Neural Greedy Constituent Parsing with Dynamic Oracles

Maximin Coavoux
Benoît Crabbé

Université Paris Diderot – ALPAGE / Inria

ACL 2016 – Berlin

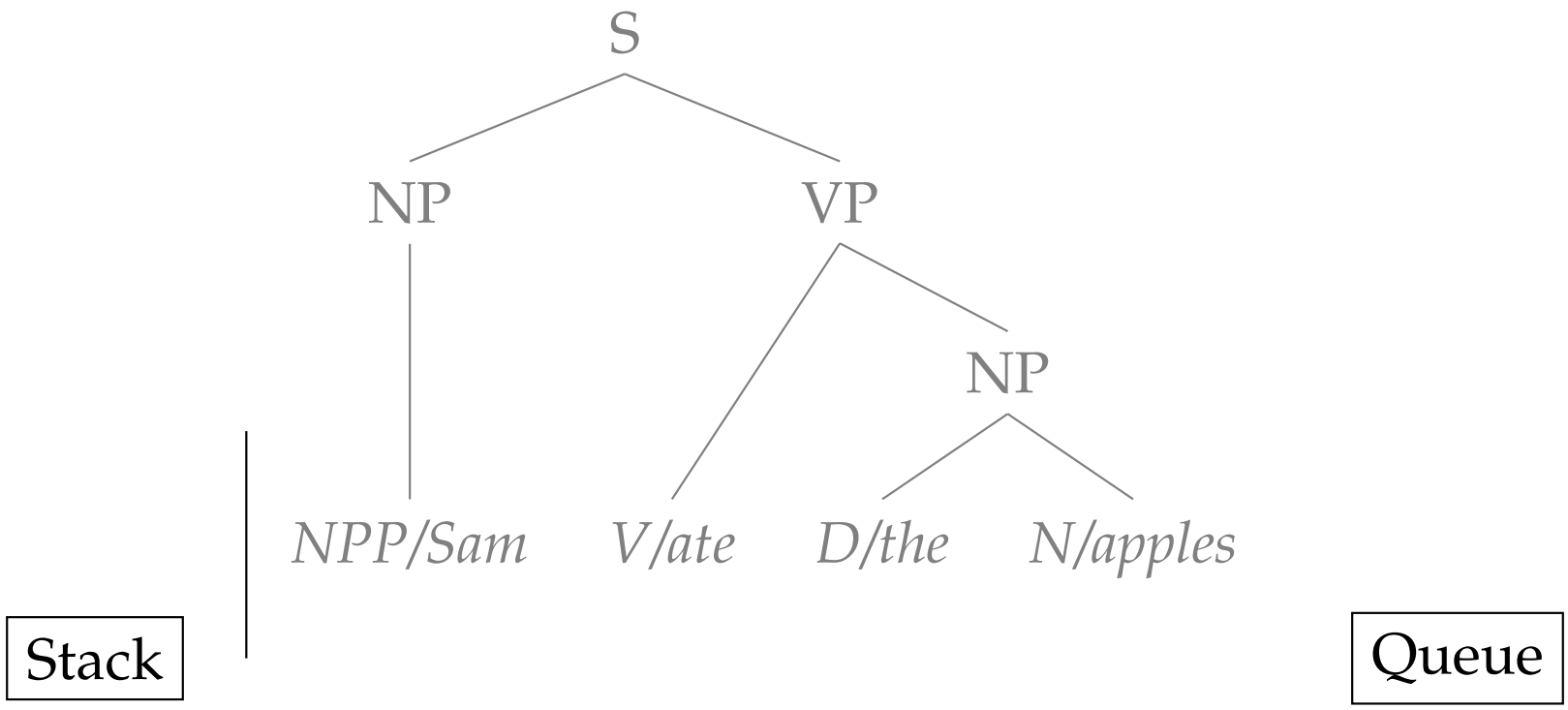
Context and Contribution

- **Greedy constituent parsing**
  - Morphologically rich languages
  - Neural parsing (feedforward)

- **Dynamic oracle** for transition-based constituent parsing
  - Result : improvements on almost all languages of the SPMRL datasets (Arabic, Basque, French, German, Hebrew, Hungarian, Korean, Polish, Swedish) + English (PTB).

