Dynamic oracles

Marco Kuhlmann

Department of Computer and Information Science

LINKÖPING UNIVERSITY

Oracles in transition-based dependency parsing

- Training a transition-based parser requires an **oracle** that translates gold-standard trees into sequences of transitions.

- The classical way to conceptualise this oracle is as a deterministic function – a **static oracle**.

  transition sequences can be computed off-line

- Under this view, the transition sequences are used to train the parser using teacher forcing.

  next transition = oracle transition

Static oracle

- Choose LA if this would create an arc from the gold-standard tree, and if all arcs from the second-topmost word on the stack have already been assigned by the parser.
- Choose RA if this would create an arc from the gold-standard tree, and if all arcs from the topmost word on the stack have already been assigned by the parser.
- Otherwise, choose SH.

  must always be valid, unless the tree is non-projective

Two problems with the static oracle

- **Problem 1:** For each configuration, the static oracle yields only one transition. In general however, several different transitions can be used to recover the arcs in the gold-standard tree.

- **Problem 2:** The transitions yielded by the static oracle are all taken in optimal configurations. The parser never learns how to make transitions out of sub-optimal configurations.

  exposure bias

Training with dynamic oracles

- We consider **dynamic oracles** that can propose ‘good’ transitions out of even potentially sub-optimal configurations.

- To take the step to the next configuration, we use the transition predicted by the classifier.

  may lead the parser into sub-optimal configurations

- To update the parameters of the classifier, we use the highest-scoring ‘good’ transition.

  trains the parser to deal with sub-optimal configurations

Dynamic oracles, formal definition

- We introduce a **cost function** $C(A, T)$, which measures the cost of outputting a parse tree A when the gold-standard tree is T.
- We define this cost as the Hamming loss between the arc sets of the two trees – the cardinality of their symmetric difference:

$$C(A, T) = |\underbrace{(A \setminus T)}_{\text{gold-standard arcs not predicted}} \cup \underbrace{(T \setminus A)}_{\text{predicted arcs not in the gold standard}}|$$

Goldberg and Nivre (2013)

Dynamic oracles, formal definition

- The cost of a transition t in a configuration c is the difference in cost between the best tree reachable from $t(c)$ and c, respectively.

$$C(t\,; c, T) \;=\; \underbrace{\min_{A : t(c) \rightsquigarrow A}}_{\text{tree reachable from t(c)}} C(A, T) - \underbrace{\min_{A : c \rightsquigarrow A}}_{\text{tree reachable from c}} C(A, T)$$

- A **dynamic oracle** is a non-deterministic function that returns the set of transitions with zero cost.

Goldberg and Nivre (2013)

Computational problems

- Computing the cost of a transition boils down to a minimisation problem over reachable dependency trees.

- For the arc-standard algorithm, solving this problem requires the use of a polynomial-time dynamic programming algorithm.

  Goldberg et al. (2014)

- For the slightly different **arc-hybrid algorithm**, the cost of each transition can be computed more efficiently.

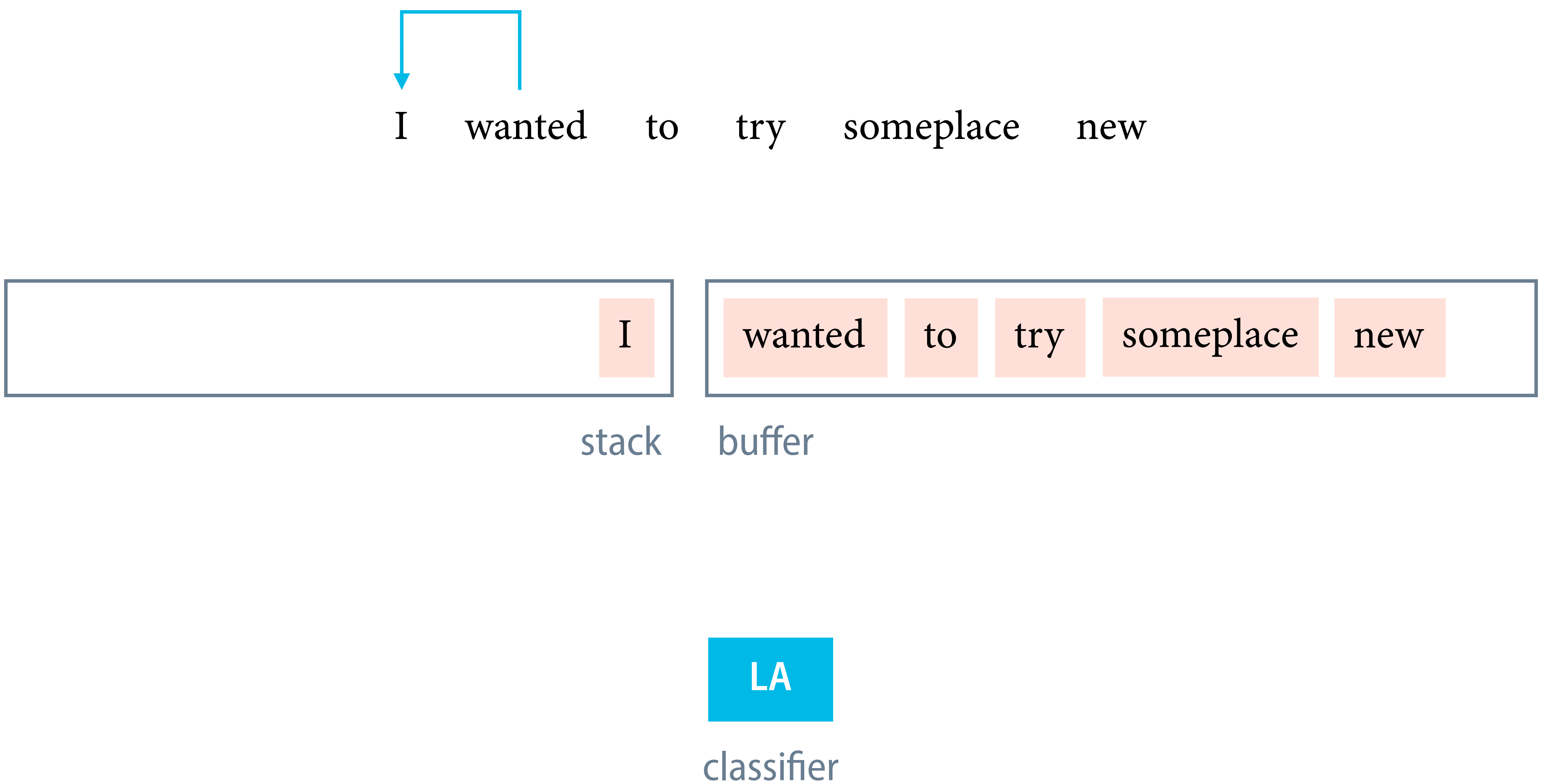
  Goldberg and Nivre (2013): arc decomposability

