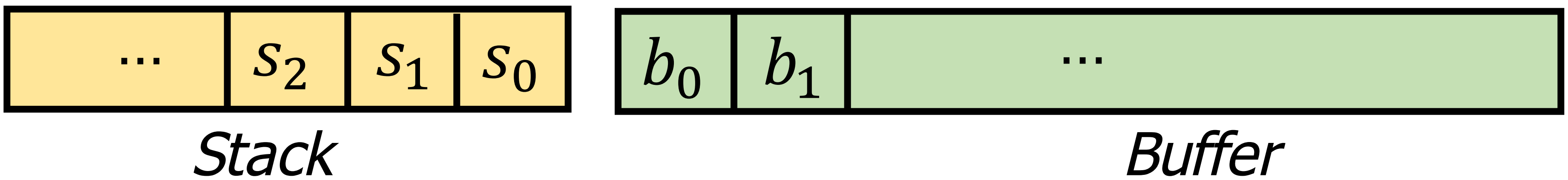
	DP Complexity	# Transitions
Arc-standard	$O(n^4)$	**3**
Arc-eager	$\boldsymbol{O(n^3)}$	4
Arc-hybrid	$\boldsymbol{O(n^3)}$	**3**

In our paper

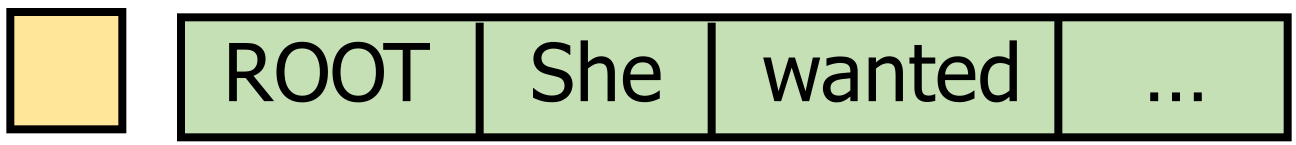
Presentational convenience

Arc-hybrid Transition System

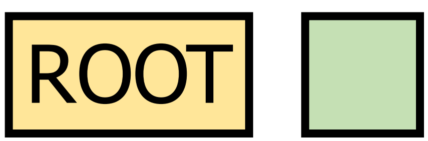
Search State



Initial State



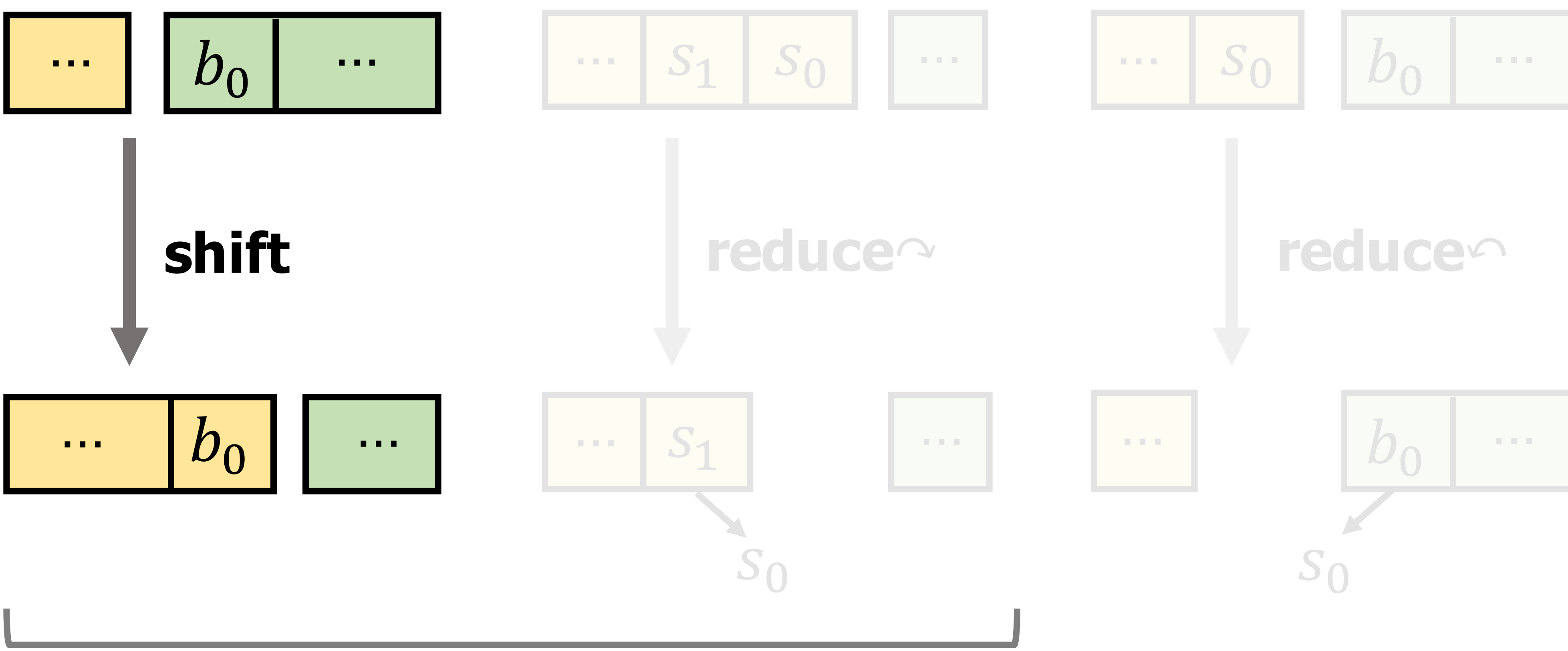
Terminal State



(Yamada and Matsumoto, 2003)
(Gómez-Rodríguez et al., 2008)
(Kuhlmann et al., 2011)

Arc-hybrid Transition System

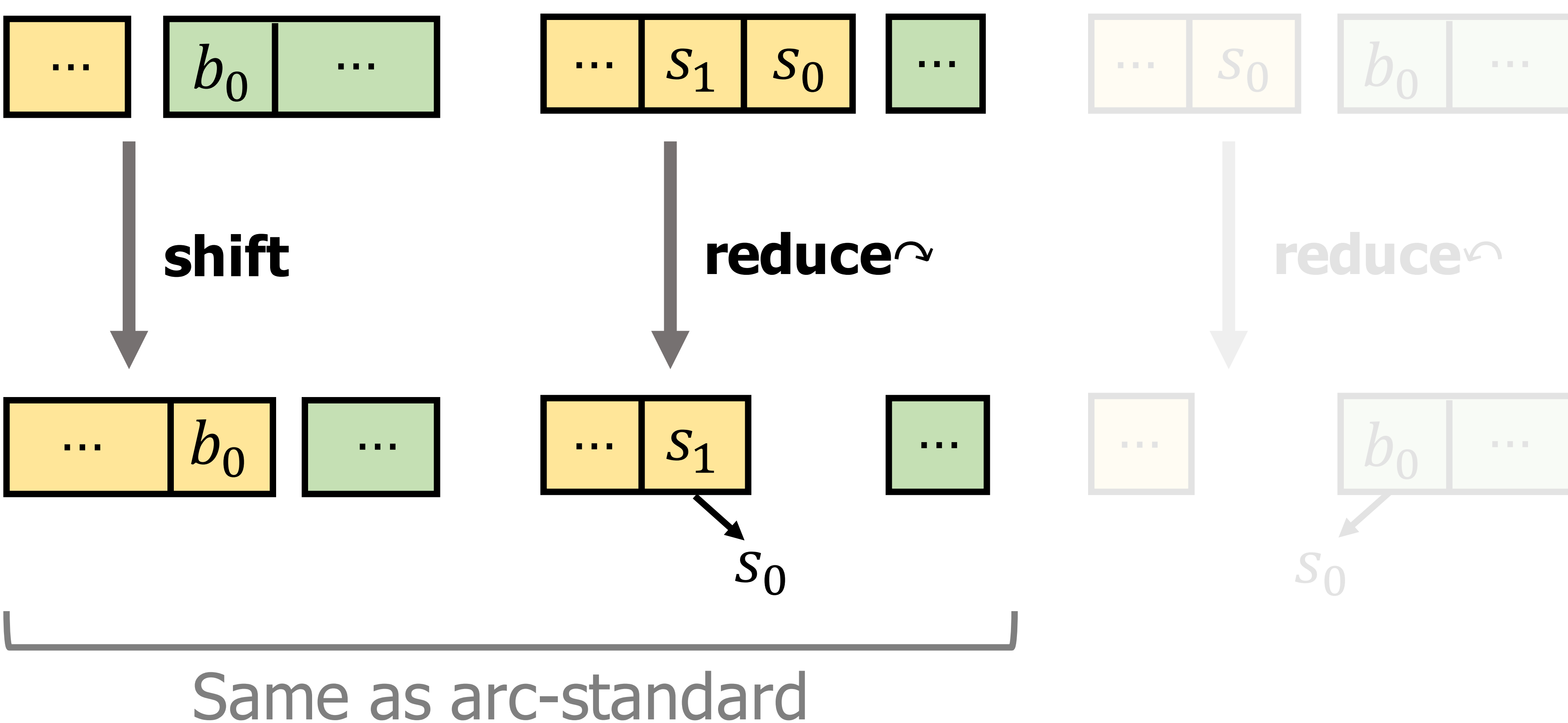
Transitions



Same as arc-standard

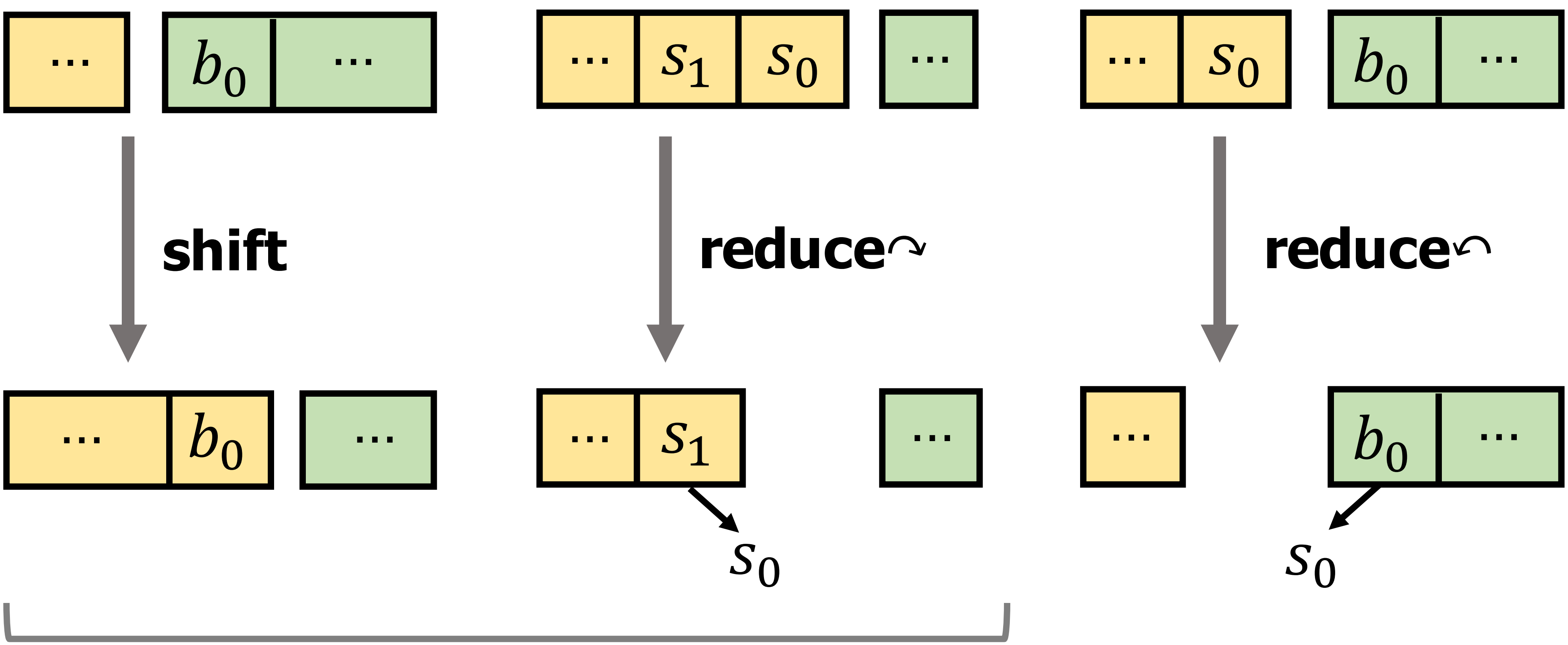
Arc-hybrid Transition System

Transitions



Arc-hybrid Transition System

Transitions



Same as arc-standard

Arc-hybrid Transition System

	Stack	*Buffer*
initial		ROOT · She · wanted · to · eat · an · apple
shift	ROOT	She · wanted · to · eat · an · apple
shift	ROOT · She	wanted · to · eat · an · apple
reduce↶	ROOT	wanted · to · eat · an · apple
		She
shift	ROOT · wanted	to · eat · an · apple
shift	ROOT · wanted · to	eat · an · apple

Arc-hybrid Transition System

	Stack	*Buffer*
initial		ROOT She wanted to eat an apple
shift	ROOT	She wanted to eat an apple
shift	ROOT She	wanted to eat an apple
reduce↶	ROOT	wanted to eat an apple
		She
shift	ROOT wanted	to eat an apple
shift	ROOT wanted to	eat an apple

Arc-hybrid Transition System

	Stack	*Buffer*
initial		ROOT She wanted to eat an apple
shift	ROOT	She wanted to eat an apple
shift	ROOT She	wanted to eat an apple
reduce↶	ROOT	wanted to eat an apple
		She
shift	ROOT wanted	to eat an apple
shift	ROOT wanted to	eat an apple

Arc-hybrid Transition System

	Stack	*Buffer*
initial		ROOT She wanted to eat an apple
shift	ROOT	She wanted to eat an apple
shift	ROOT She	wanted to eat an apple
reduce↶	ROOT	wanted to eat an apple
		She
shift	ROOT wanted	to eat an apple
shift	ROOT wanted to	eat an apple

Arc-hybrid Transition System

	Stack	*Buffer*
initial		ROOT | She | wanted | to | eat | an | apple
shift	ROOT	She | wanted | to | eat | an | apple
shift	ROOT | She	wanted | to | eat | an | apple
reduce↶	ROOT	wanted | to | eat | an | apple
		She
shift	ROOT | wanted	to | eat | an | apple
shift	ROOT | wanted | to	eat | an | apple

Arc-hybrid Transition System

	Stack	*Buffer*
initial		ROOT She wanted to eat an apple
shift	ROOT	She wanted to eat an apple
shift	ROOT She	wanted to eat an apple
reduce↶	ROOT	wanted to eat an apple
		She
shift	ROOT wanted	to eat an apple
shift	ROOT wanted to	eat an apple

Arc-hybrid Transition System

	Stack	*Buffer*
reduce↶	ROOT wanted	eat an apple
		to
shift	ROOT wanted eat	an apple
shift	ROOT wanted eat an	apple
reduce↶	ROOT wanted eat	apple
		an
shift	ROOT wanted eat apple	