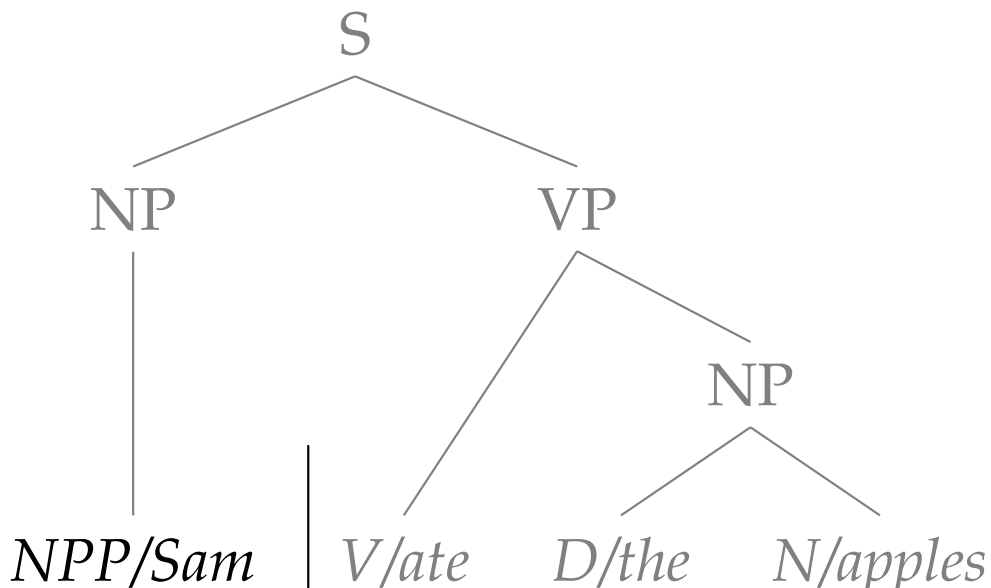
Plan

SHIFT-REDUCE



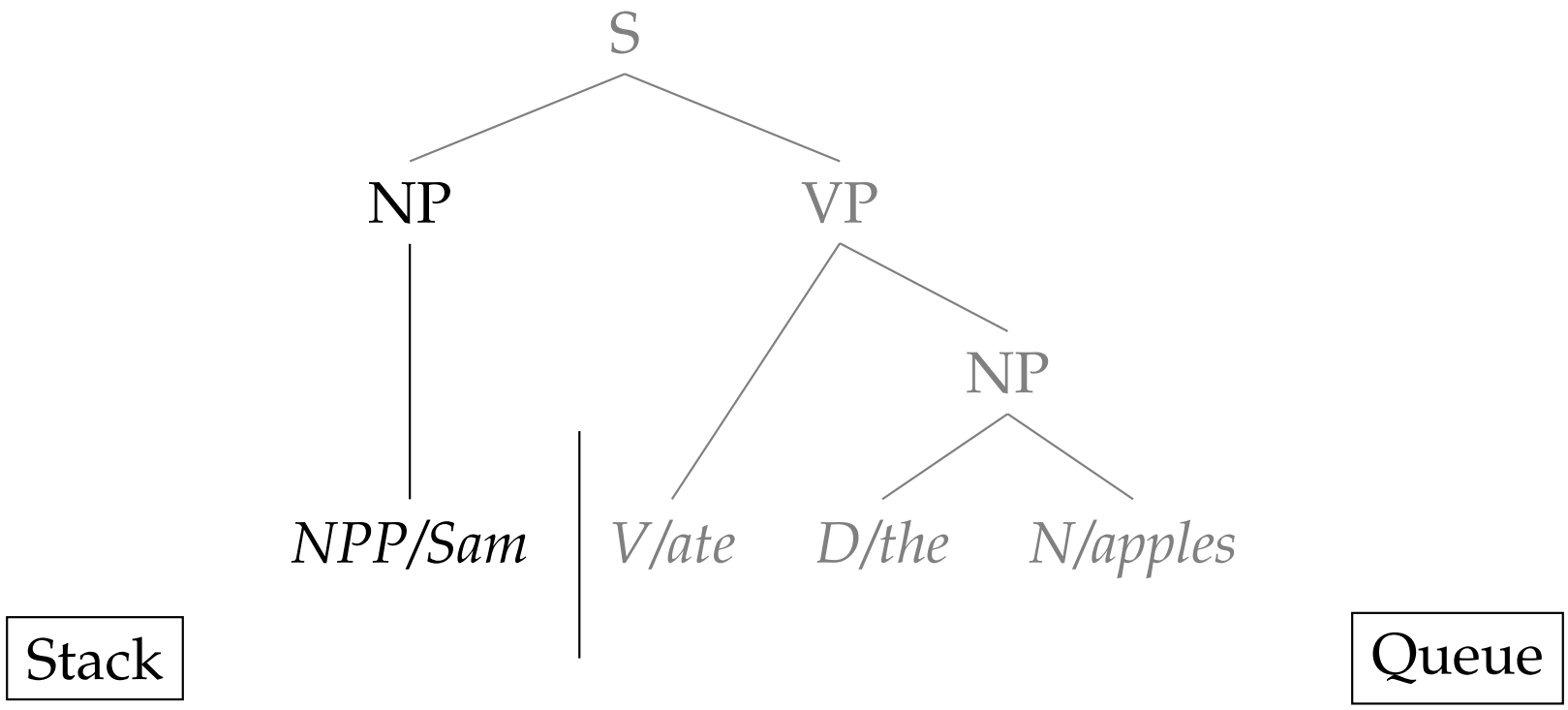
Derivation :
Predicted set : {}

SHIFT-REDUCE



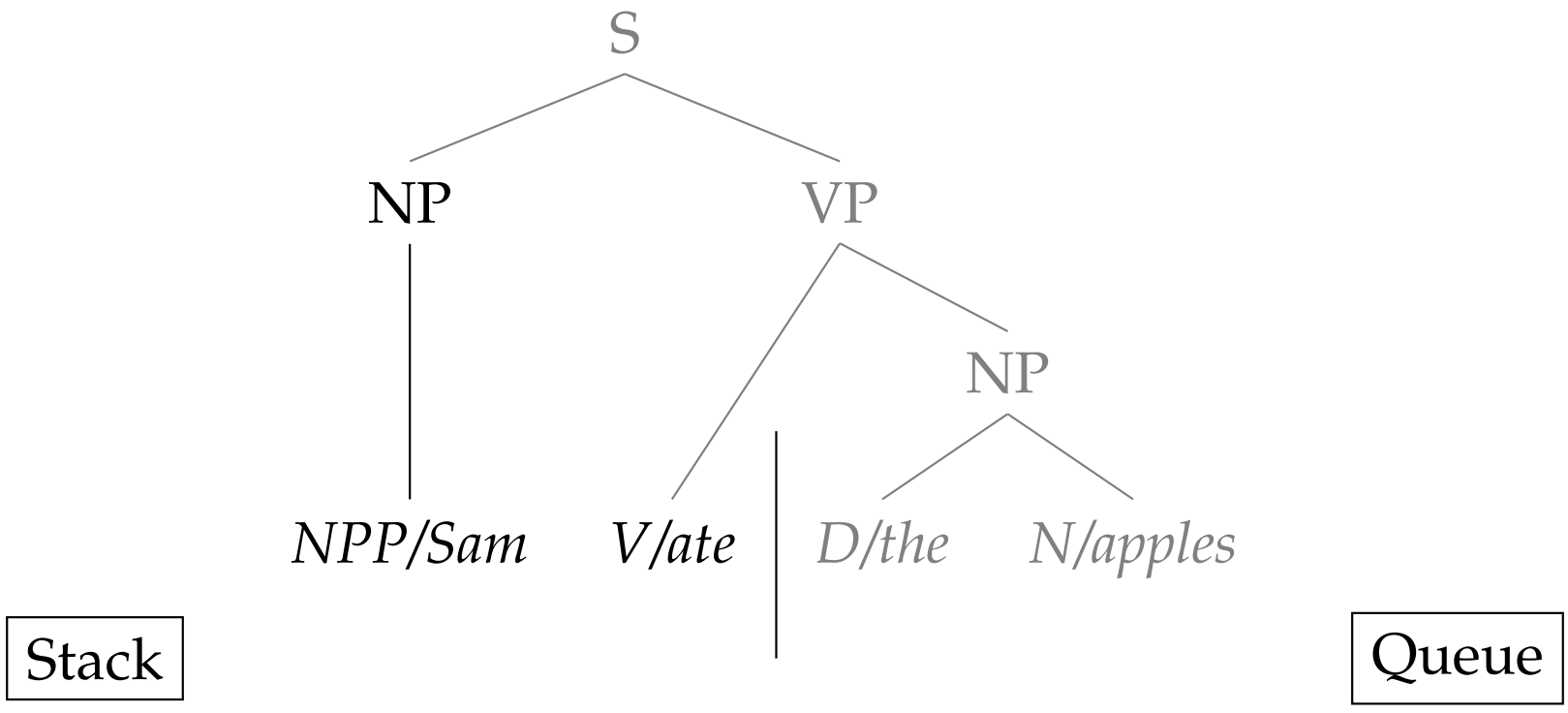
Derivation : SHIFT
Predicted set : {}