The arc-hybrid algorithm

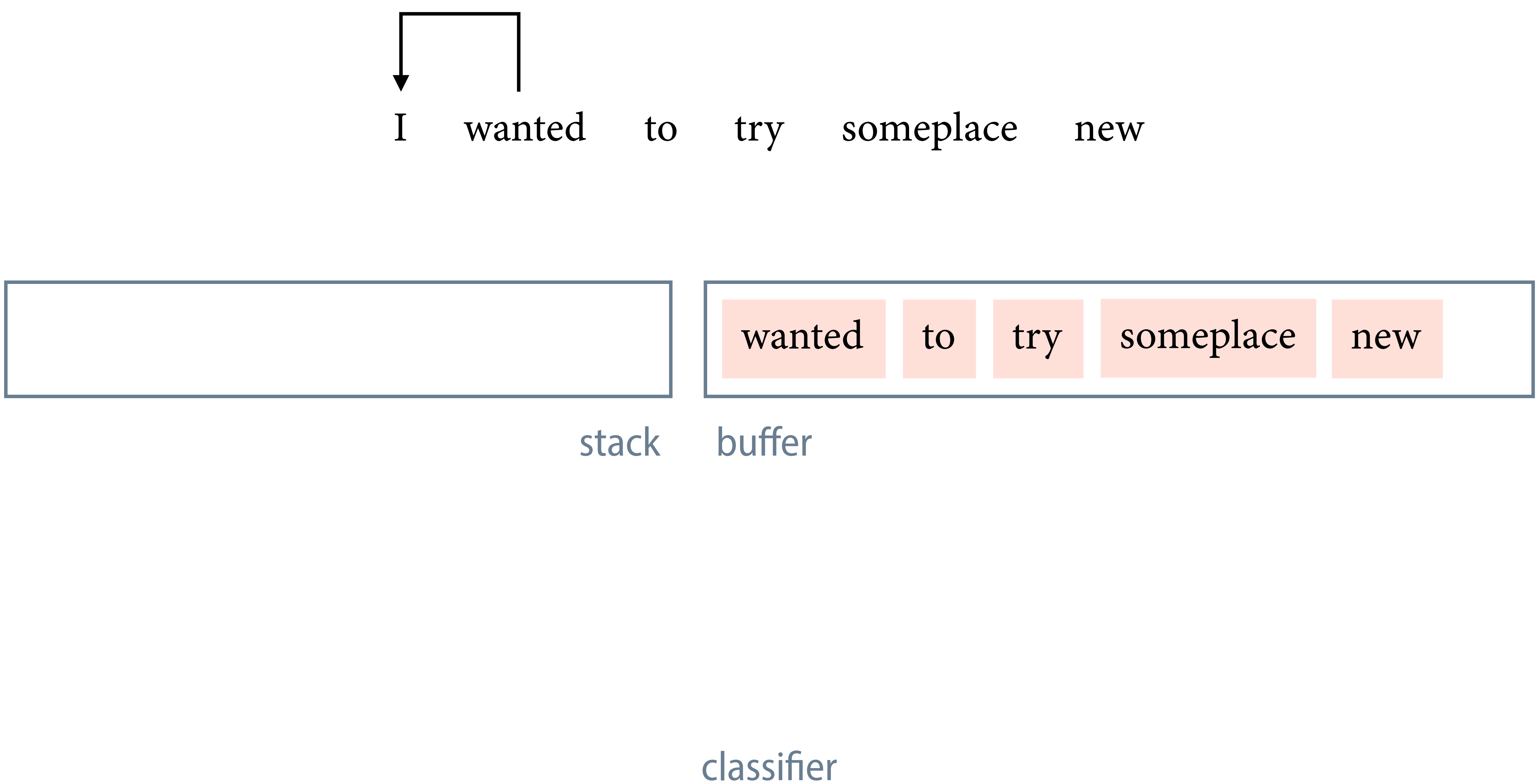
- The **shift transition (SH)** removes the frontmost word from the buffer and pushes it to the top of the stack.
- The **left-arc transition (LA)** creates a dependency from the frontmost word in the buffer to the topmost word on the stack, and pops the topmost word on the stack.
- The **right-arc transition (RA)** creates a dependency from the second-topmost word on the stack to the topmost word, and pops the topmost word.