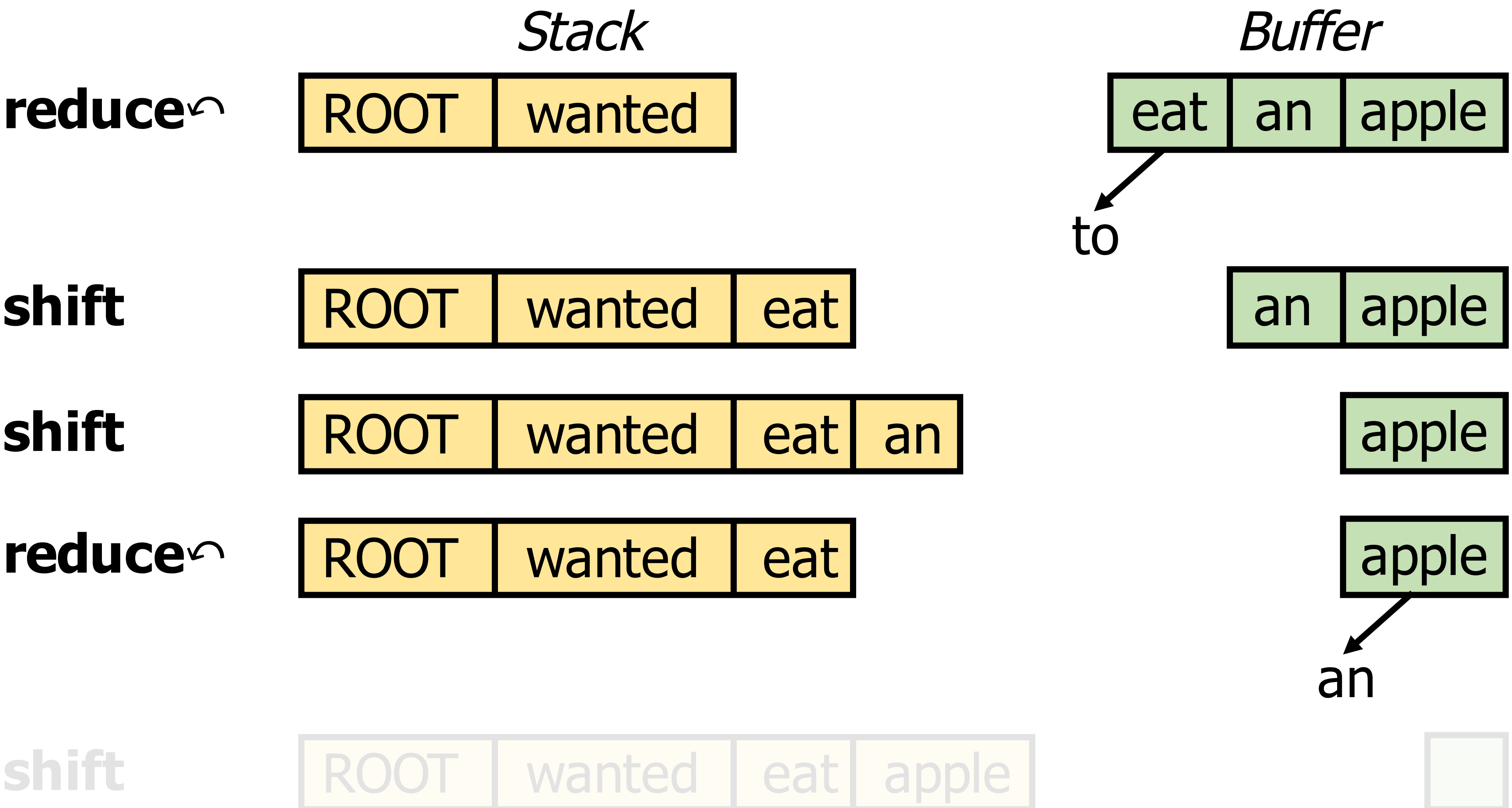
Arc-hybrid Transition System

	Stack	*Buffer*
reduce↶	ROOT | wanted	eat | an | apple
		to
shift	ROOT | wanted | eat	an | apple
shift	ROOT | wanted | eat | an	apple
reduce↶	ROOT | wanted | eat	apple
		an
shift	ROOT | wanted | eat | apple	

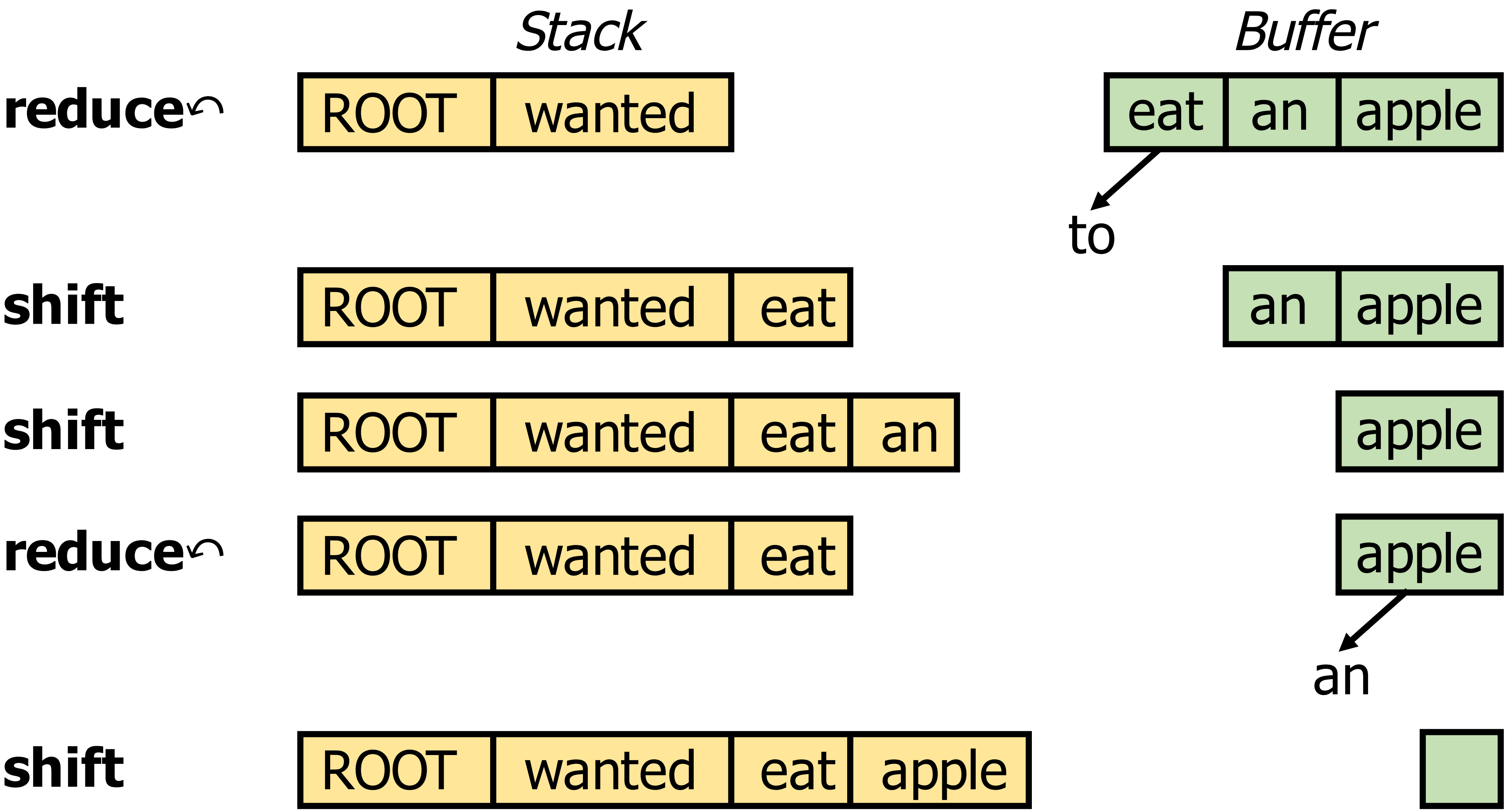
Arc-hybrid Transition System

	Stack	*Buffer*
reduce↶	ROOT | wanted	eat | an | apple (eat → to)
shift	ROOT | wanted | eat	an | apple
shift	ROOT | wanted | eat | an	apple
reduce↶	ROOT | wanted | eat	apple (apple → an)
shift	ROOT | wanted | eat | apple	