SHIFT-REDUCE



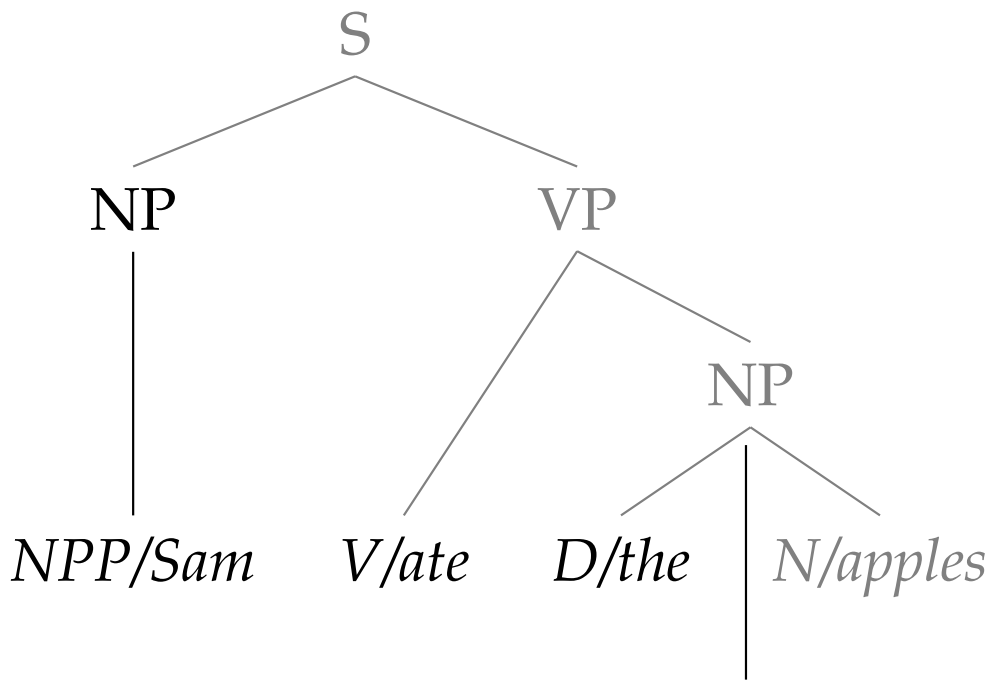
Derivation : SHIFT, RU-NP
Predicted set : $\{(NP, 0, 1)\}$

SHIFT-REDUCE



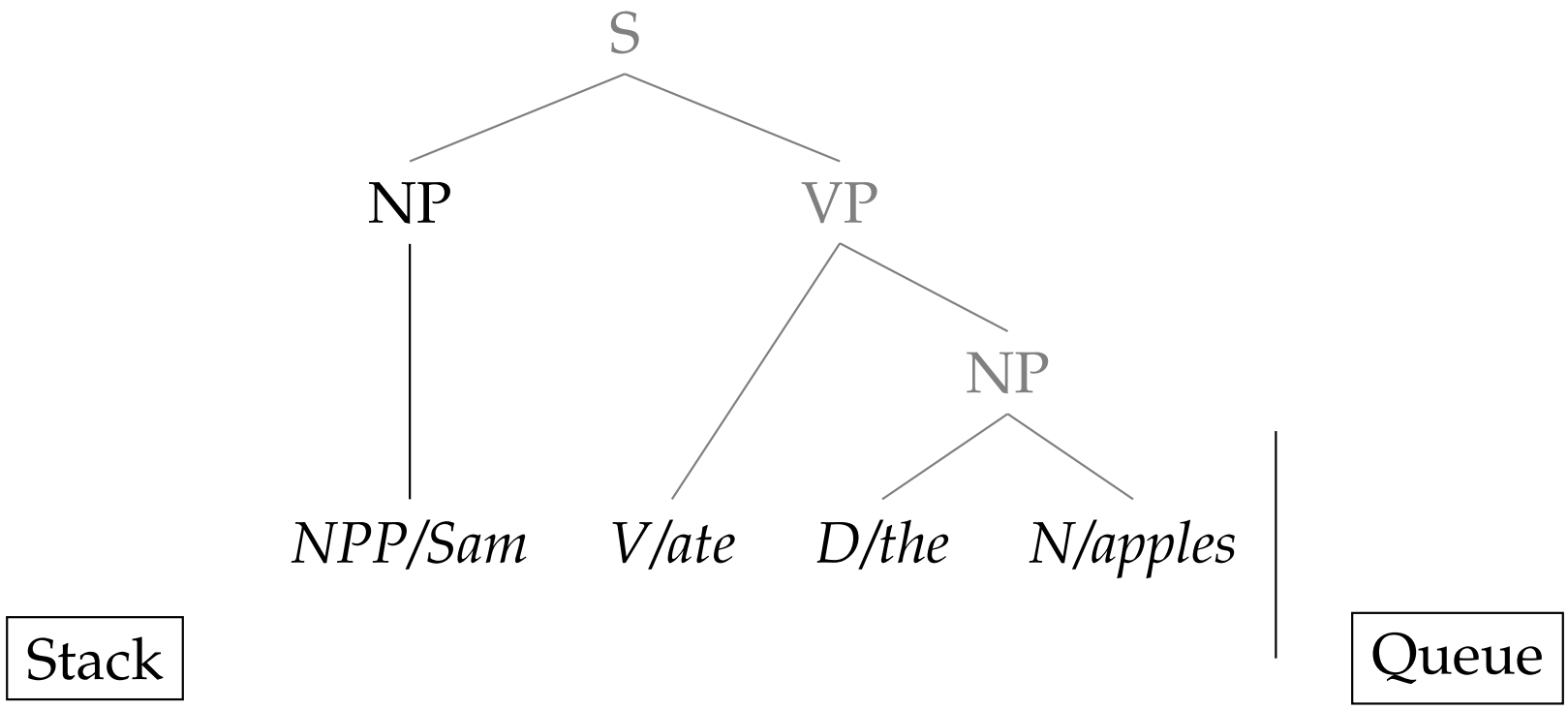
Derivation : SHIFT, RU-NP, SHIFT
Predicted set : $\{(NP, 0, 1)\}$