Kuhlmann et al. (2011)

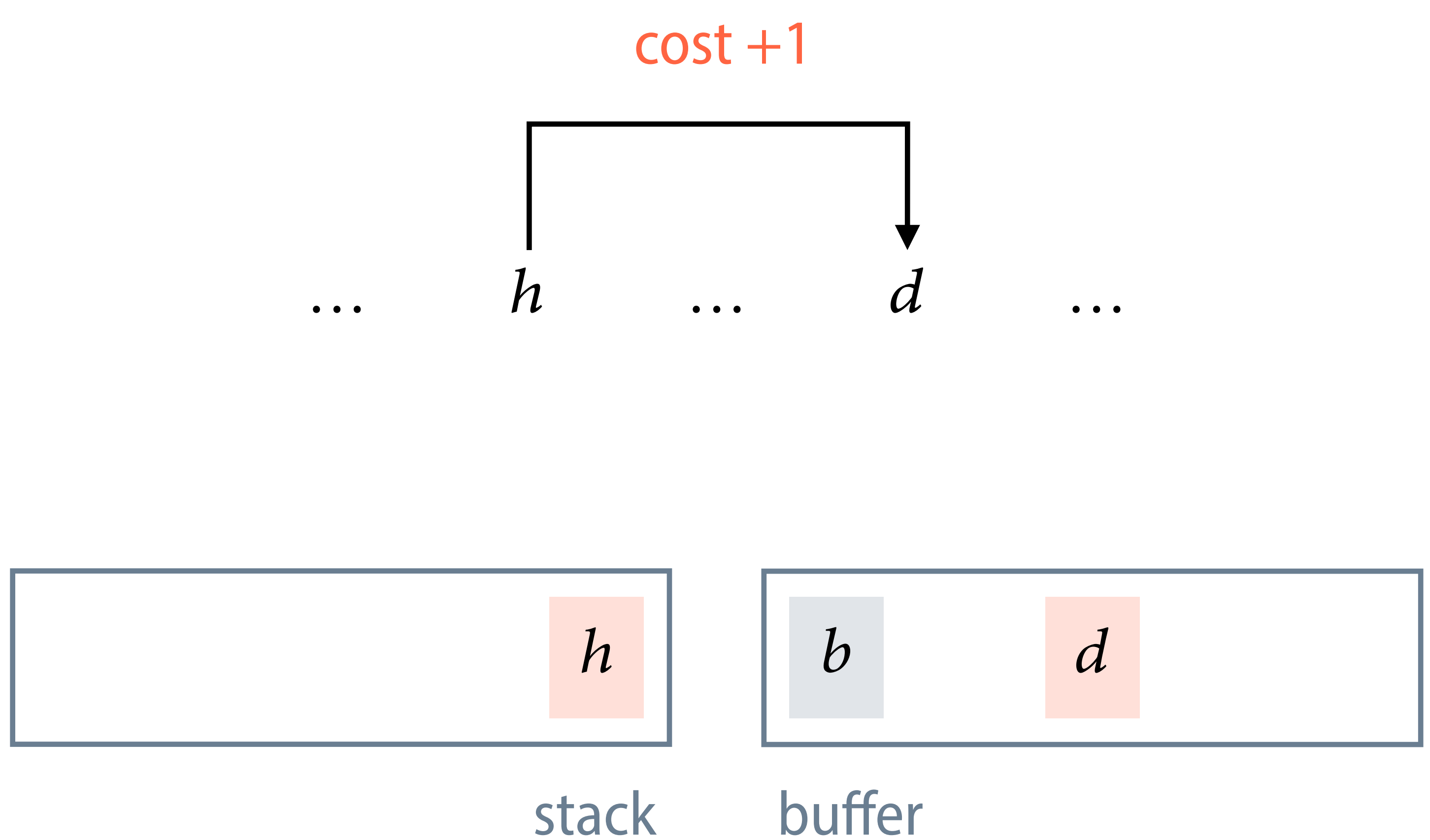
Left-arc in the arc-hybrid algorithm



Left-arc in the arc-hybrid algorithm



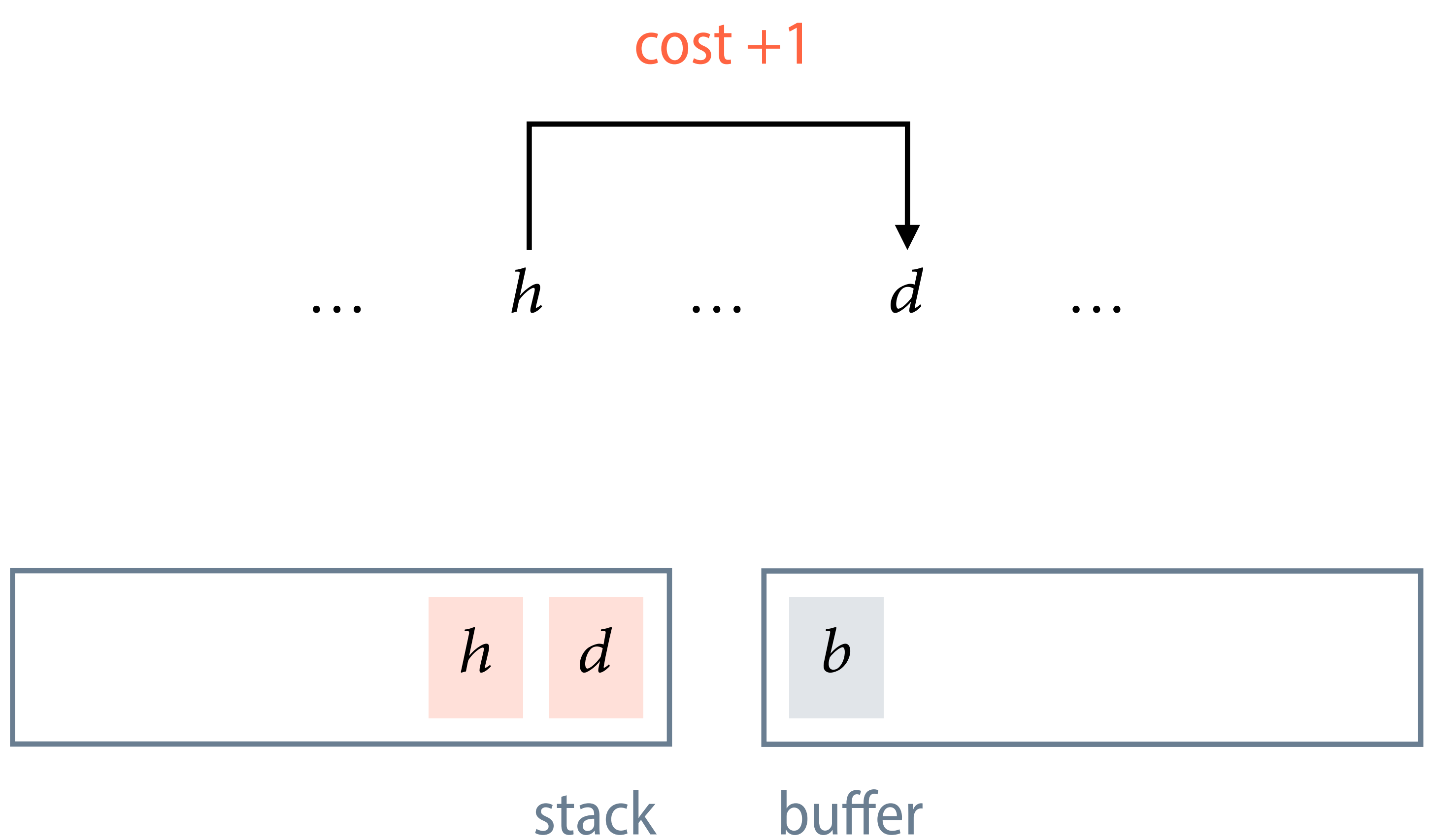
The cost of the left-arc transition (1)