Arc-hybrid Transition System



Arc-hybrid Transition System



Arc-hybrid Transition System

Stack *Buffer*

reduce↷ ROOT | wanted | eat

apple

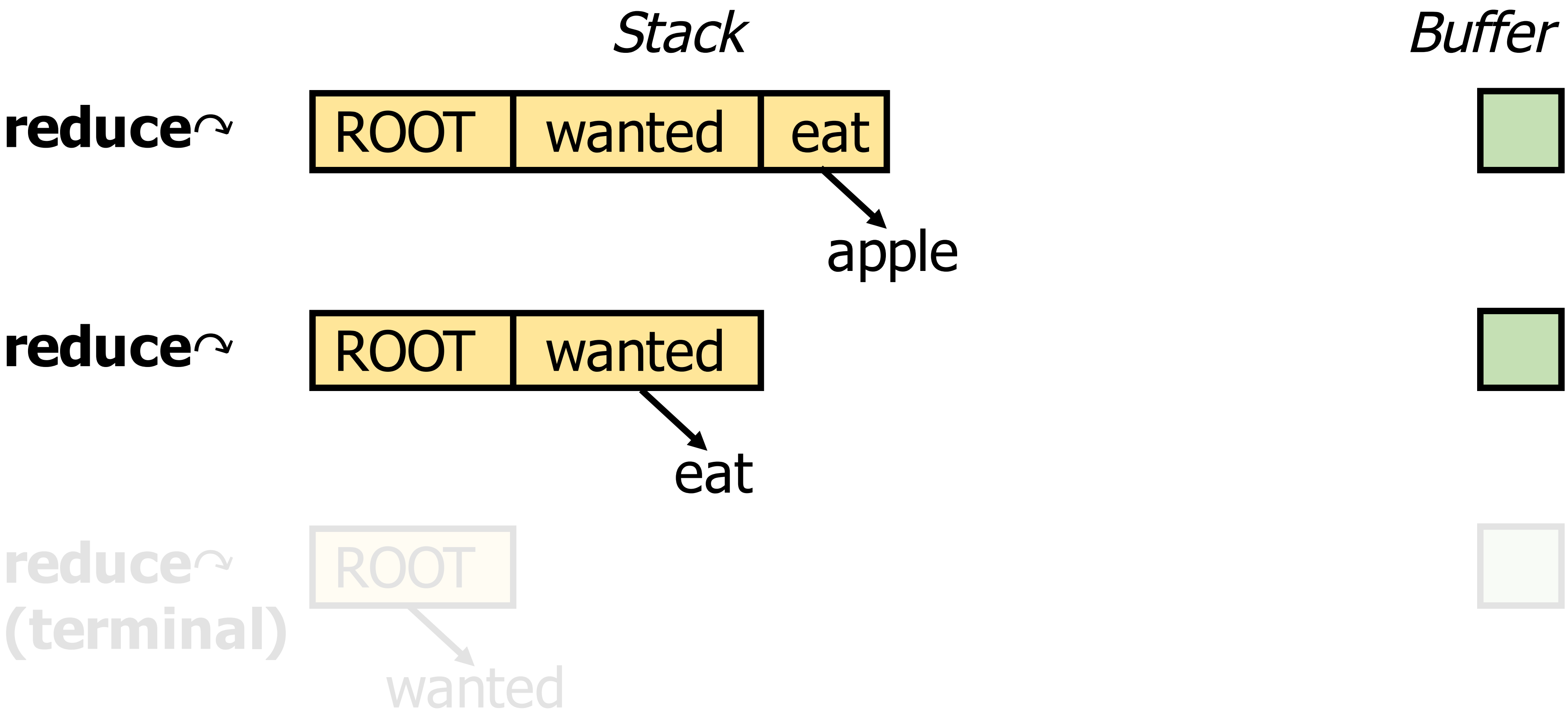
reduce↷ ROOT | wanted

eat

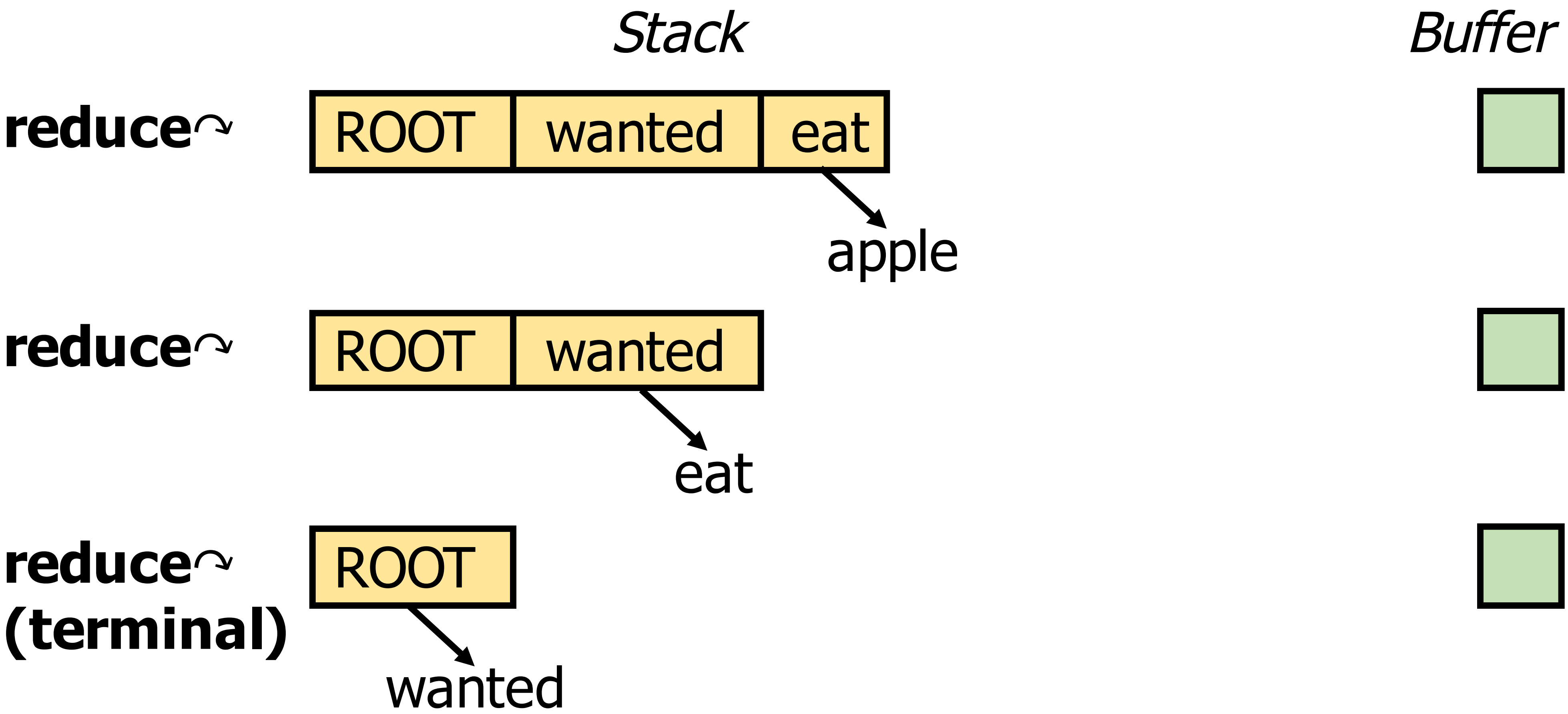
reduce↷ (terminal) ROOT

wanted

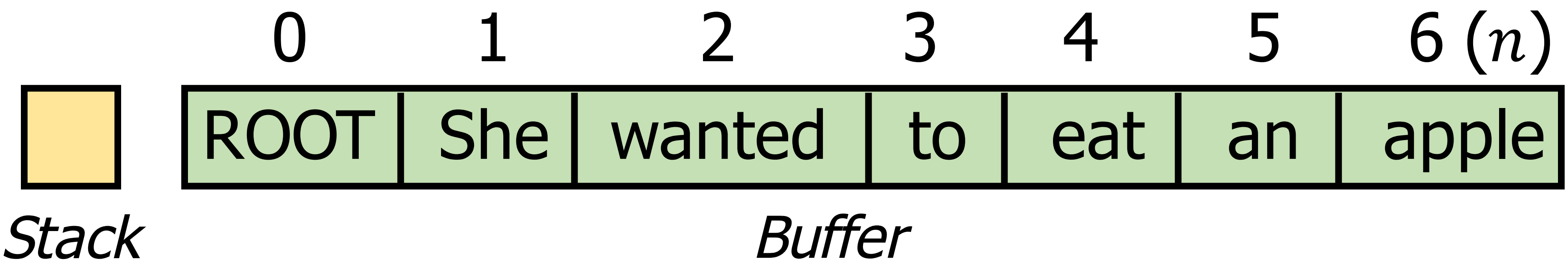
Arc-hybrid Transition System



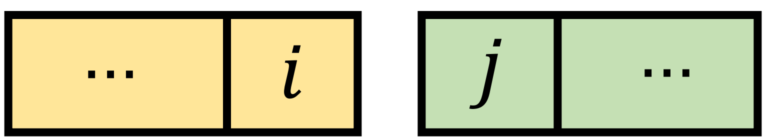
Arc-hybrid Transition System



Deduction System for Arc-hybrid

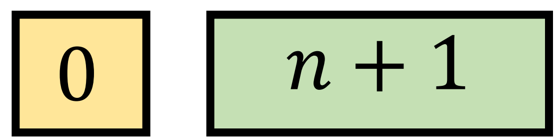


- Deduction Item



$$[i, j]$$

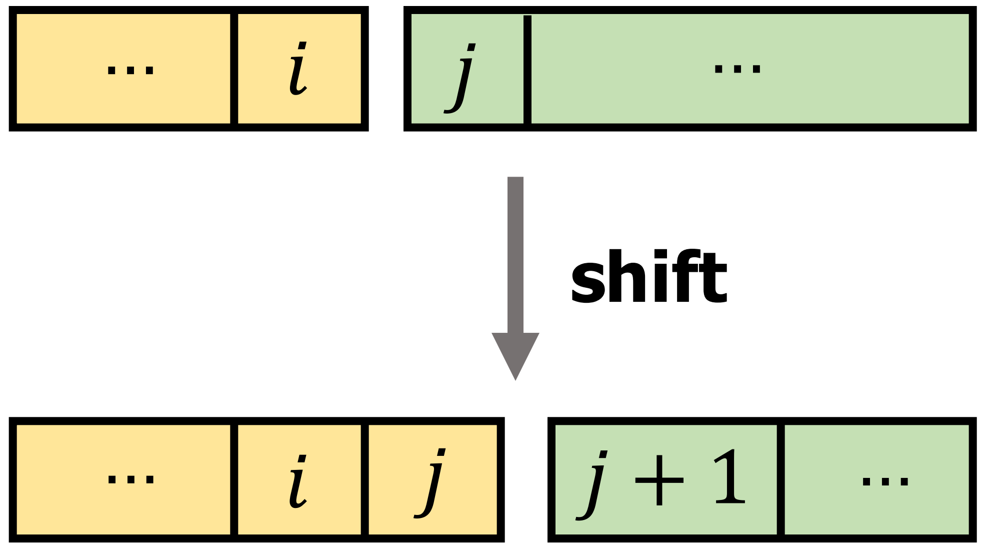
- Goal



$$[0, n+1]$$

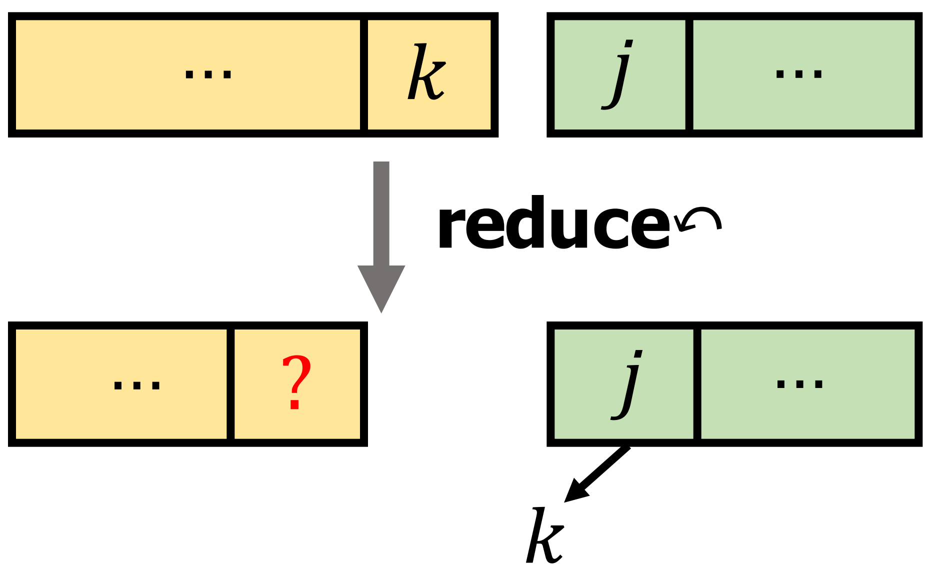
Deduction System for Arc-hybrid

shift $\dfrac{[i,j]}{[j,j+1]}$



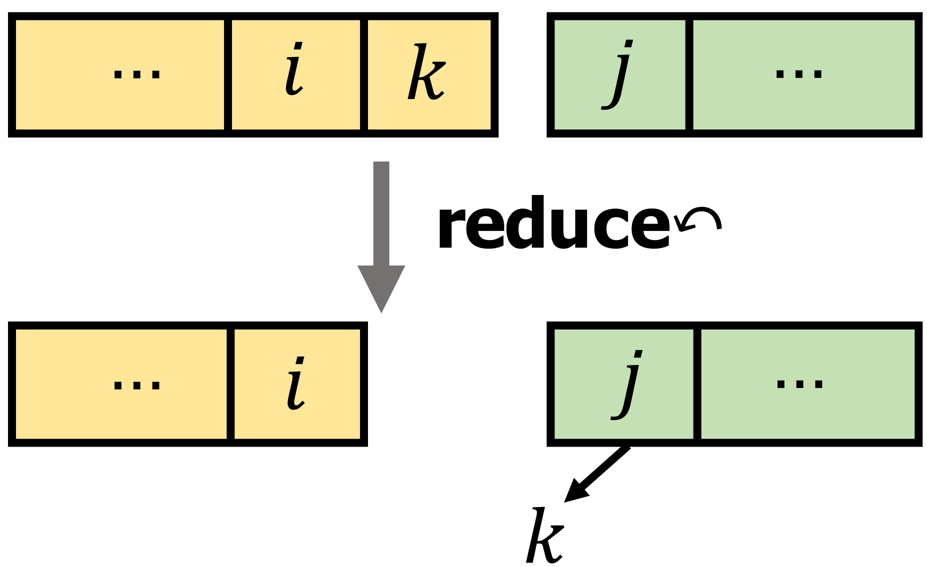
Deduction System for Arc-hybrid

reduce↶ $$\frac{[k, j]}{[?, j]}$$



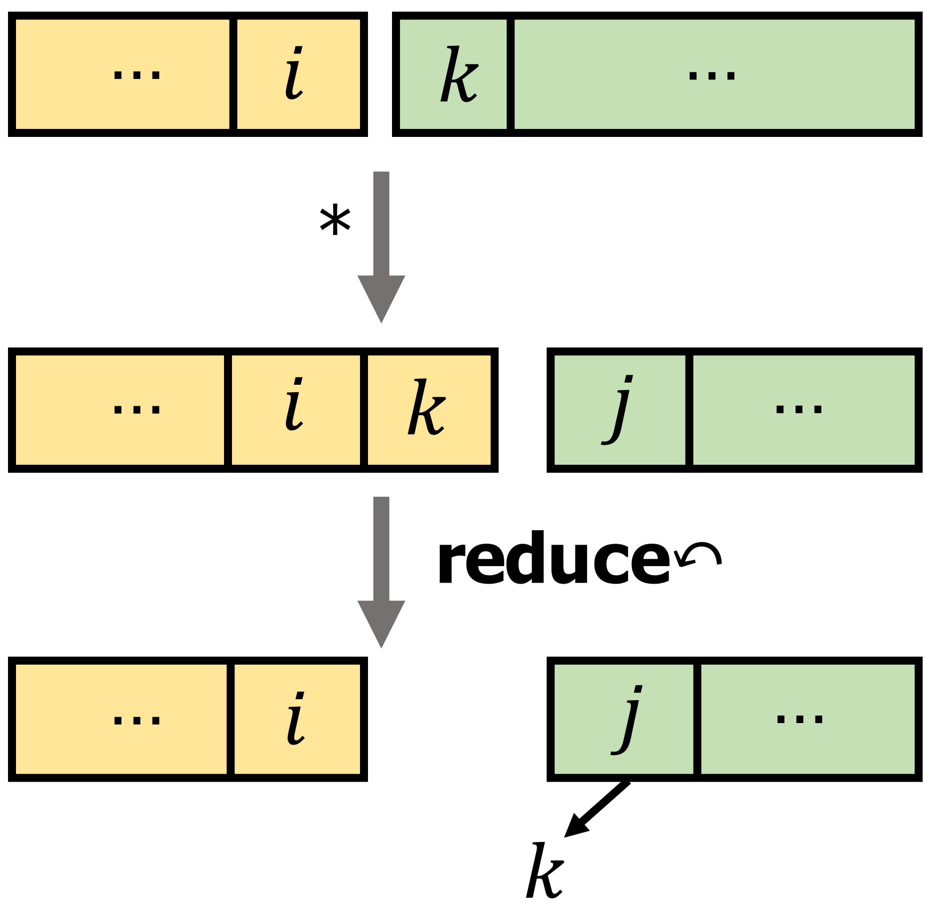
Deduction System for Arc-hybrid

$$\textbf{reduce}{\curvearrowleft} \quad \frac{[k, j]}{[i, j]}$$



Deduction System for Arc-hybrid

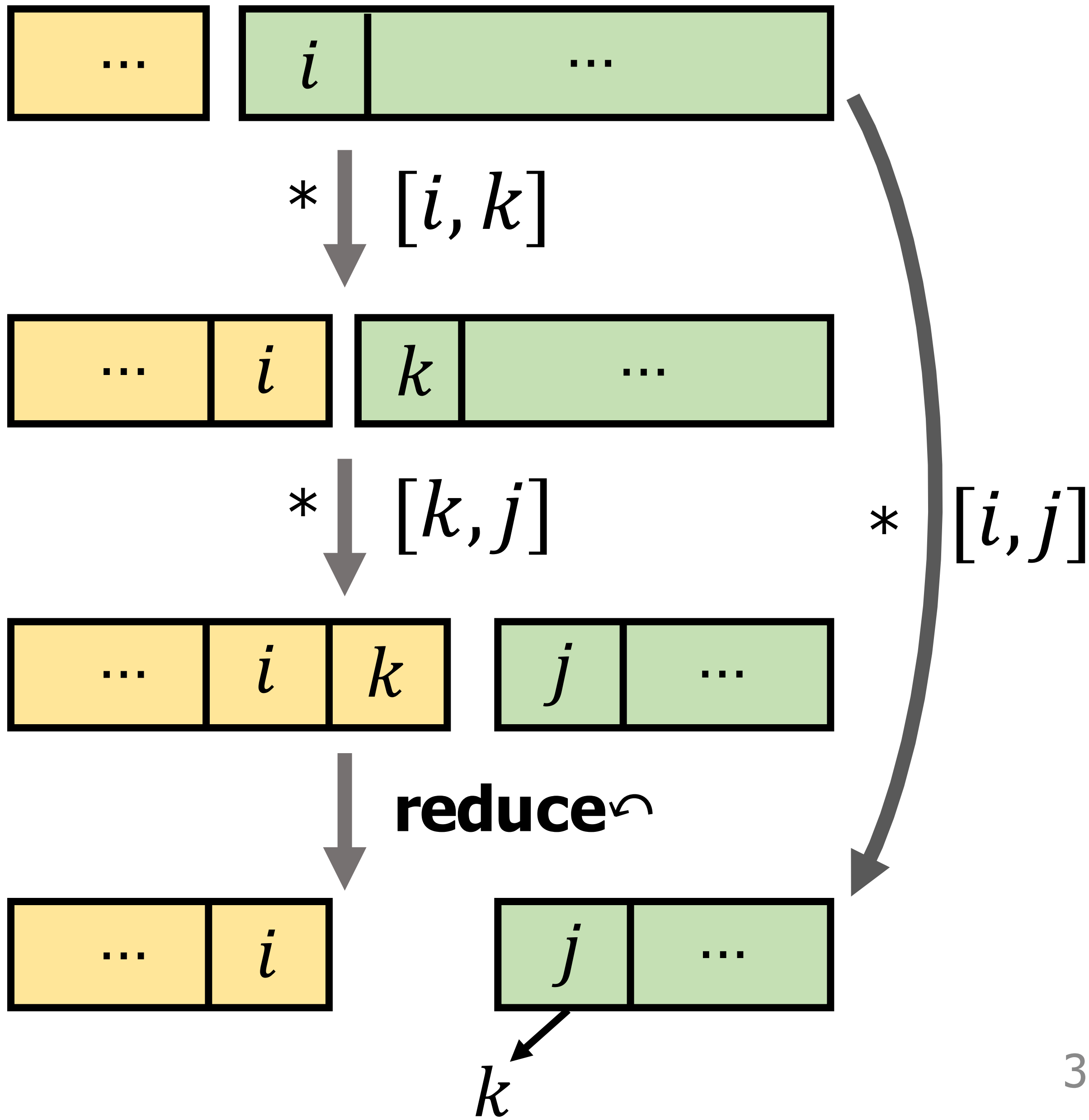
reduce$\curvearrowleft$ $$\frac{[i, k]\ [k, j]}{[i, j]}$$