SHIFT-REDUCE



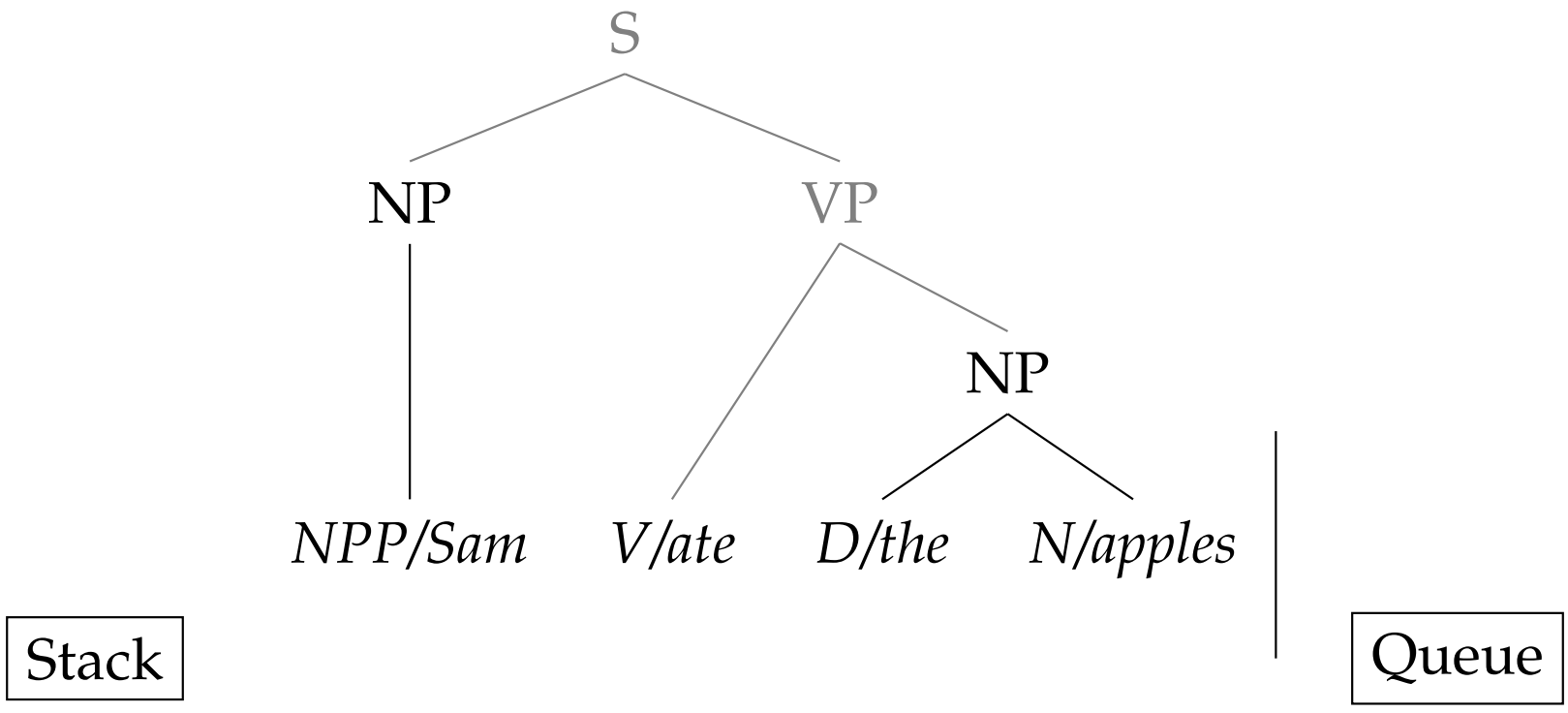
Derivation : SHIFT, RU-NP, SHIFT, SHIFT
Predicted set : $\{(NP, 0, 1)\}$

SHIFT-REDUCE



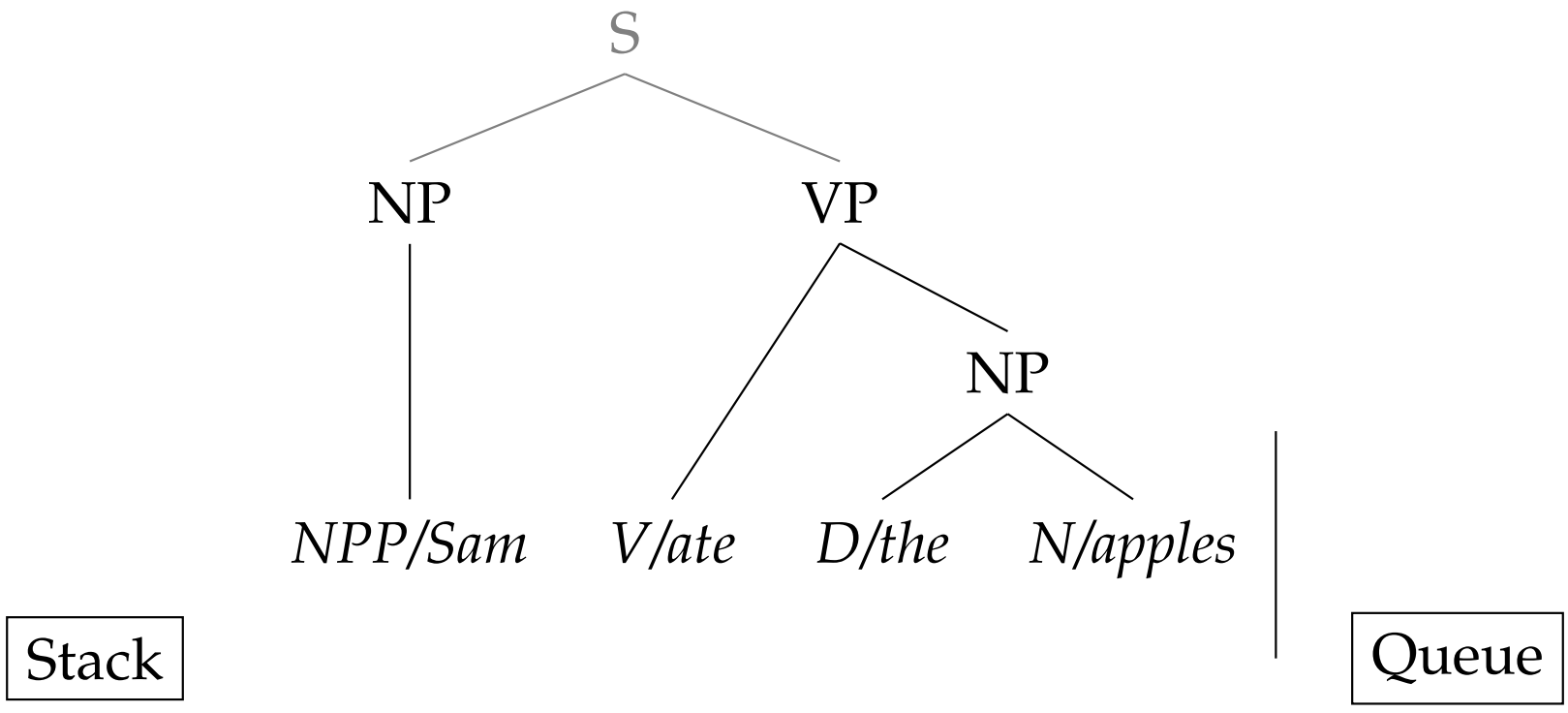
Derivation : SHIFT, RU-NP, SHIFT, SHIFT, SHIFT
Predicted set : $\{(NP, 0, 1)\}$