Making a LA transition will pop *h* from the stack, so that it can no longer find its gold-standard dependent *d*.

Goldberg and Nivre (2013)

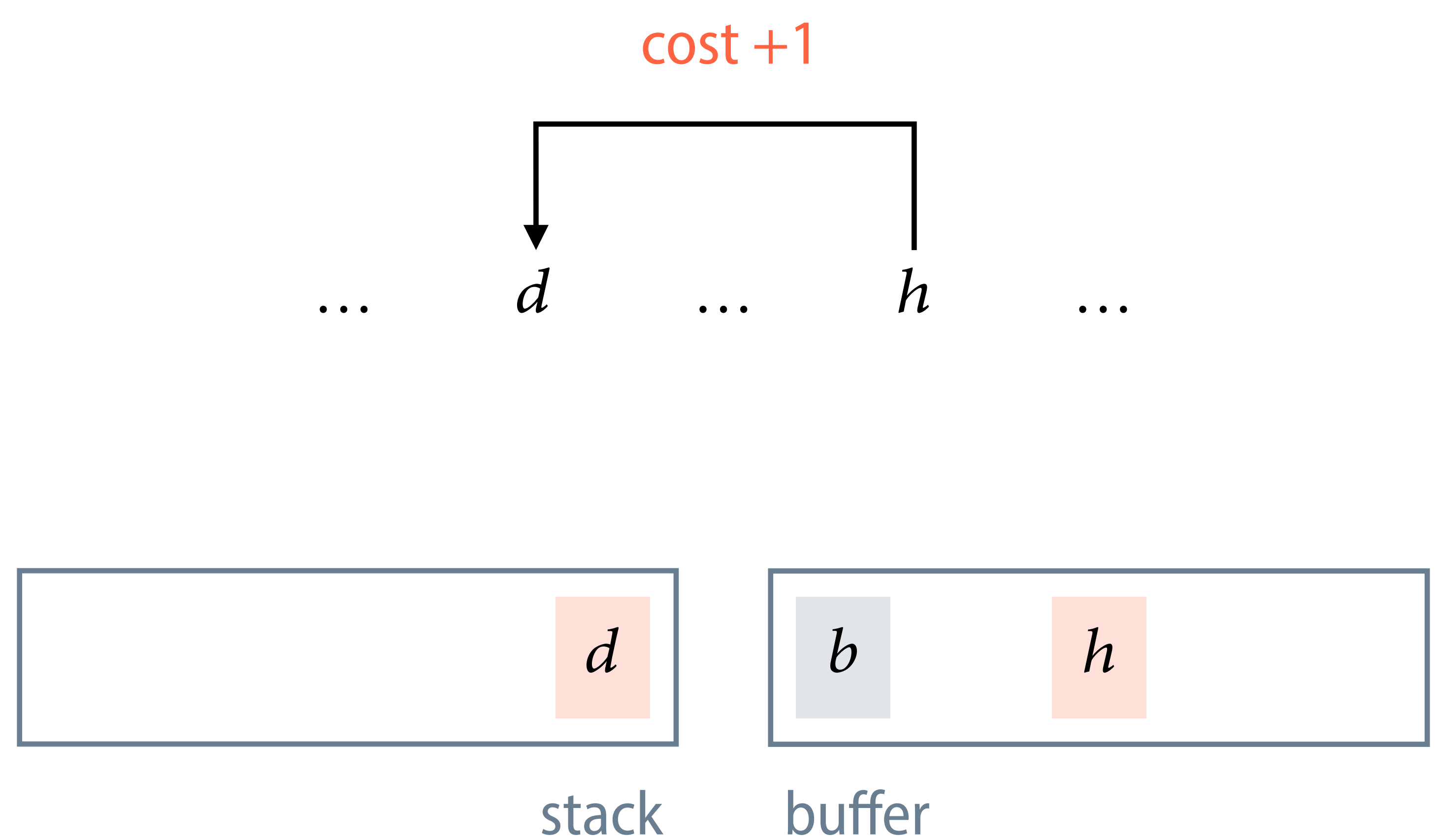
The cost of the left-arc transition (2)