Deduction System for Arc-hybrid

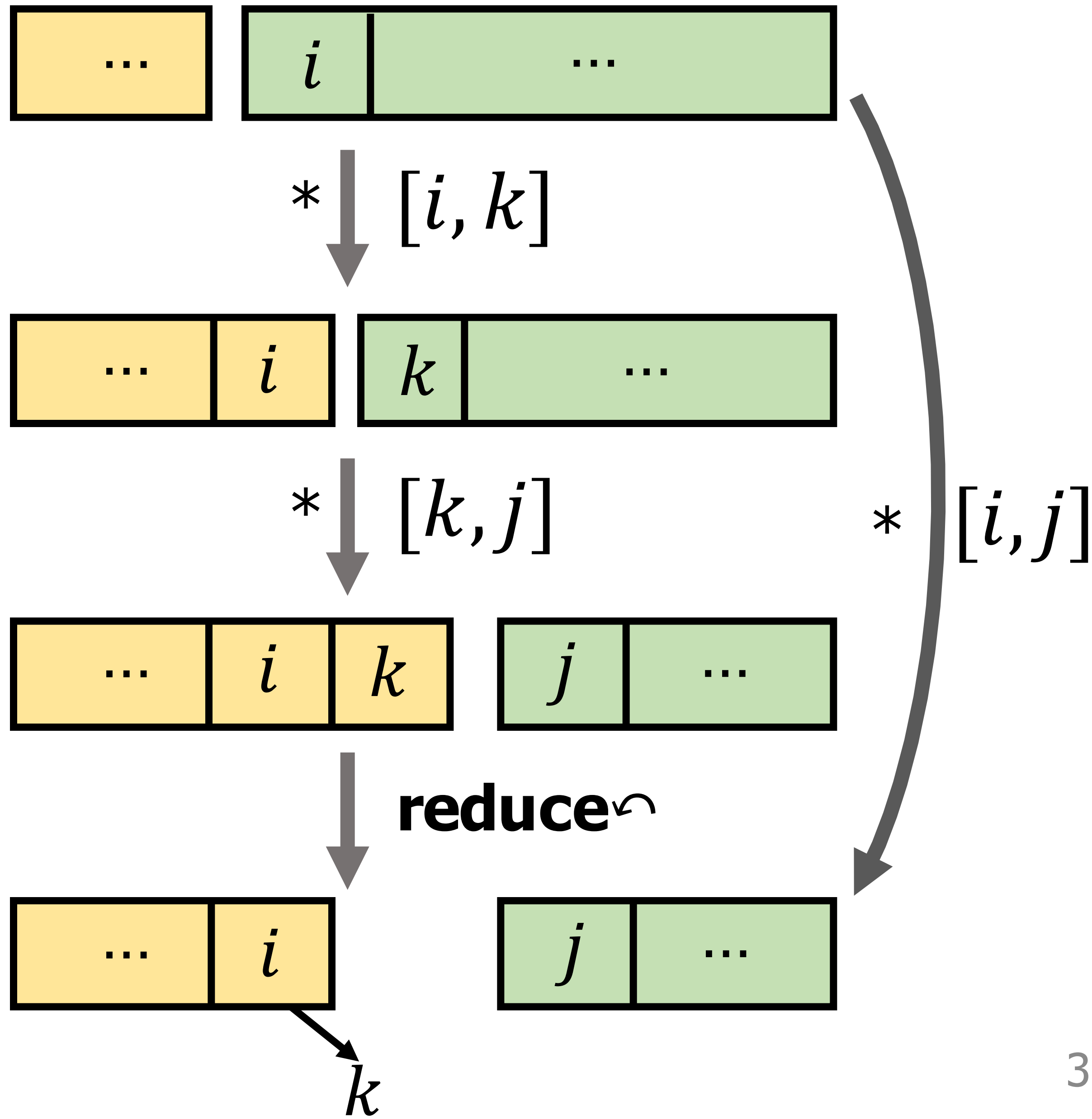
In Kuhlmann et al. (2011)'s notation

reduce↶ $\dfrac{[i,k]\ [k,j]}{[i,j]}$



Deduction System for Arc-hybrid

reduce↷ $\dfrac{[i,k]\ [k,j]}{[i,j]}$



Deduction System for Arc-hybrid

shift $$\frac{[i,j]}{[j,j+1]}$$

Goal: $[0, n+1]$

reduce$\curvearrowleft$ $$\frac{[i,k]\ [k,j]}{[i,j]}\ k \curvearrowleft j$$

$$O(n^3)$$

reduce$\curvearrowright$ $$\frac{[i,k]\ [k,j]}{[i,j]}\ i \curvearrowright k$$

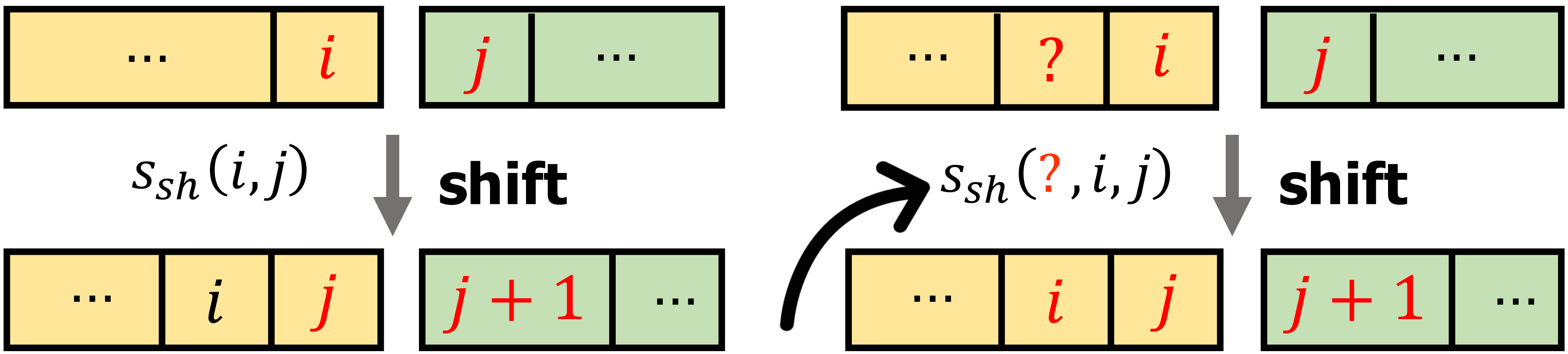
Time Complexity in Practice

- Complexity depends on feature representation!
- Typical feature representation:
  - Feature templates look at specific *positions* in the stack and in the buffer

Time Complexity in Practice

- Compare the following features

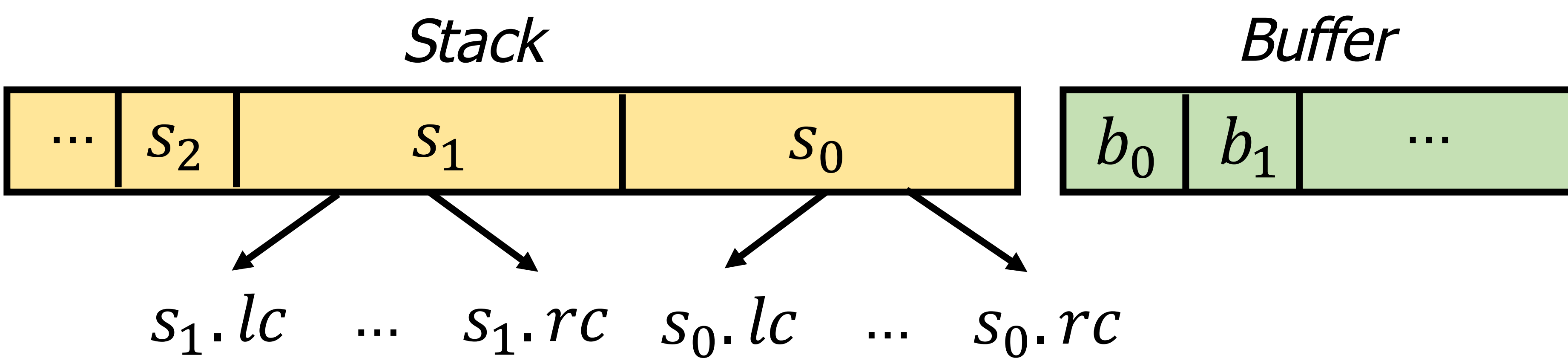
- Time complexities are different!!!



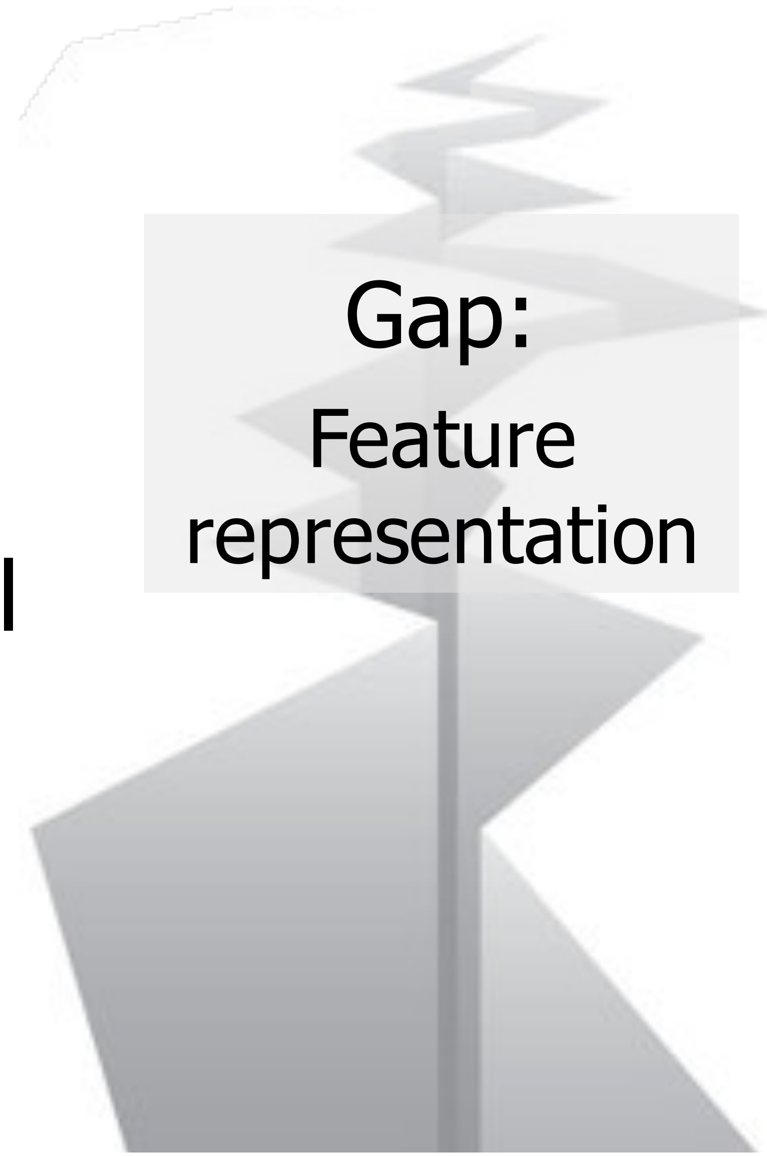
Information about s_1 is not available, needs extra bookkeeping

Time Complexity in Practice

- Complexity depends on feature representation!
- Typical feature representation:
  - Feature templates look at specific *positions* in the stack and in the buffer
- Best-known complexity in practice: $O(n^6)$ (Huang and Sagae, 2010)



Best-known Time Complexities (recap)



$O(n^3)$

Theoretical

Gap:

Feature representation

$O(n^6)$

Practical

In Practice, Instead of Exact Decoding …

- Greedy search (Nivre, 2003, 2004, 2008; Chen and Manning, 2014)
- Beam search (Zhang and Clark, 2011; Weiss et al.,2015)
- Best-first search (Sagae and Lavie, 2006; Sagae and Tsujii, 2007; Zhao et al., 2013)
- Dynamic oracles (Goldberg and Nivre, 2012, 2013)
- "Global" normalization on the beam (Zhou et al., 2015; Andor et al., 2016)
- Reinforcement learning (Lê and Fokkens, 2017)
- Learning to search (Daumé III and Marcu, 2005; Chang et al., 2016; Wiseman and Rush, 2016)
- …

How Many Positional Features Do We Need?

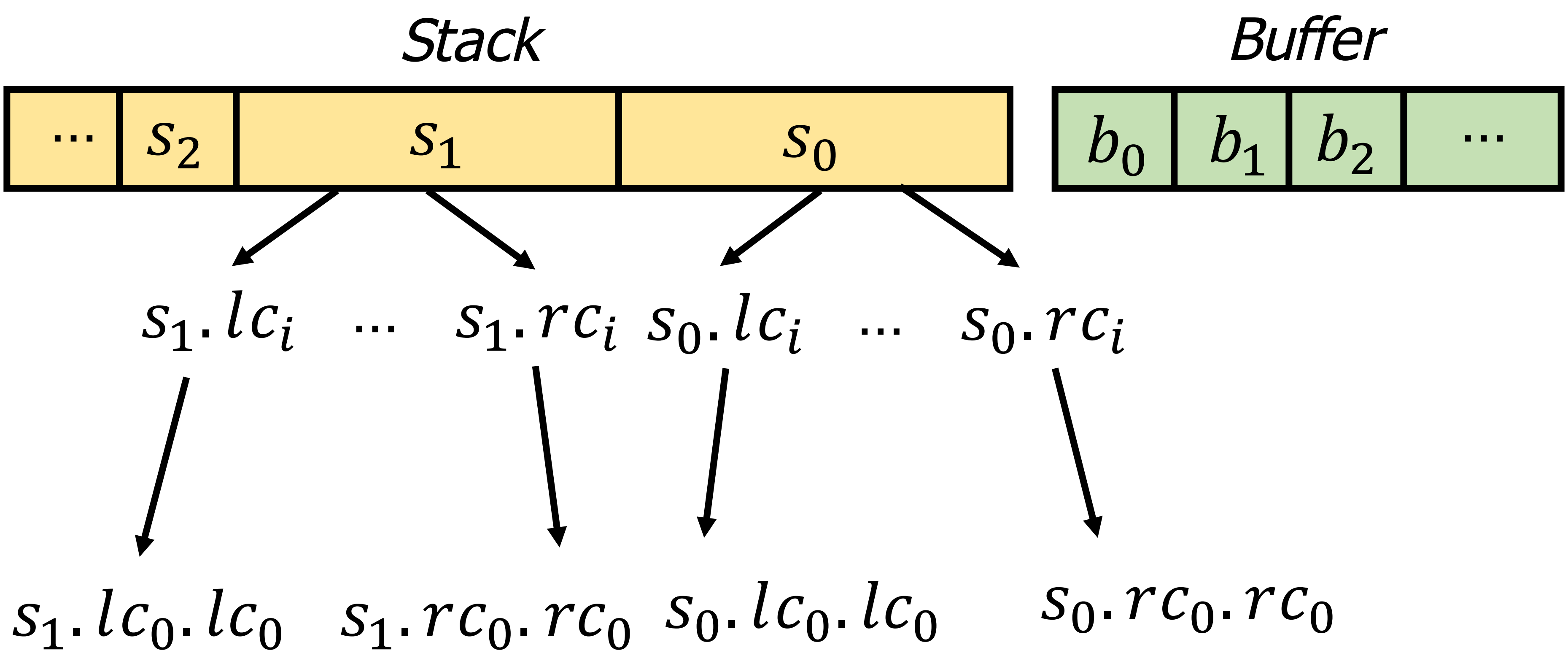
Non-neural (manual engineering)

↓

Chen and Manning (2014)

How Many Positional Features Do We Need?

- Chen and Manning (2014)



How Many Positional Features Do We Need?

Non-neural (manual engineering)

Chen and Manning (2014)

Stack LSTM
(Dyer et al., 2016)

Bi-LSTM
Kiperwasser and Goldberg (2016)

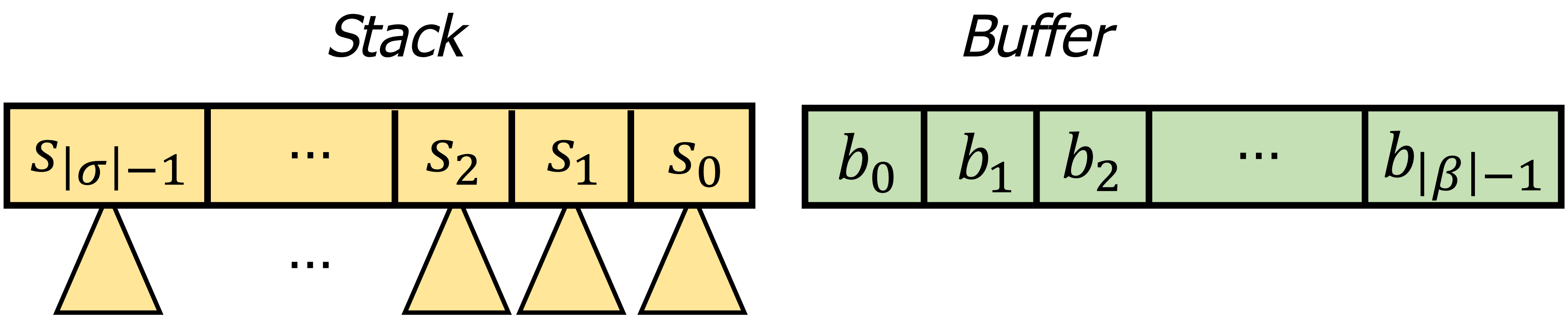
Cross and Huang (2016)

More tree-structure information

Exponential DP   Slow DP   Fast DP

How Many Positional Features Do We Need?