SHIFT-REDUCE



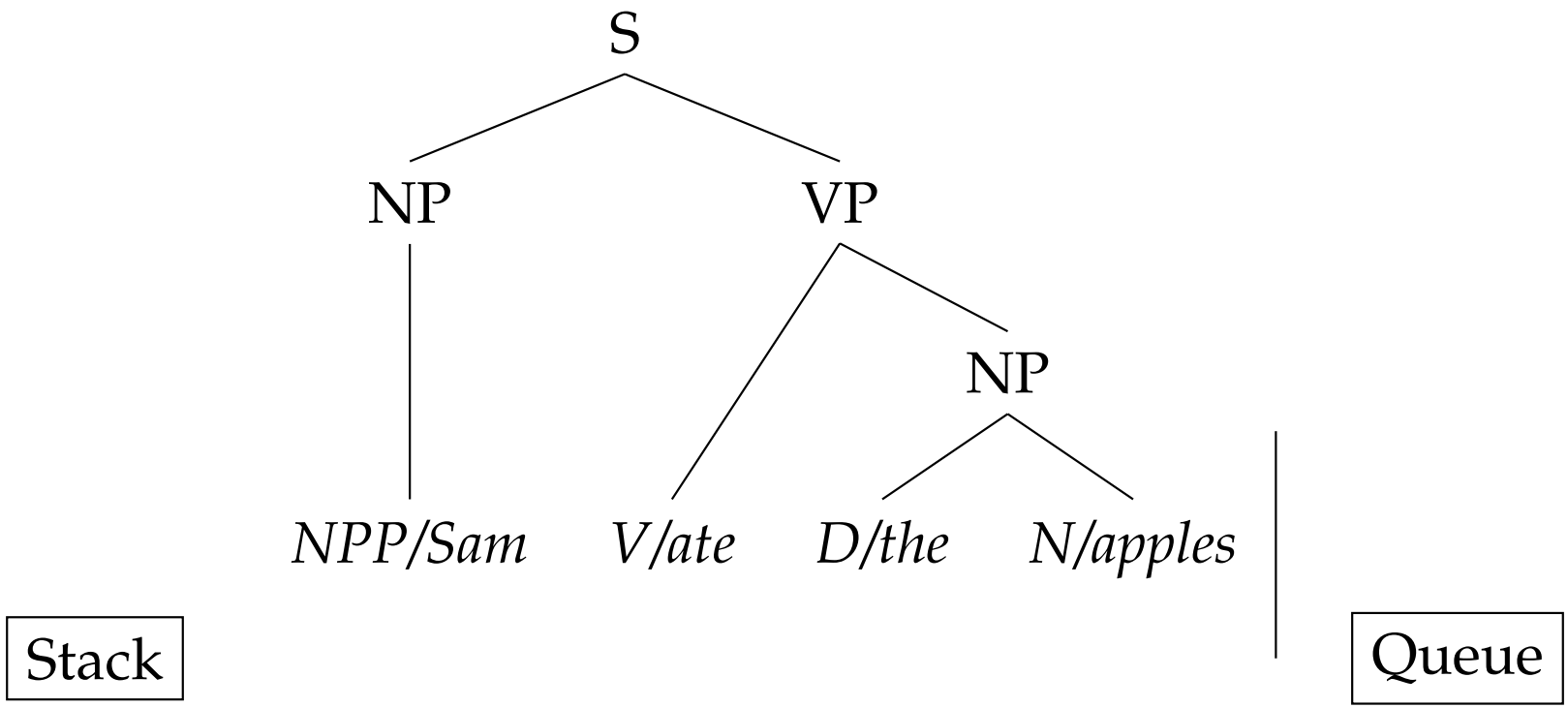
Derivation : SHIFT, RU-NP, SHIFT, SHIFT, SHIFT, R-NP
Predicted set : $\{(NP, 0, 1), (NP, 2, 4)\}$

SHIFT-REDUCE



Derivation : SHIFT, RU-NP, SHIFT, SHIFT, SHIFT, R-NP, R-VP
Predicted set : $\{(NP, 0, 1), (NP, 2, 4), (VP, 1, 4)\}$

SHIFT-REDUCE



Derivation : SHIFT, RU-NP, SHIFT, SHIFT, SHIFT, R-NP, R-VP, R-S

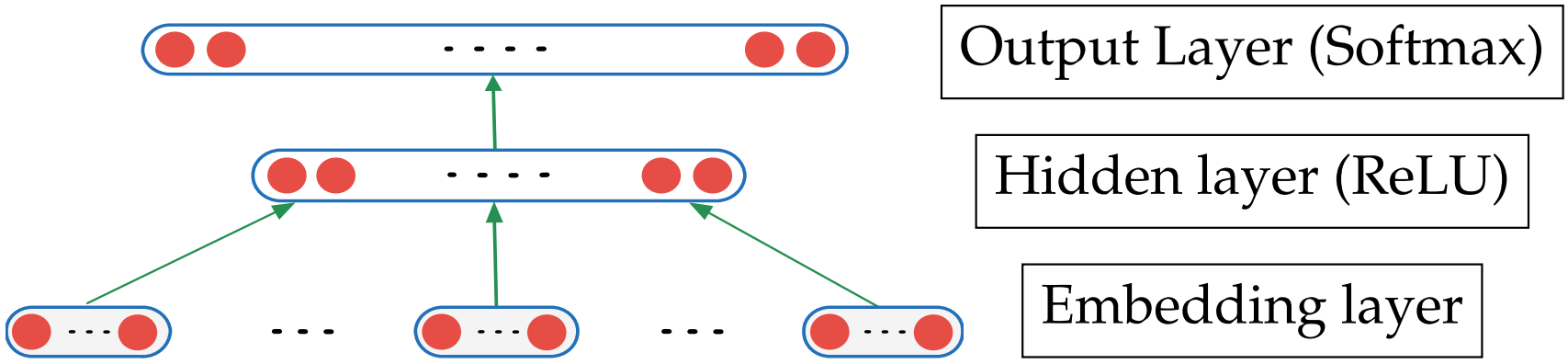
Predicted set : $\{(NP, 0, 1), (NP, 2, 4), (VP, 1, 4), (S, 0, 4)\}$