Making a LA transition will pop d from the stack, so that it can no longer find its gold-standard head h.

Goldberg and Nivre (2013)

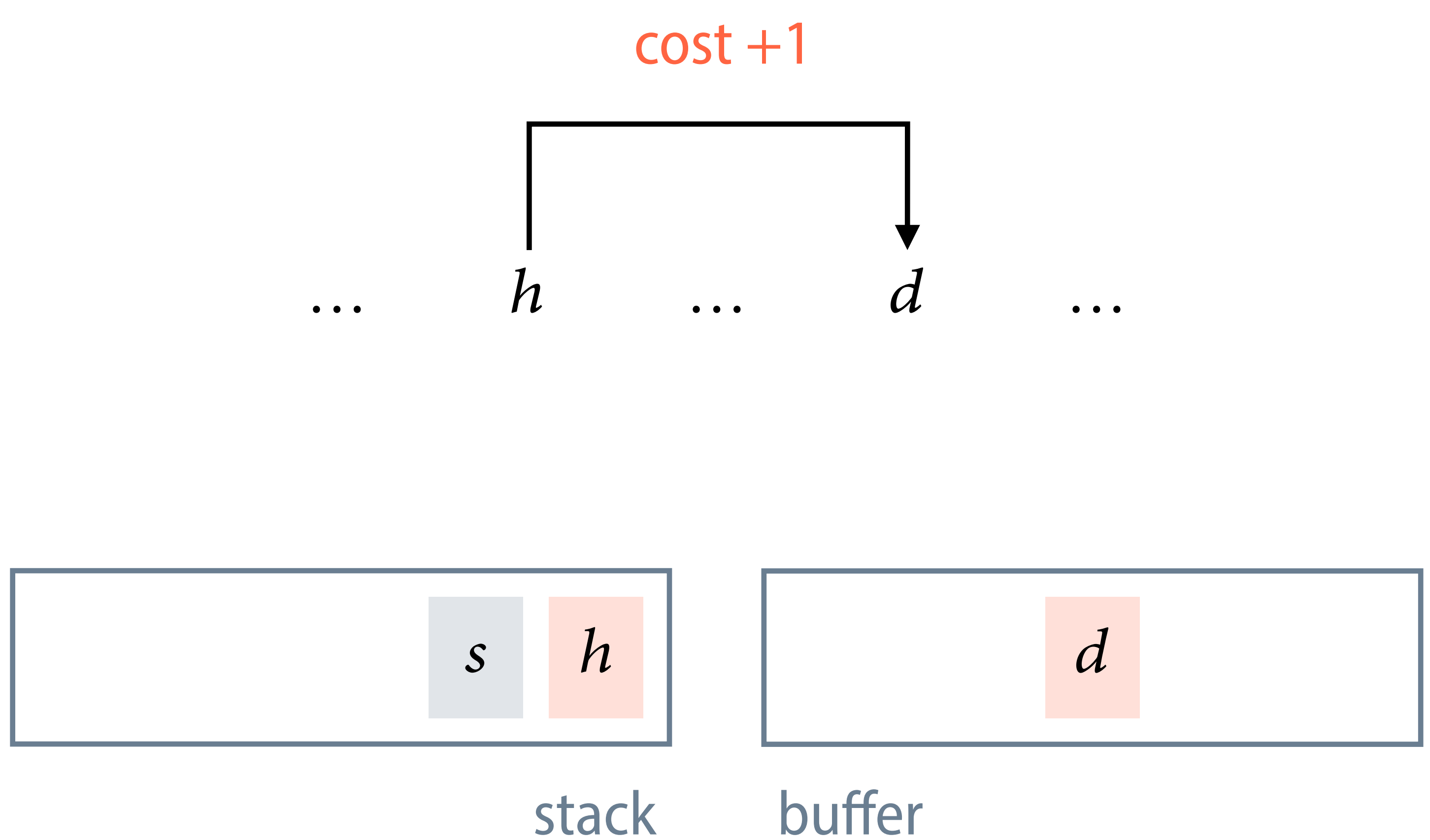
The cost of the left-arc transition (3)



Making a LA transition will pop d from the stack, so that it can no longer find its gold-standard head h.