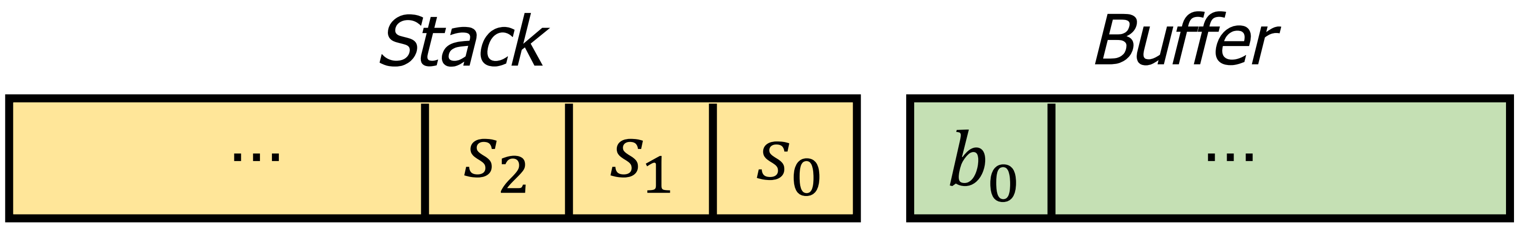
- LSTMs can be used to encode the entire stack and buffer (Dyer et al., 2016)



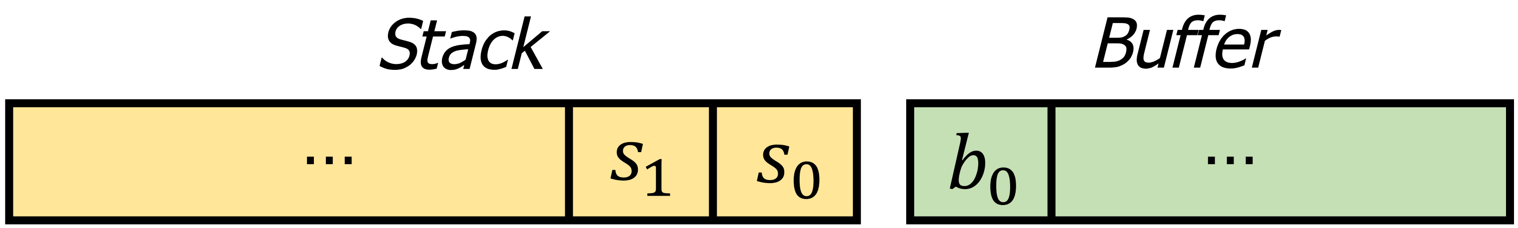
How Many Positional Features Do We Need?

- Bi-LSTMs give compact feature representations (Kiperwasser and Goldberg, 2016; Cross and Huang, 2016)

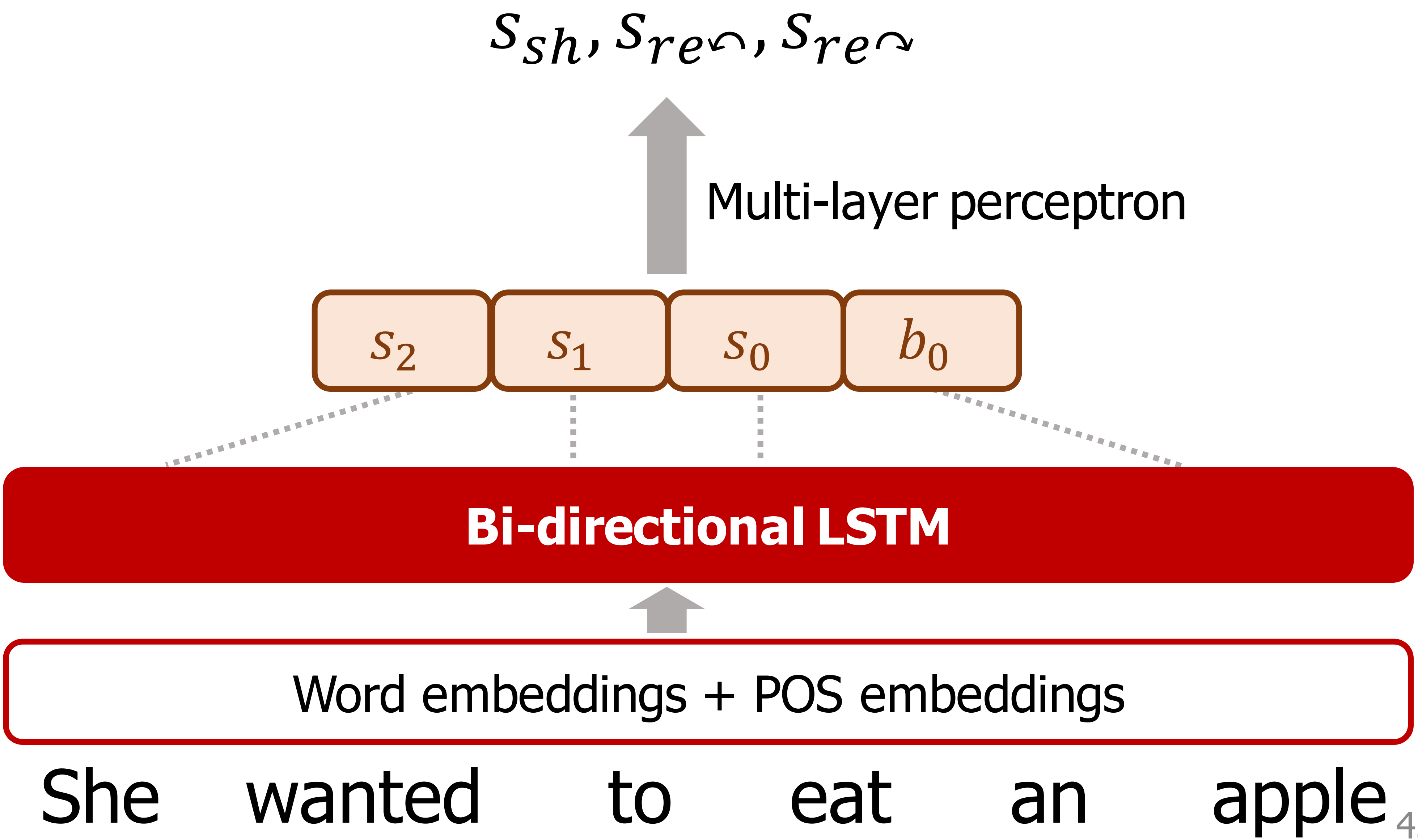
- Features used in Kiperwasser and Goldberg (2016)



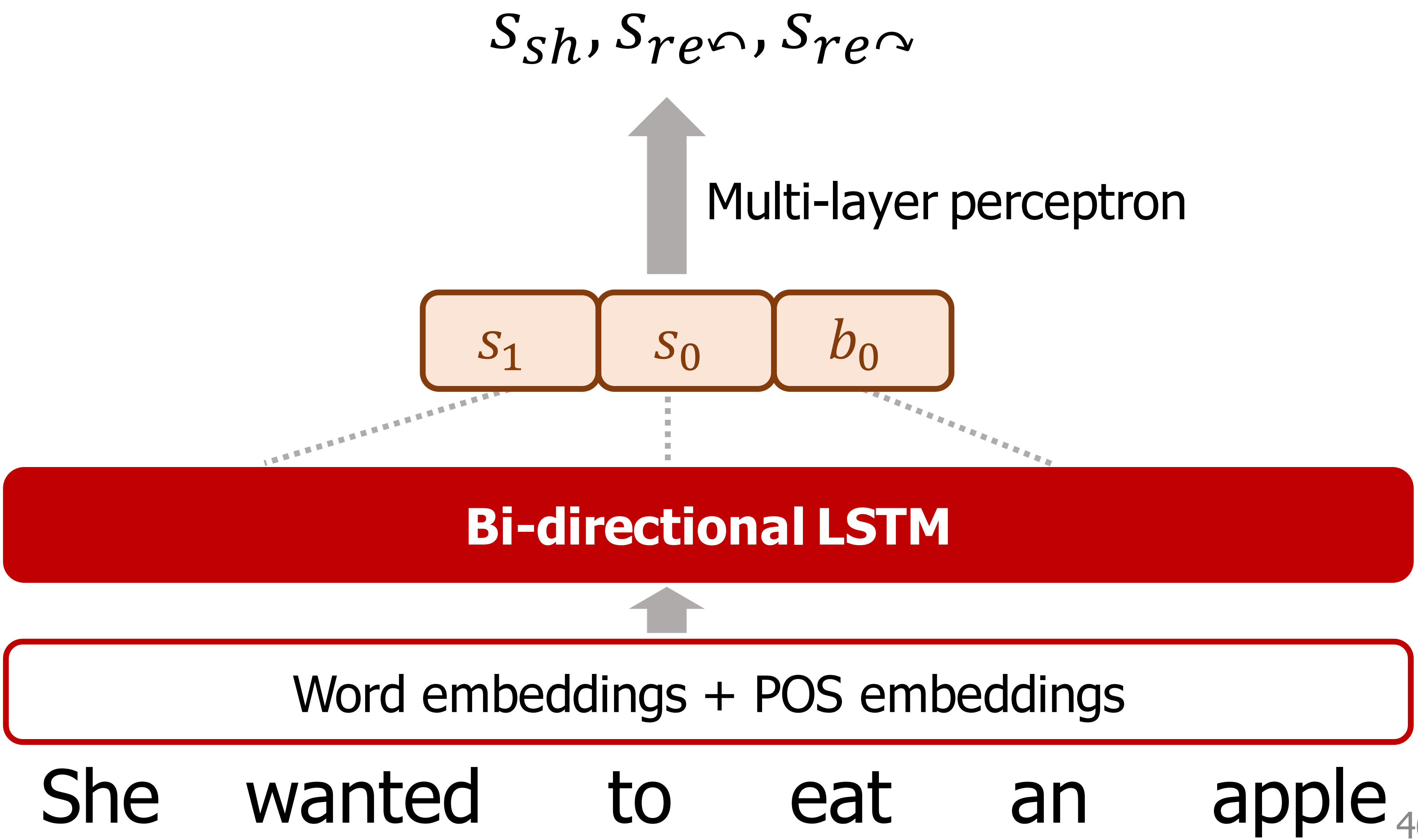
- Features used in Cross and Huang (2016)