SHIFT-REDUCE

- Requires a **binary** treebank – order-0 head-Markovisation
  - Binarization introduces temporary symbols
- **Lexicalisation** : *Reduction-Left* or *Reduction-Right* for head assignation and feature extraction
- No unary reductions, except possibly once above pre-terminals

WEIGHTED SEARCH

- Greedy search
- Feedforward neural net to score actions [Chen & Manning 2014] (local classification)



Plan

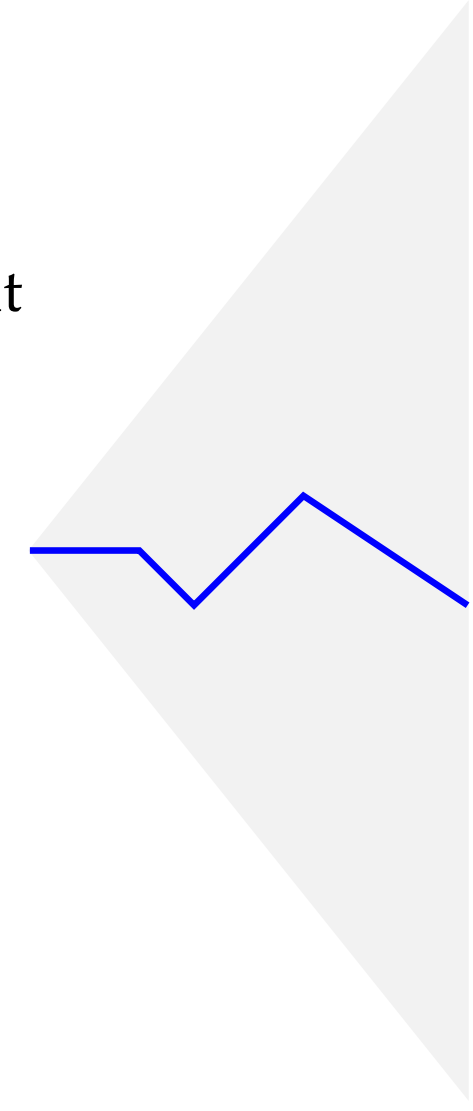
Training with an Oracle

Standard training procedures (**local** classifier) :

- An oracle transforms trees into gold sequences of actions
  - i.e. Shift, Shift, Reduce-NP ...
- Maximize log-likelihood of dataset

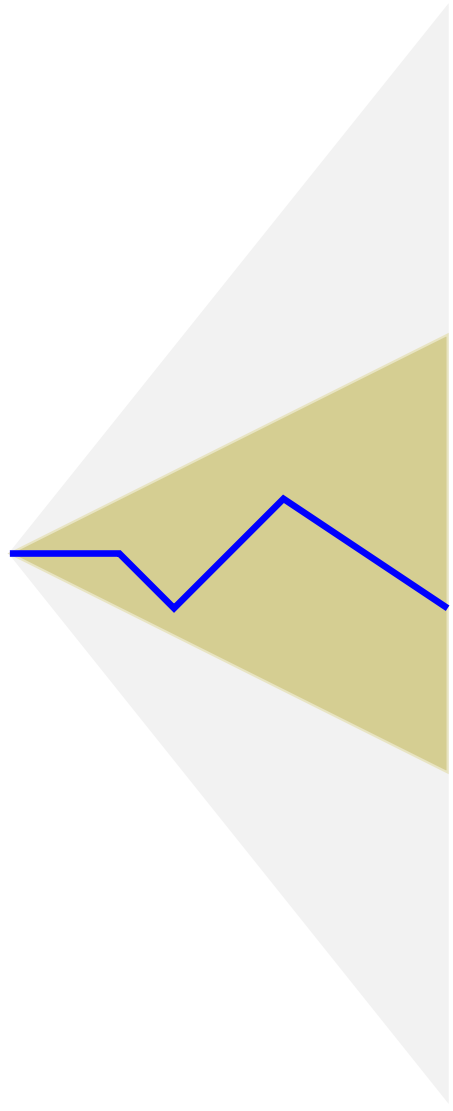
Training with an Oracle : Limitations

- The parser only sees gold configurations at training time
- Only a tiny portion of the search space is used during training
- Error propagation : the parser is not good at recovering after errors



Dynamic Oracle

- Solution : explore **a greater part of the search space** during training
- Sample paths
- Train the parser to predict the best actions from the (possibly non-gold) configurations
- Hopefully learn to recover after usual mistakes



Dynamic Oracle

Dynamic oracle

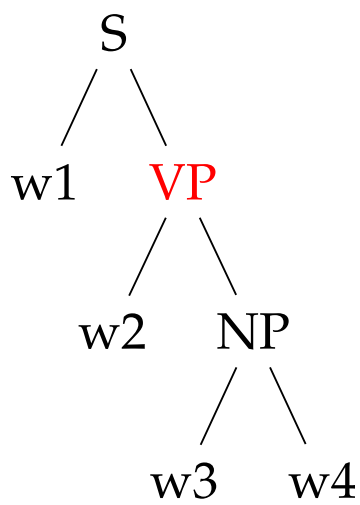
Oracle well-defined for any configuration and that returns the non-empty set of all best actions [Goldberg & Nivre, 2012].