Goldberg and Nivre (2013)

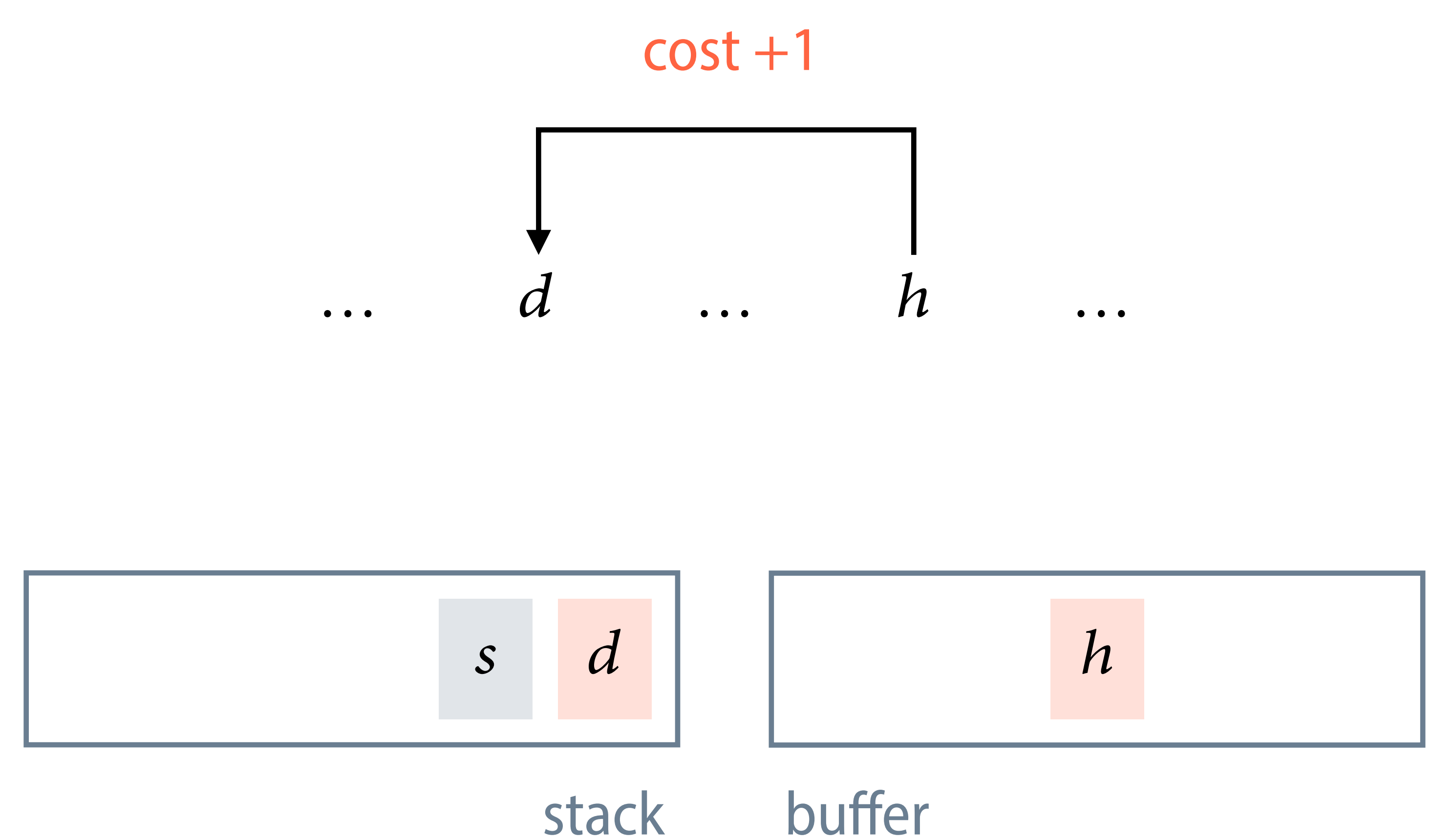
The cost of the right-arc transition (1)



Making a RA transition will pop h from the stack, so that it can no find its gold-standard dependent d.

Goldberg and Nivre (2013)