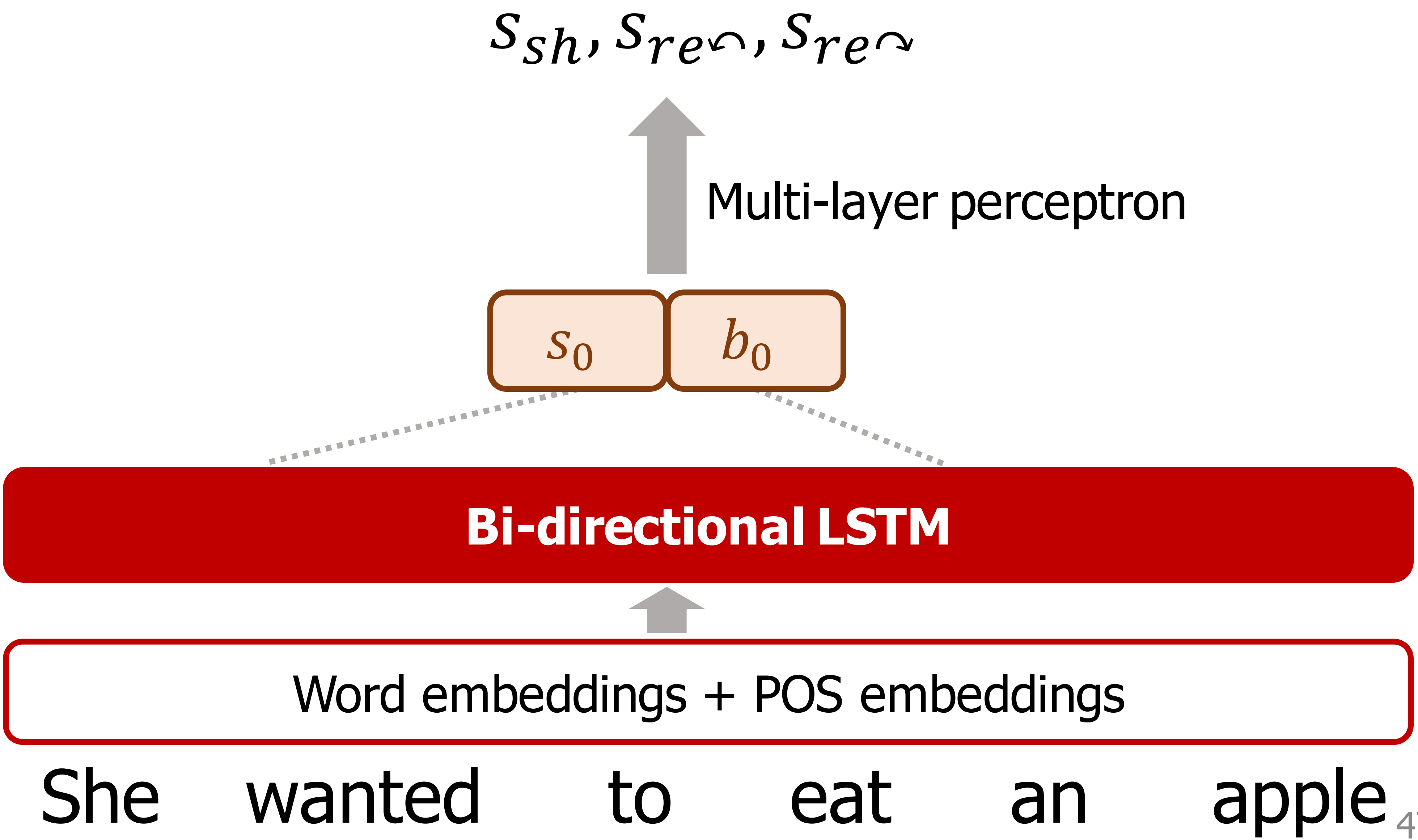
Model Architecture



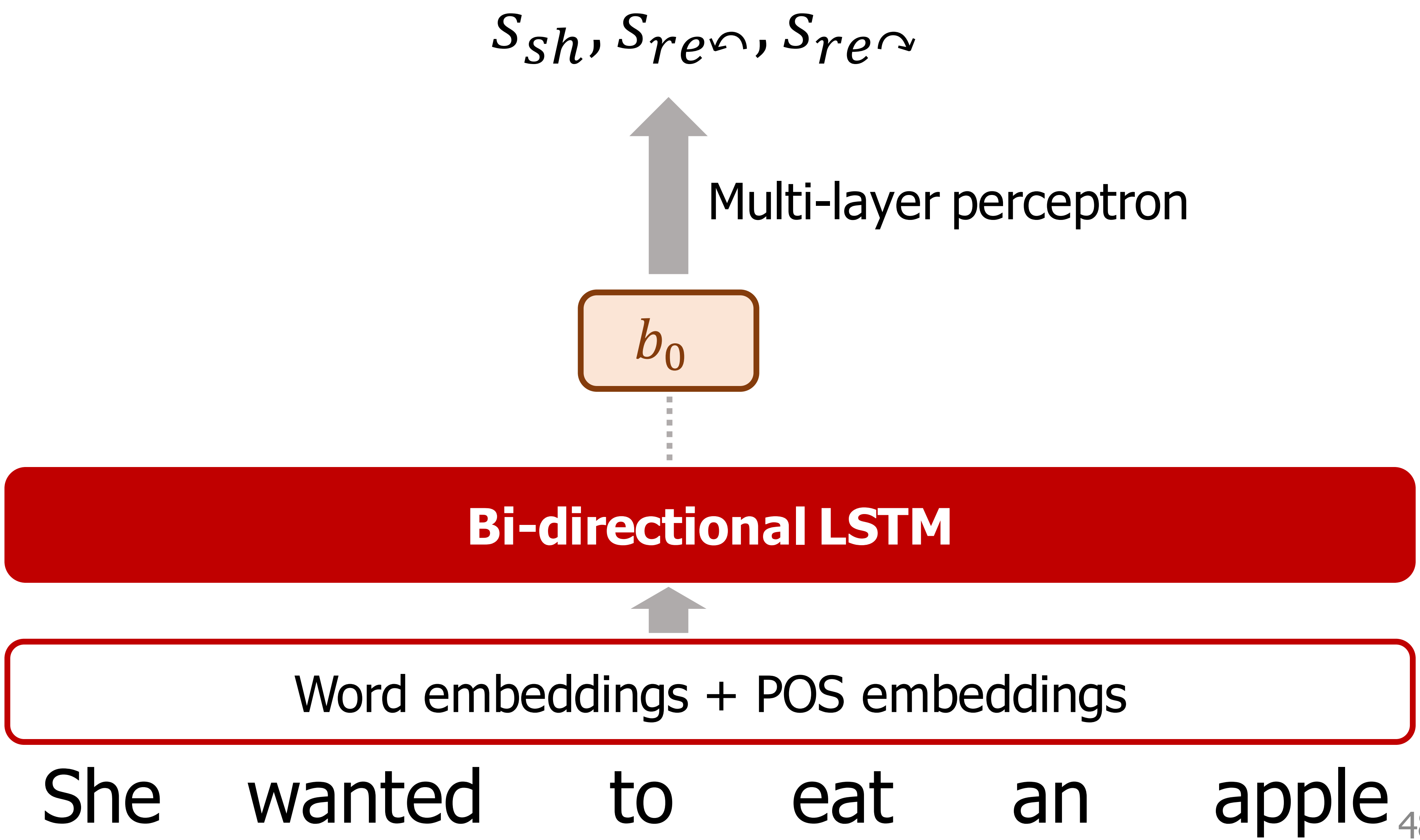
Model Architecture



Model Architecture



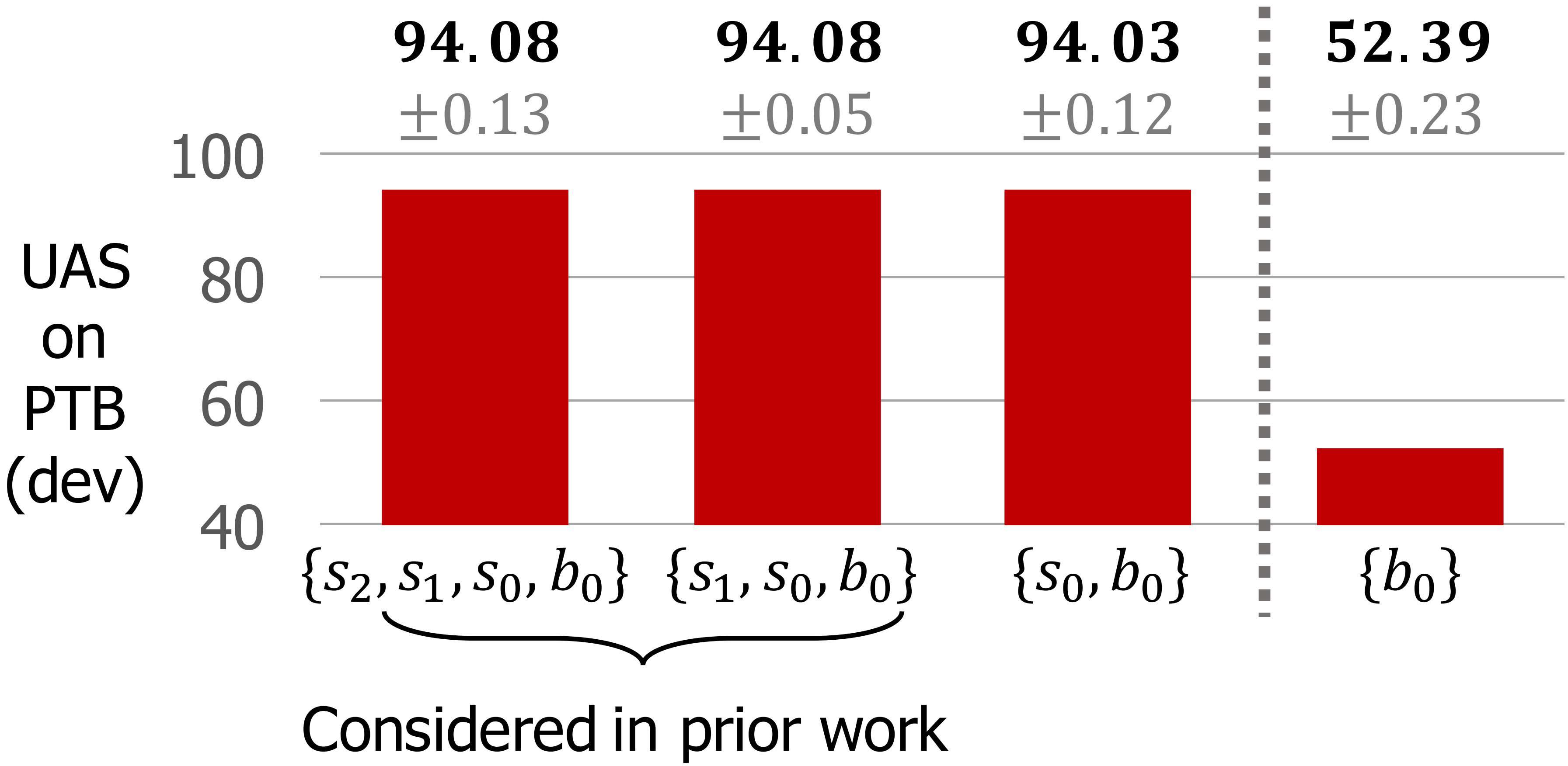
Model Architecture



How Many Positional Features Do We Need?

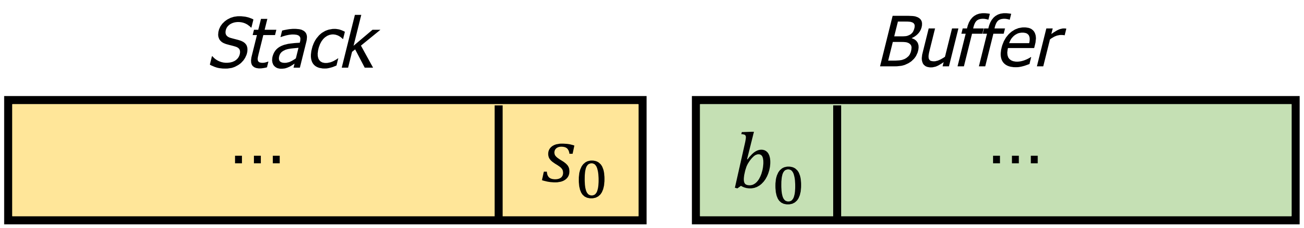
- We answer the question empirically

  … experimented with greedy decoding
- Two positional feature vectors are enough!

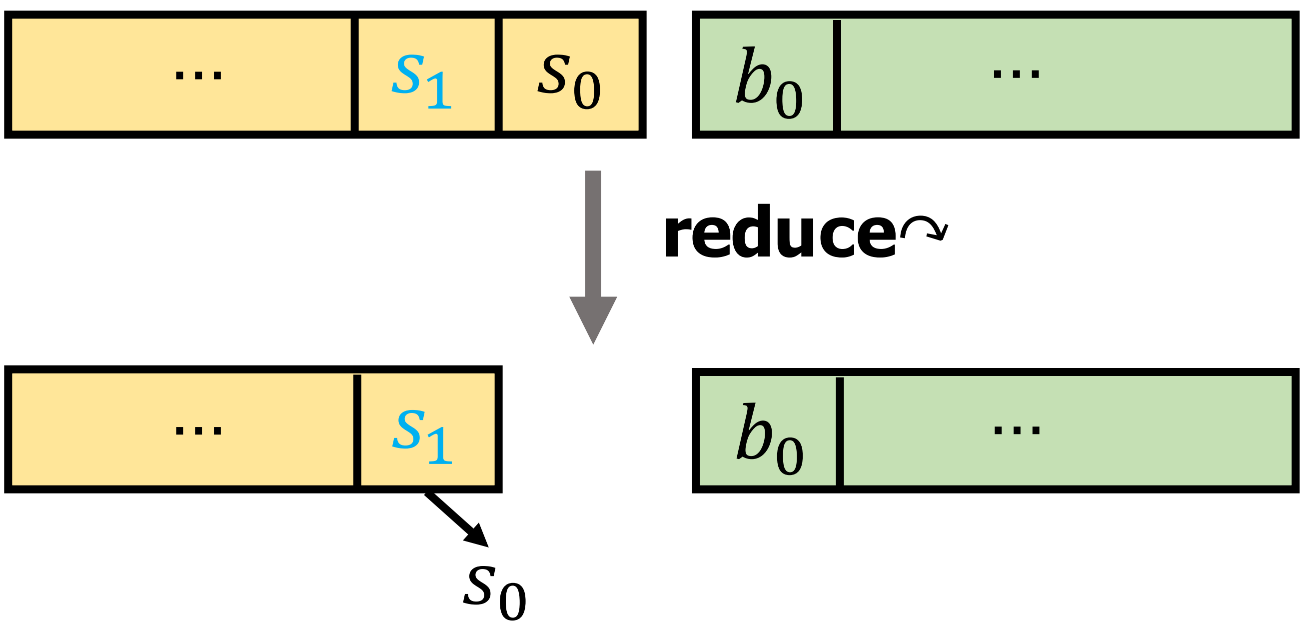


How Many Positional Features Do We Need?

- Our minimal feature set works



- Counter-intuitive, but works for *greedy decoding*



Implication of Minimal Feature Set

- The bare deduction items already contain enough information to extract features
- We don't need extra book keeping
- Leads to the first $O(n^3)$ implementation of global decoders!

How Many Positional Features Do We Need?

Non-neural (manual engineering)

Chen and Manning (2014)

Stack LSTM
(Dyer et al., 2016)

Bi-LSTM
Kiperwasser and Goldberg (2016)

Cross and Huang (2016)

Our work

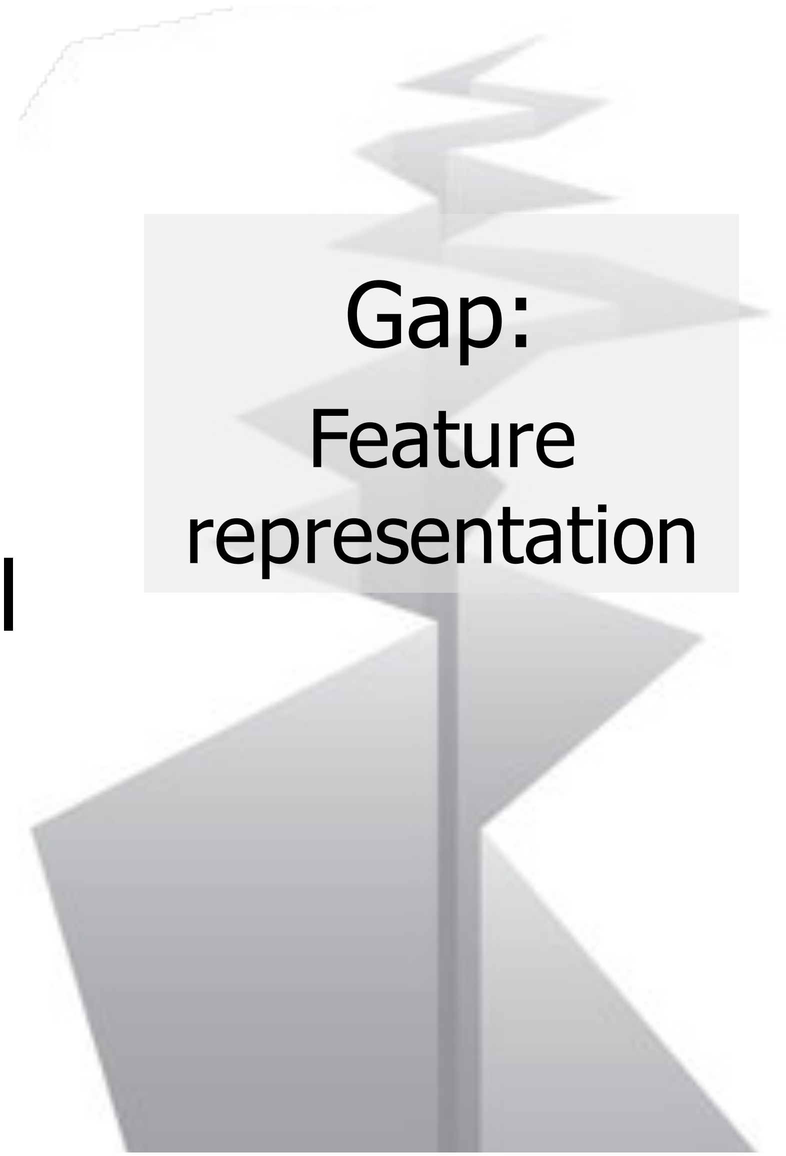
More tree-structure information

Exponential DP — Slow DP — Fast DP — ***Fast(er) DP***

Best-known Time Complexities (recap)

$O(n^3)$

Theoretical



Gap:

Feature representation

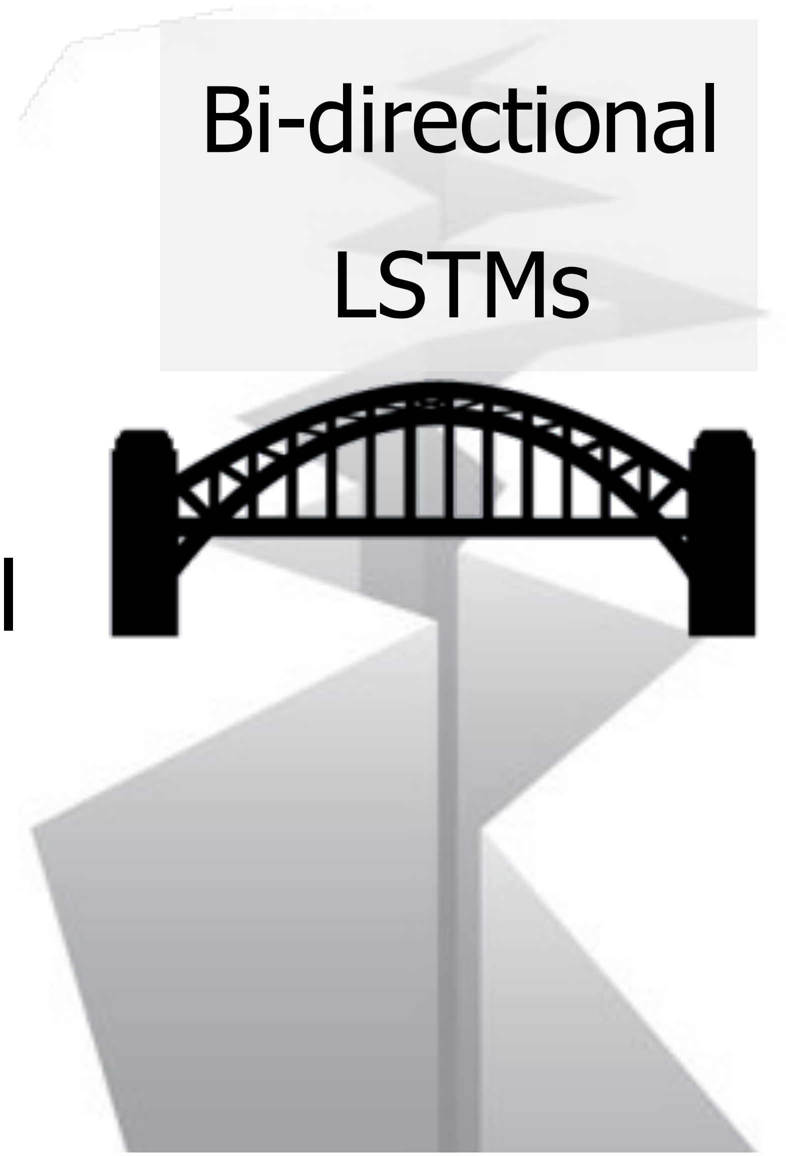
$O(n^6)$

Practical

Our contribution

$O(n^3)$

Theoretical



$O(n^3)$

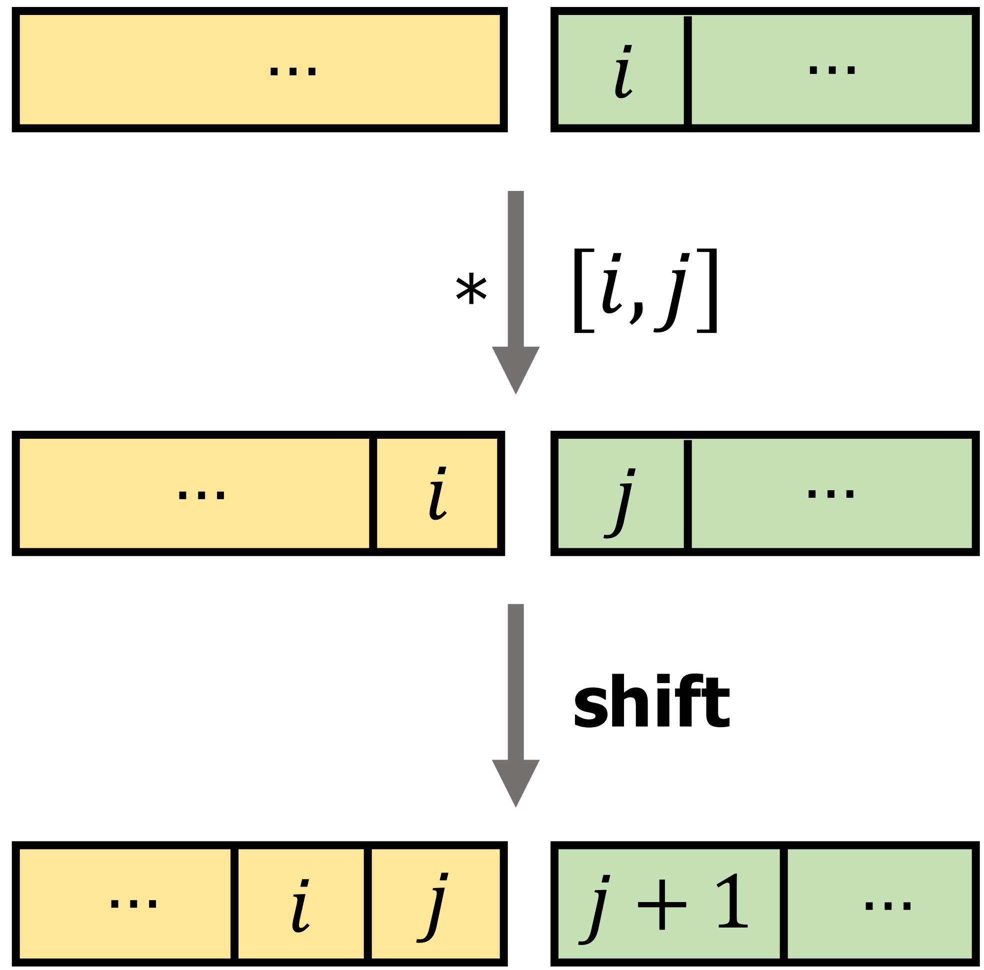
$\cancel{O(n^6)}$

Practical

Decoding

Score of the sub-sequence

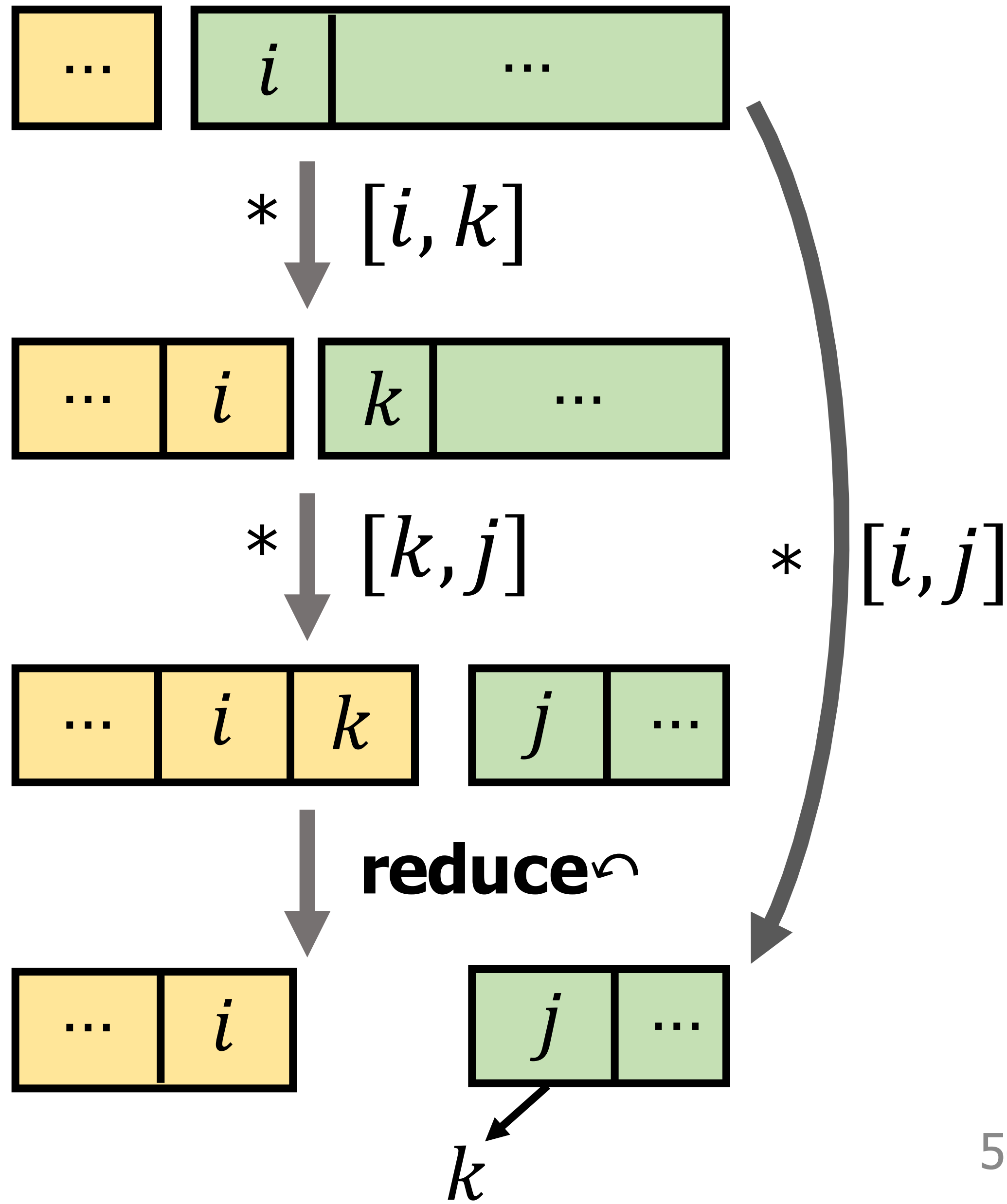
$$\textbf{shift}\ \frac{[i, j]: v}{[j, j+1]: 0}$$



Decoding

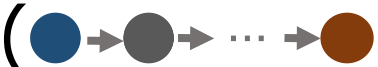
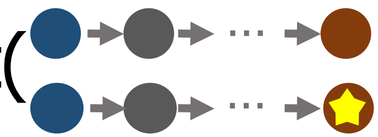
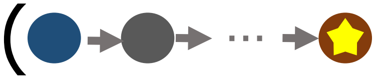
$$\textbf{reduce}\curvearrowleft \frac{[i,k]\colon v_1 \quad [k,j]\colon v_2}{[i,j]\colon v_1 + v_2 + \Delta}$$

$$\Delta = s_{sh}(i,k) + s_{re\curvearrowleft}(k,j)$$

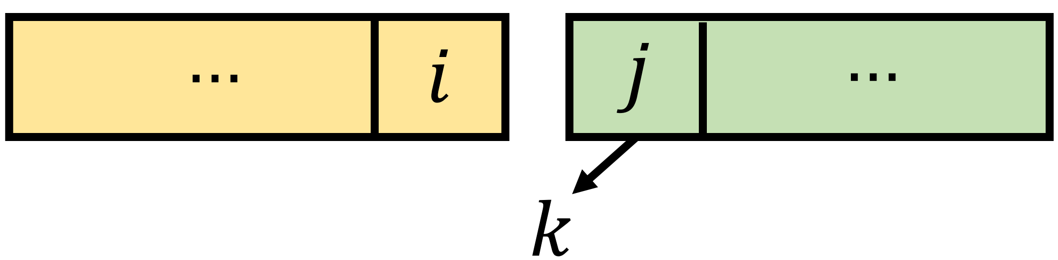


Training

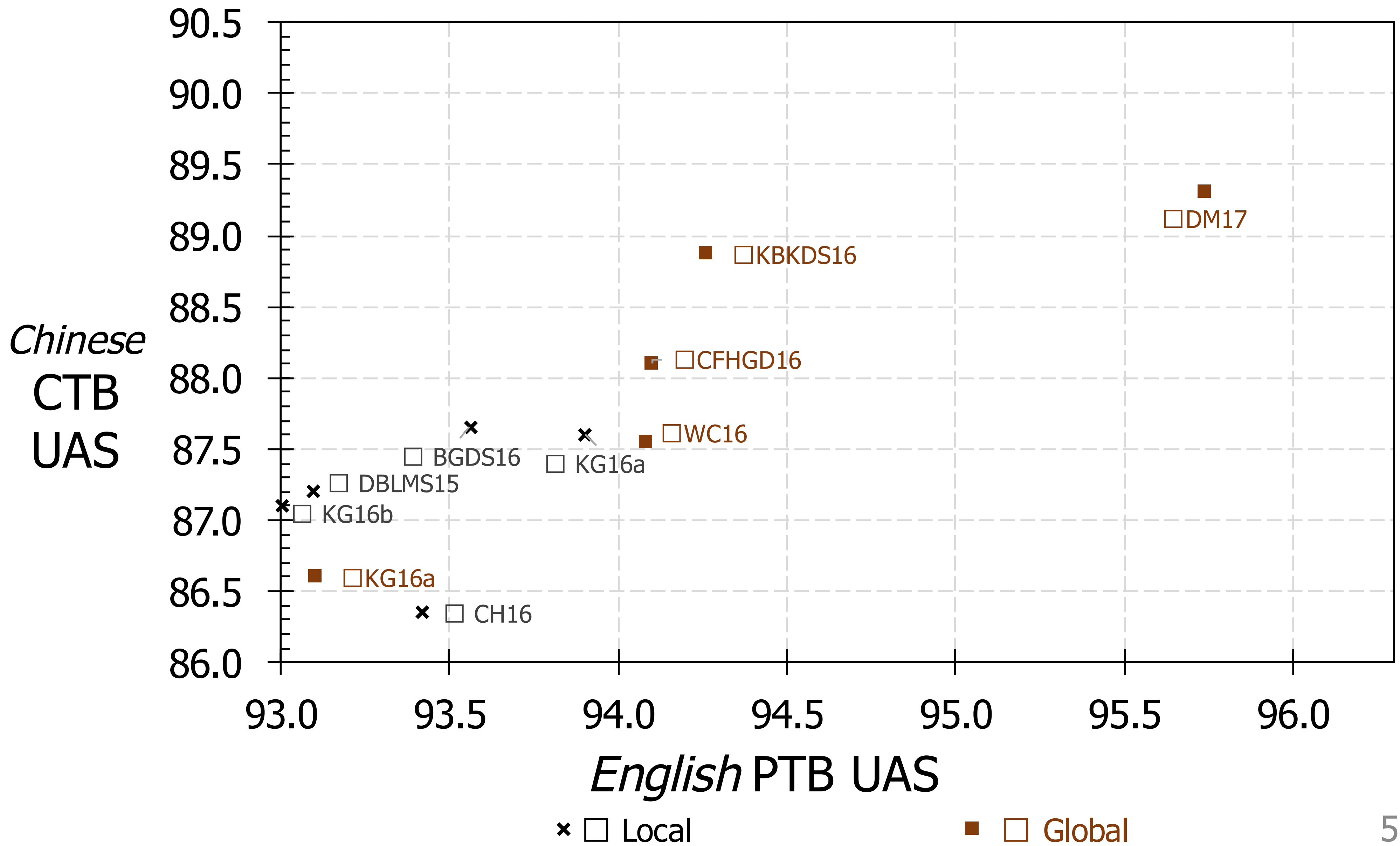
- Cost-augmented decoding (Taskar et al., 2005)

max score() + cost() - score()