- Available dynamic oracles for most **dependency parsing** transition systems
- Difficulties for constituent parsing
  - **Fscore does not decompose** easily (vs UAS/LAS) $\rightarrow$ use proxy measure
  - **Temporary symbols** (from binarisation) complicate cost calculation

Dynamic Oracle : Cost Function

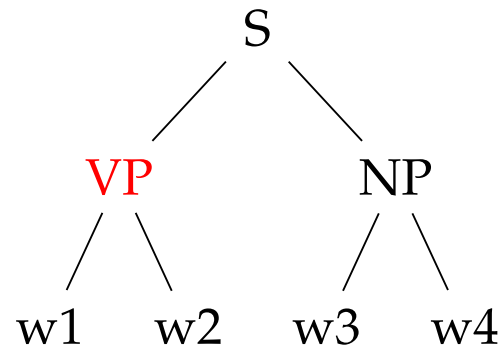
- The best actions minimize some cost function.
- Cost of prediction wrt gold = $|\gamma^* - \hat{\gamma}| = 1$

Gold



$\gamma^* = \{(S, 0, 4), (VP, 1, 4), (NP, 2, 4)\}$

Prediction



$\hat{\gamma} = \{(S, 0, 4), (VP, 0, 2), (NP, 2, 4)\}$

Dynamic Oracle : Cost Function

- Cost of an action = cost difference between best reachable tree **after** and **before** the action

Dynamic Oracle : Cost Function Example

- State of parser after 2 Shifts :

w1 w2 | w3 w4

- Gold tree : `(S w1 (VP w2 (NP w3 w4)))`

	Gold reachable set	Action cost
Now	{ (S, 0, 4), (NP, 2, 4), (VP, 1, 4) }	
After Shift	{ (S, 0, 4), (NP, 2, 4), (VP, 1, 4) }	0
After Reduce-X	{ (S, 0, 4), (NP, 2, 4) } ~~(VP, 1, 4)~~	1

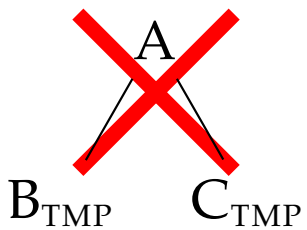
Finding 0-cost Actions

I Consider the set of all labelled binary trees :

- A tree is reachable if all its constituents are individually reachable [we extend Goldberg & Nivre 13]

II Subset of labelled binary trees **well-formed wrt temporary symbols**

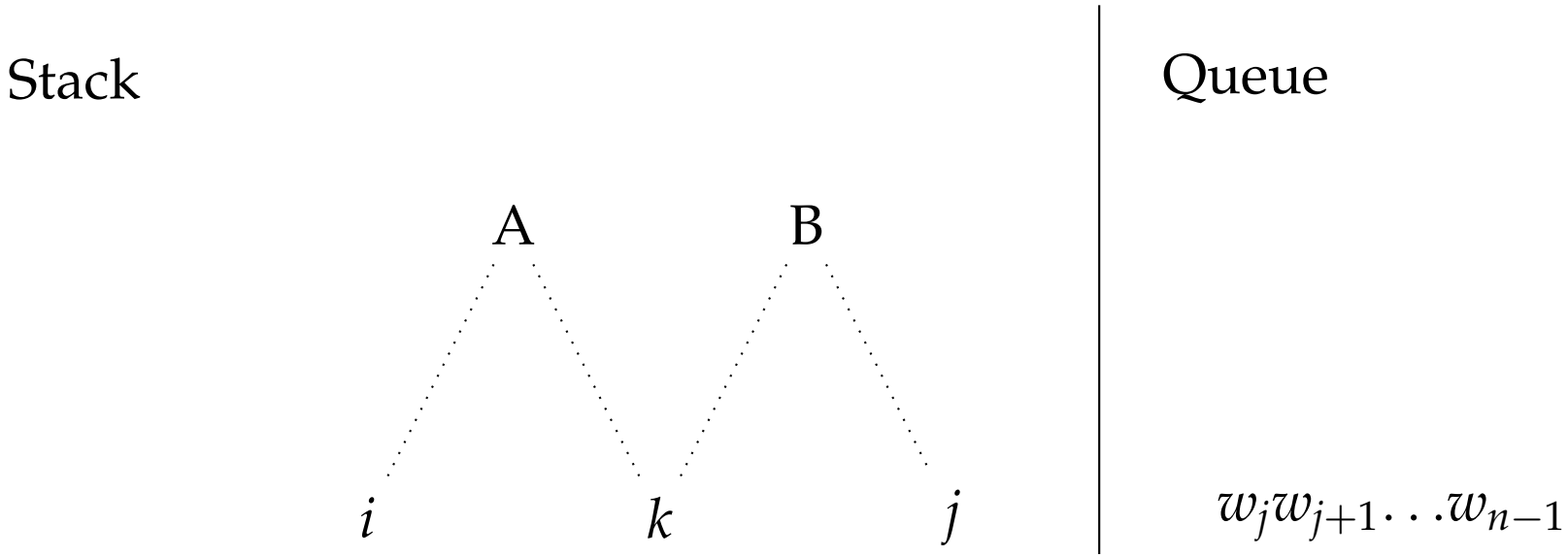
- Grammar constraints to deal with temporary symbols :