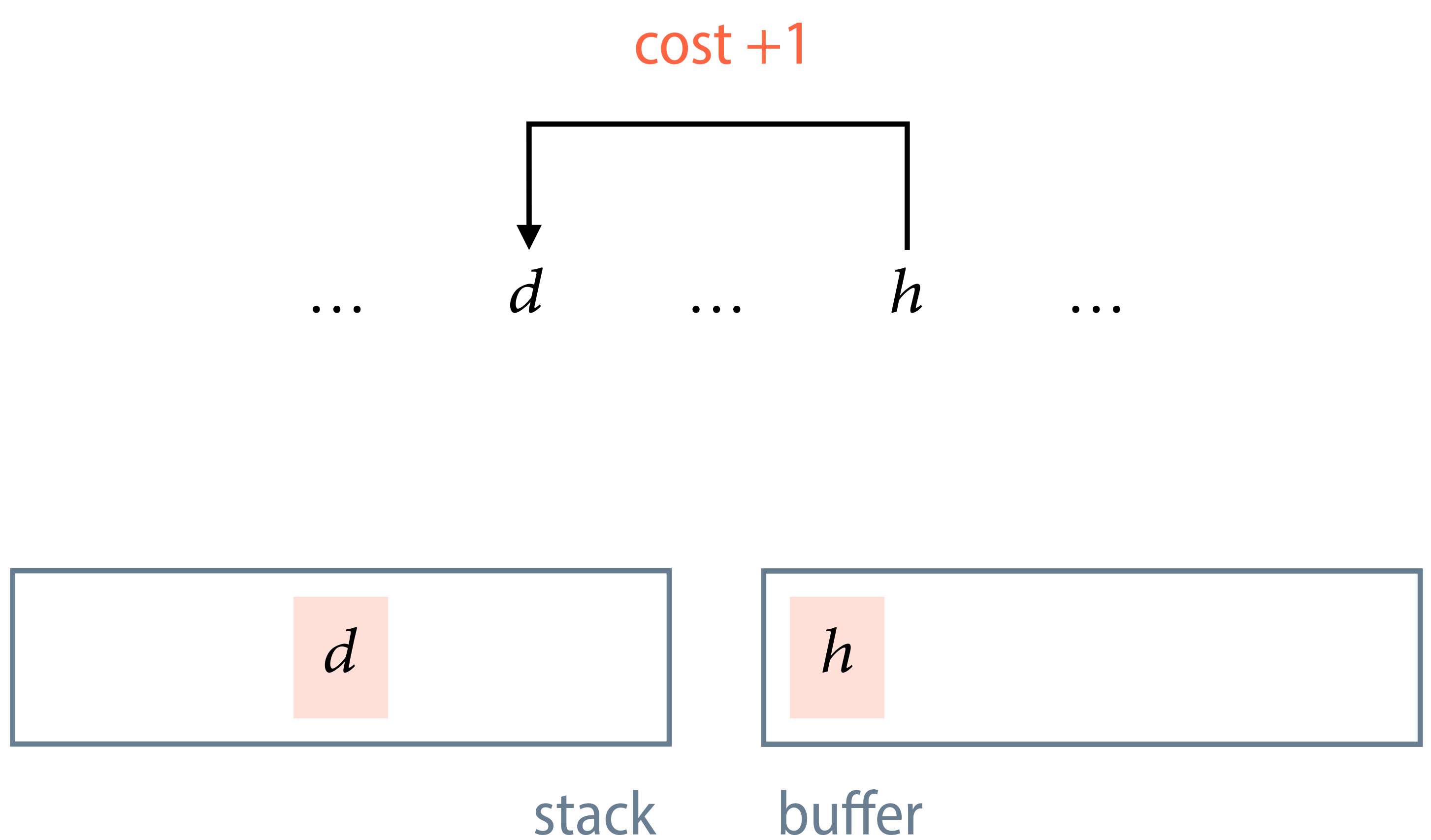
The cost of the right-arc transition (2)



Making a RA transition will pop d from the stack, so that it can no longer find its gold-standard head h.

Goldberg and Nivre (2013)

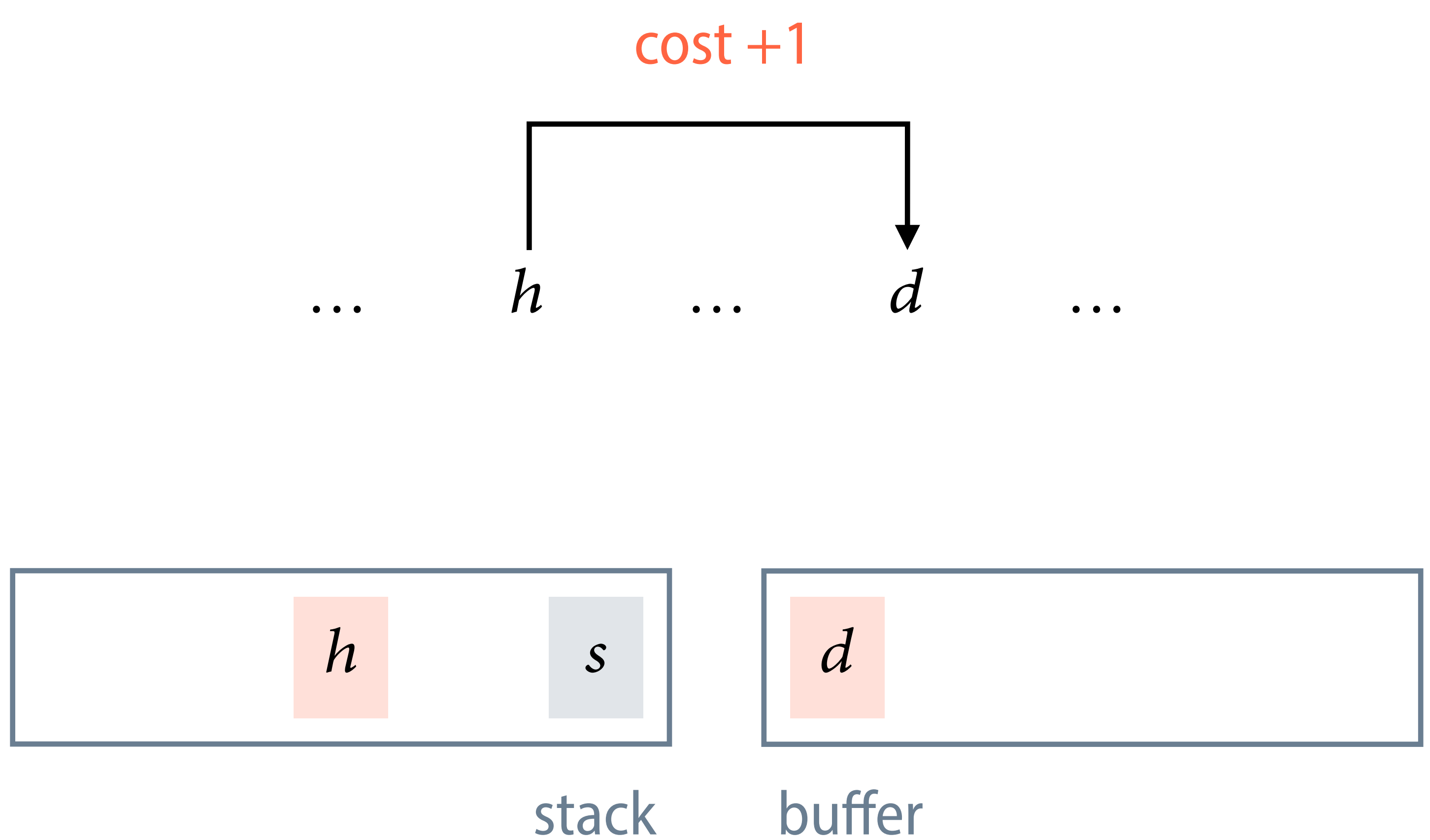
The cost of the shift transition (1)