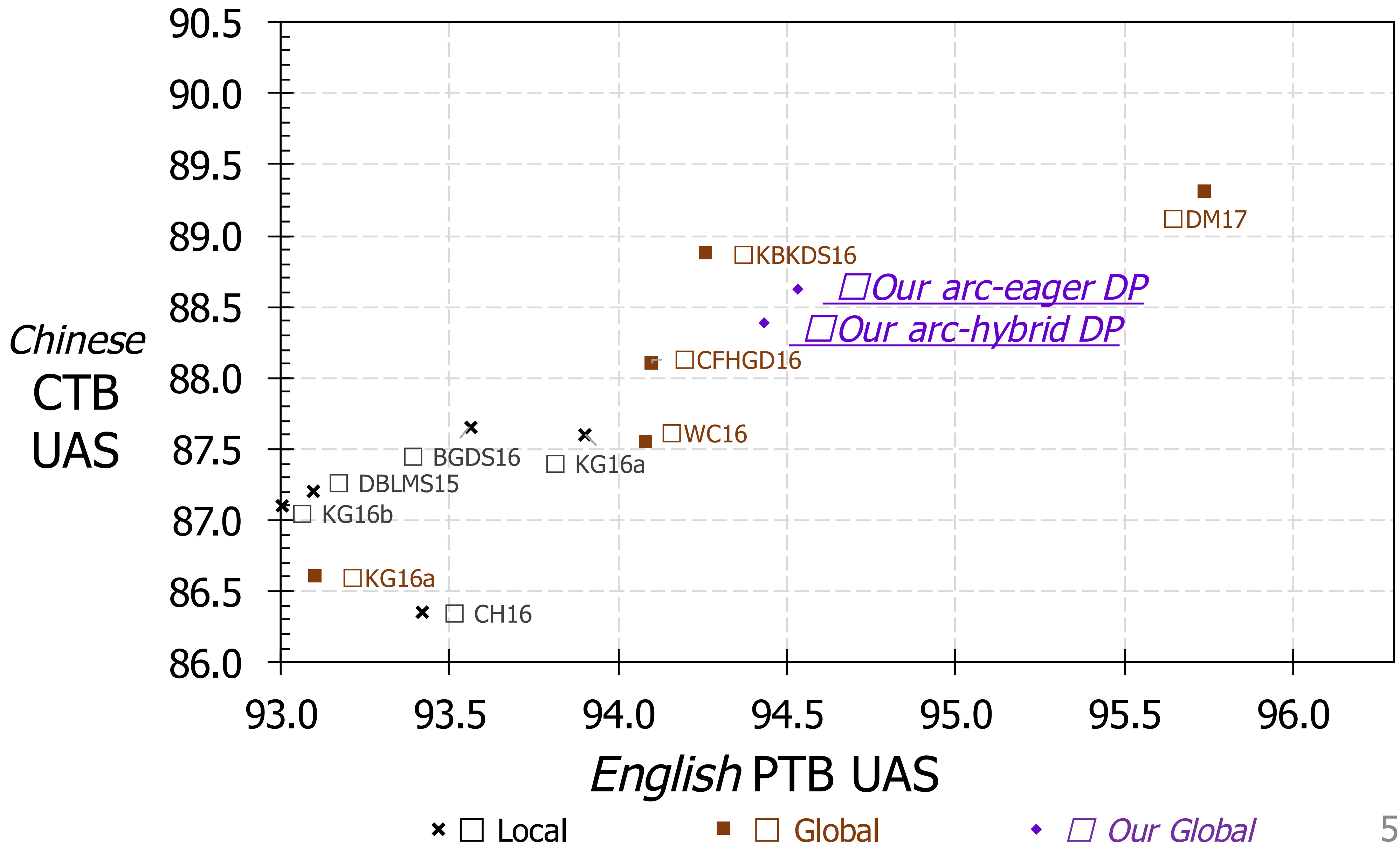
reduce$\curvearrowleft$ $$\frac{[i,k]: v_1 \qquad [k,j]: v_2}{[i,j]: v_1 + v_2 + s_{sh}(i,k) + s_{re\curvearrowleft}(k,j) + \mathbf{1}(head(k) \neq j)}$$



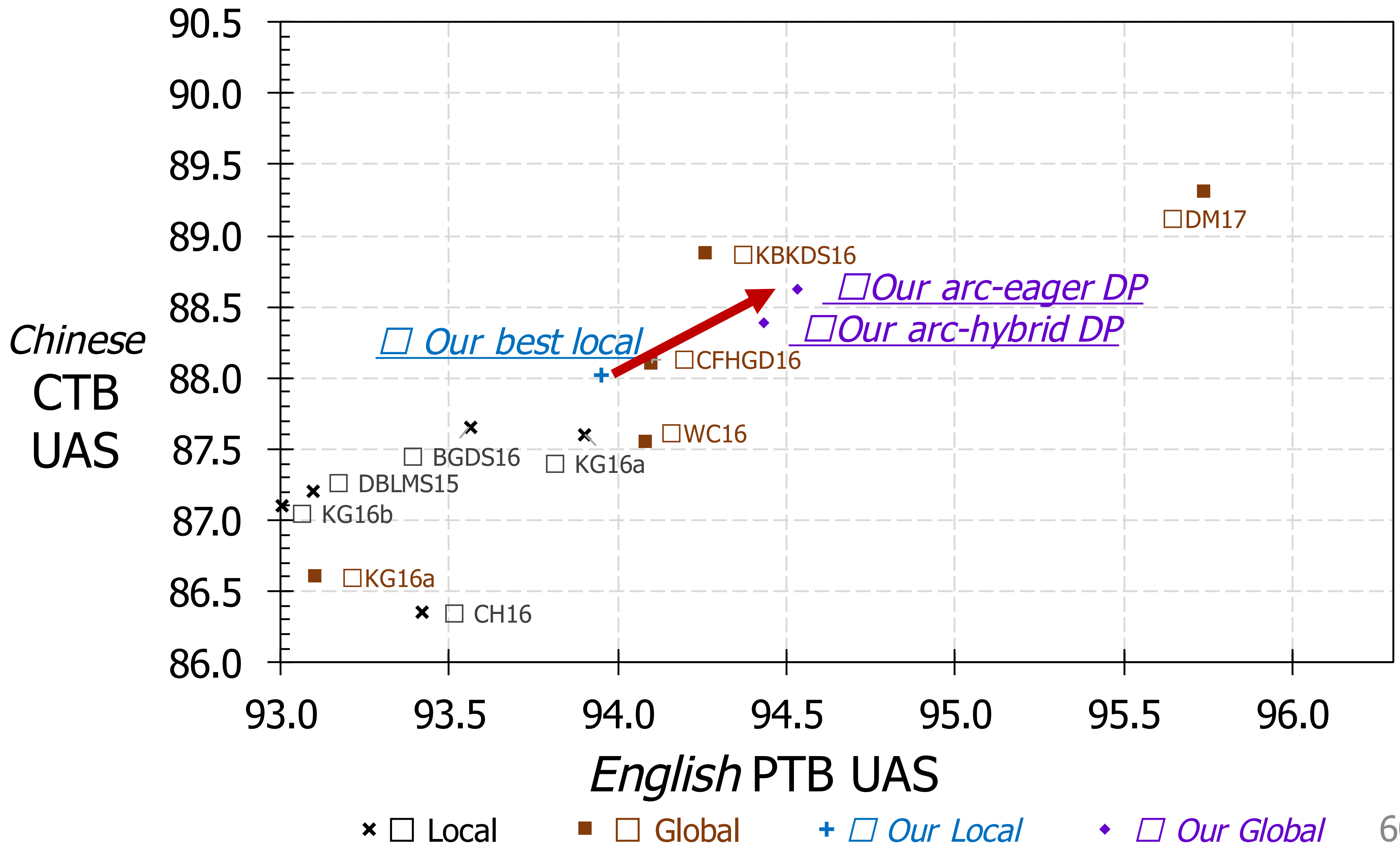
Comparing with State-of-the-art



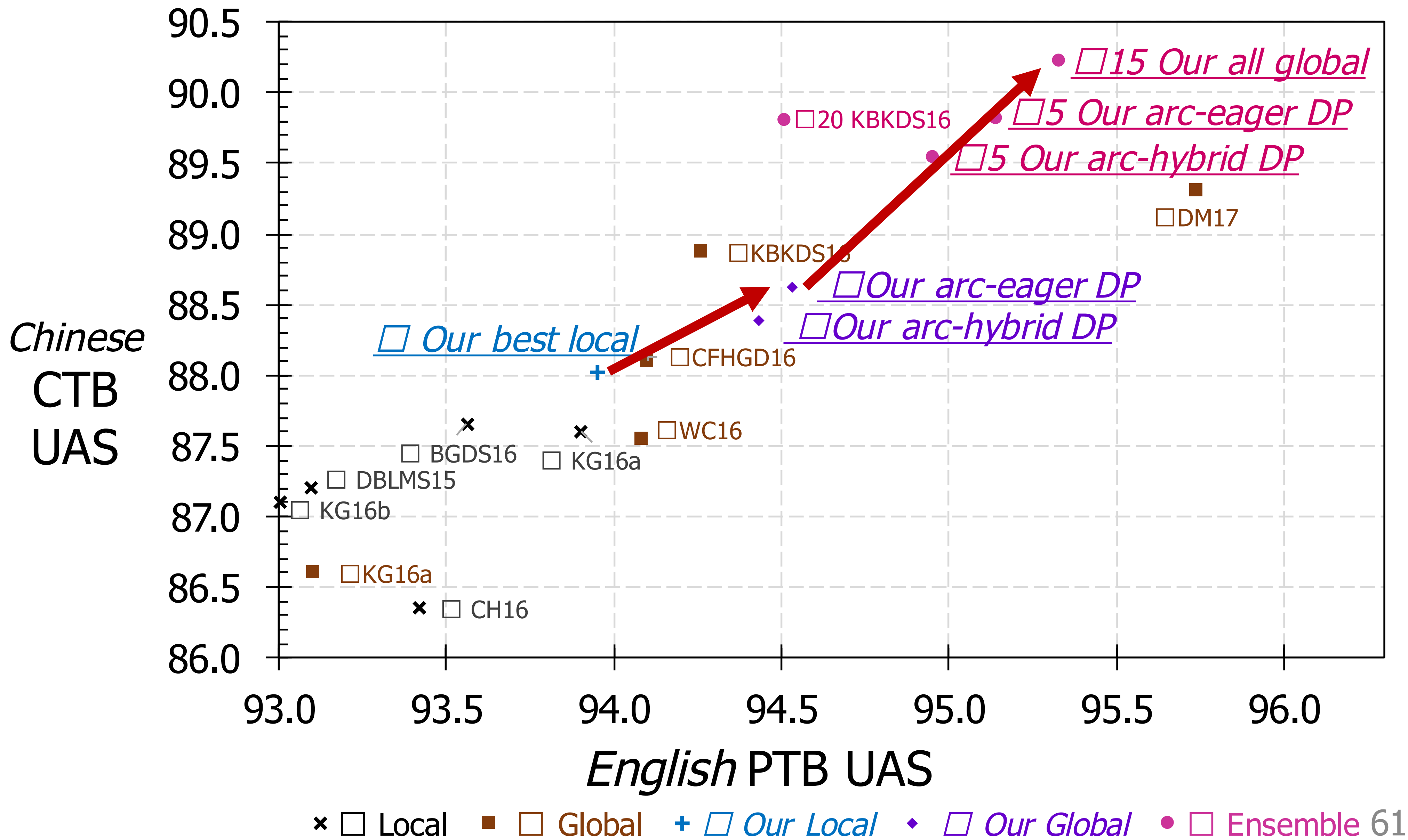
Comparing with State-of-the-art



Comparing with State-of-the-art

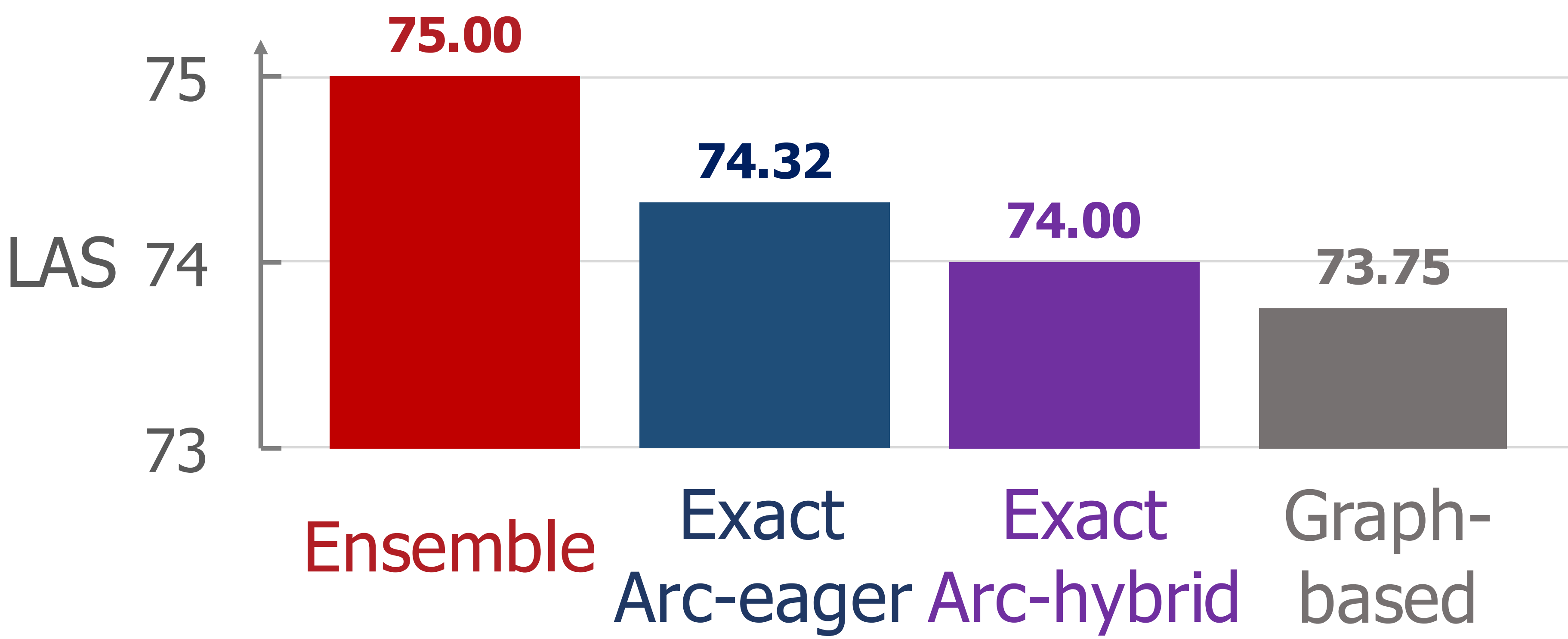


Comparing with State-of-the-art



Results – CoNLL'17 Shared Task

- Macro-average of 81 treebanks in 49 languages
- 2nd–highest overall performance



(Shi, Wu, Chen and Cheng, 2017; Zeman et al., 2017)

Conclusion

- Bi-LSTM feature set is minimal yet highly effective
- First $O(n^3)$ implementation of exact decoders
- Global training and decoding gave high performance

More in Our Paper

- Description and analysis of three transition systems (arc-standard, arc-hybrid, arc-eager)
- CKY-style representations of the deduction systems
- Theoretical analysis of the global methods
  - Arc-eager models can “simulate” arc-hybrid models
  - Arc-eager models can “simulate” edge-factored models

Fast(er) Exact Decoding and Global Training for Transition-Based Dependency Parsing via a Minimal Feature Set

https://github.com/tzshi/dp-parser-emnlp17

Tianze Shi*   Liang Huang†   Lillian Lee*

* Cornell University
† Oregon State University

CKY-style Visualization

Axiom $[0, 1]$

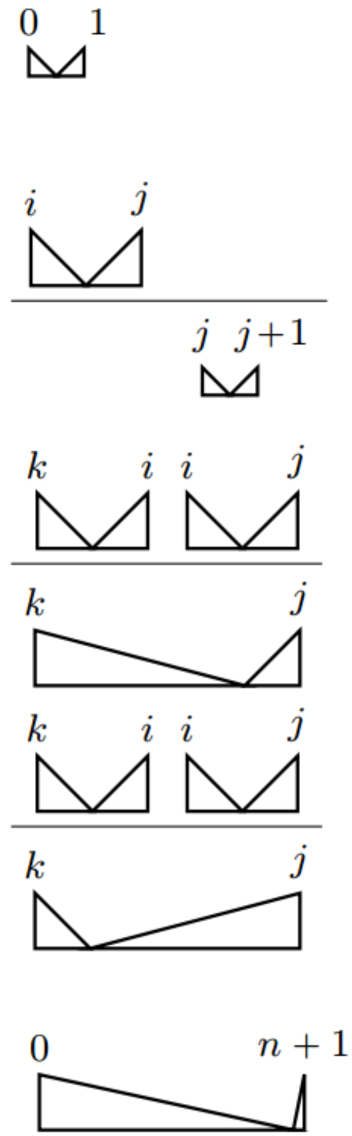
Inference Rules

sh $\dfrac{[i, j]}{[j, j+1]}$ $\quad j \leqslant n$

re$_\curvearrowright$ $\dfrac{[k, i] \quad [i, j]}{[k, j]}$ $\quad k \curvearrowright i$

re$_\curvearrowleft$ $\dfrac{[k, i] \quad [i, j]}{[k, j]}$ $\quad i \curvearrowleft j$

Goal $[0, n+1]$



(b) Arc-hybrid