Making a SH transition would move h to the stack, so that it can longer find its gold-standard dependent d.

Goldberg and Nivre (2013)

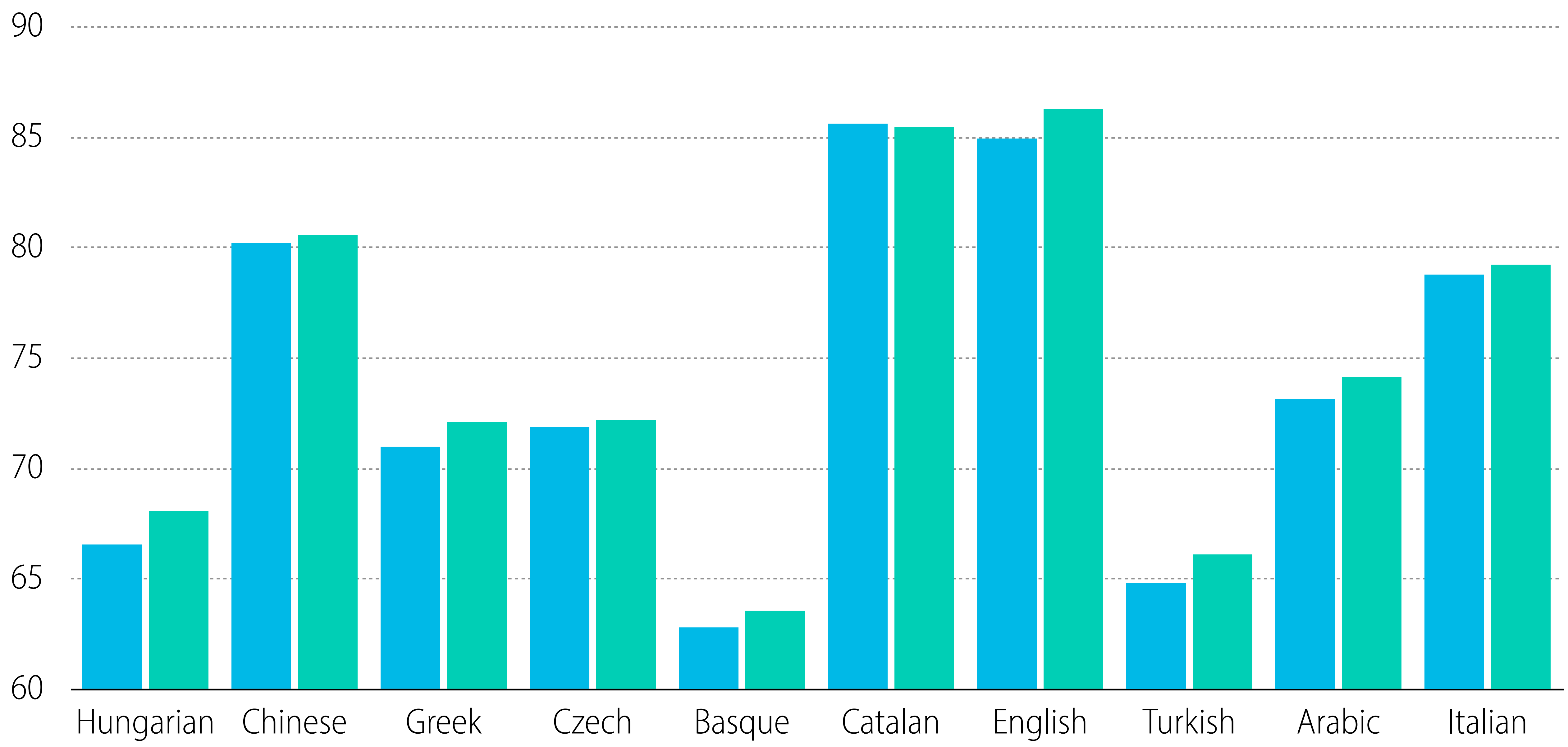
The cost of the shift transition (2)



Making a SH transition would move *d* to the stack, so that it can longer find its gold-standard head *h*.

Goldberg and Nivre (2013)

Empirical results



Labelled Attachment Score on the CoNLL 2007 data set – Goldberg and Nivre (2013)

Final comments

- We do not need to know the exact cost of a transition; we only need to know whether that cost is zero or non-zero.

- The cost computation only happens at training time. The fast runtime at test time is maintained.

- The use of dynamic oracles can be seen as a problem-specific instance of **imitation learning**.

  learn to imitate the desired behaviour; alternative to reinforcement learning