Finding 0-cost Actions

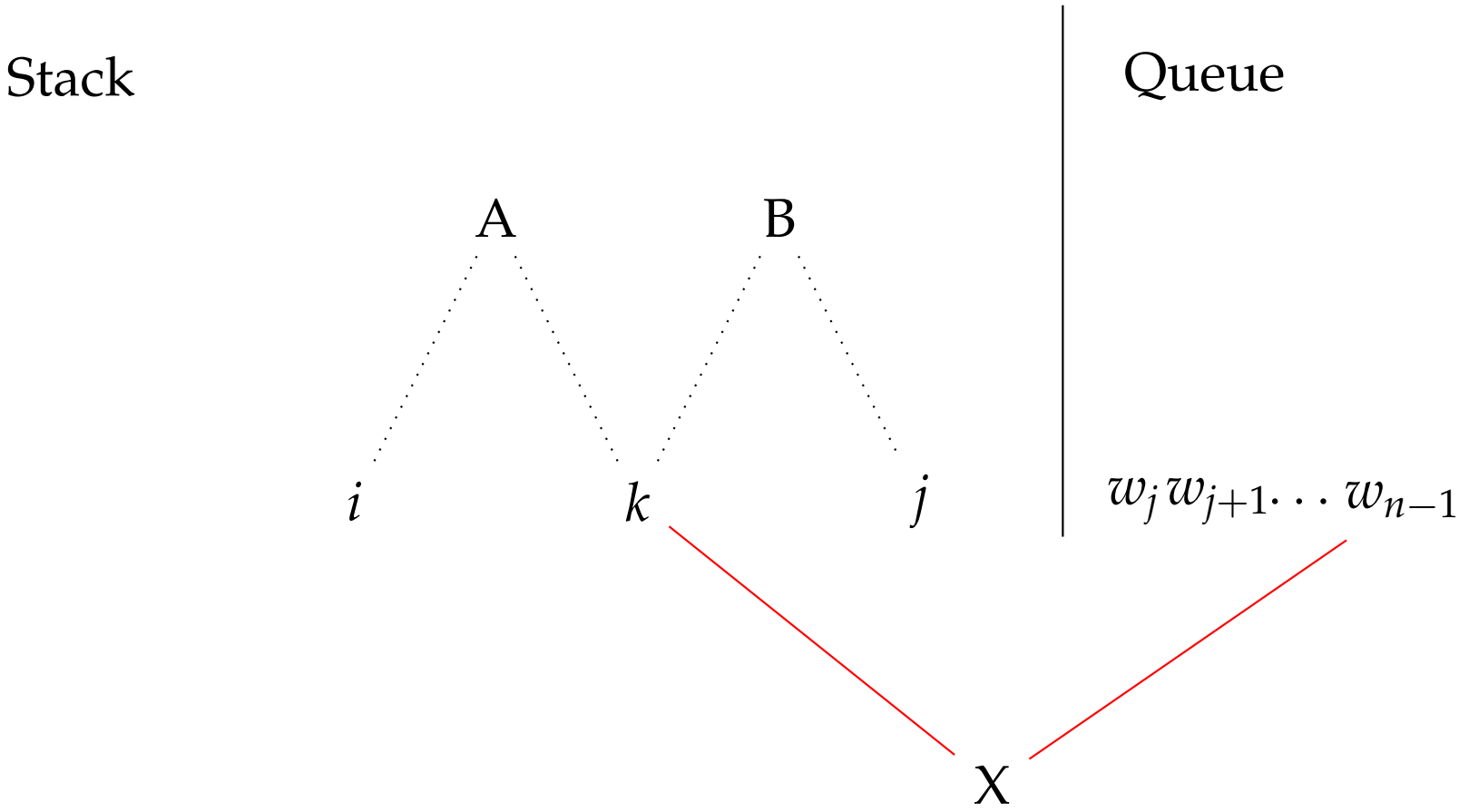
- Derive correct algorithm to find 0-cost action for scenario I
- Add heuristics to handle problematic cases in scenario II
  - Oracle might not be complete or even correct

I. Finding 0-cost Actions



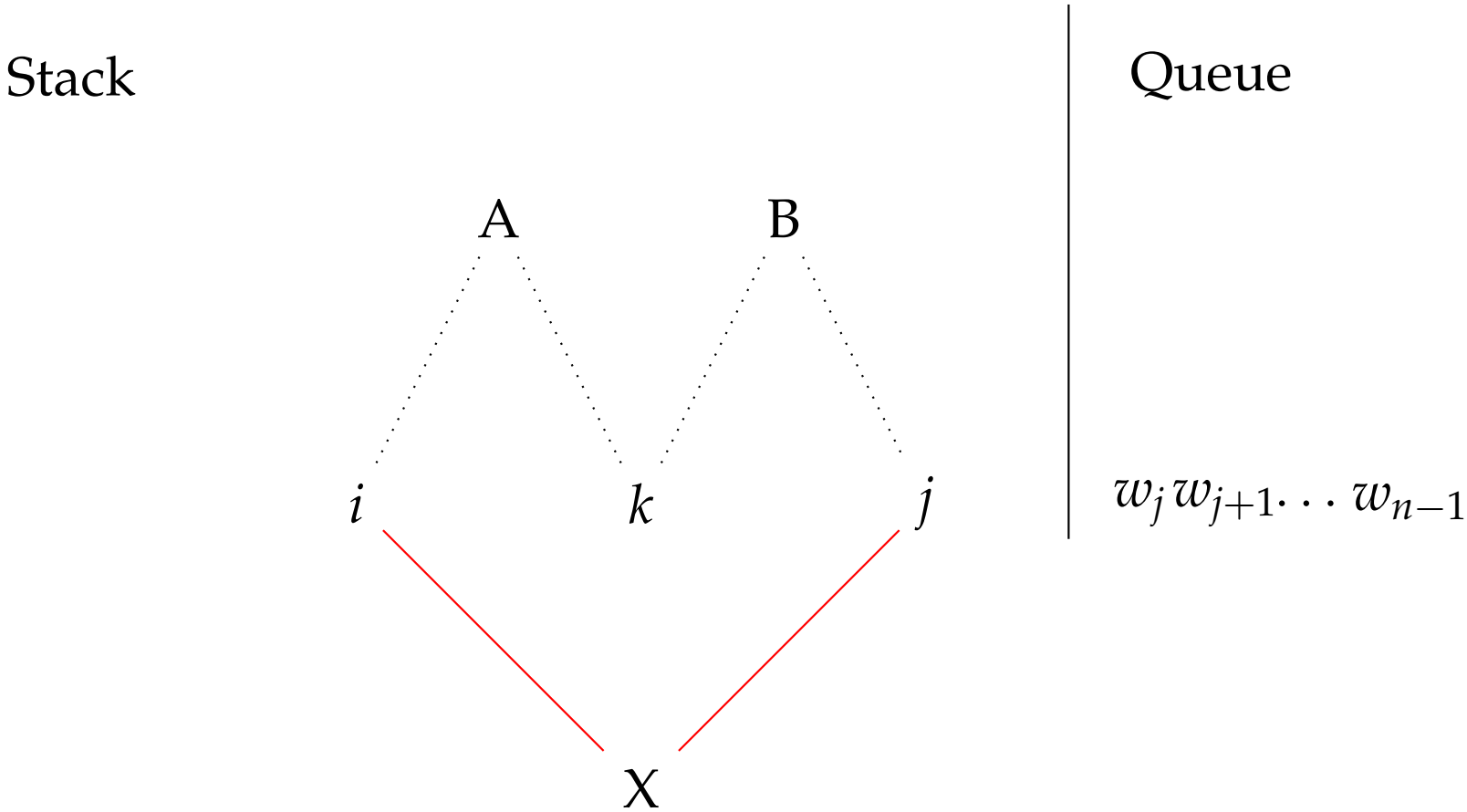
- Assume a configuration
  - B : top of stack, spans tokens from k to j
  - A : 2^{nd} element of stack, spans tokens from i to k

I. FINDING 0-COST ACTIONS : CASE 1



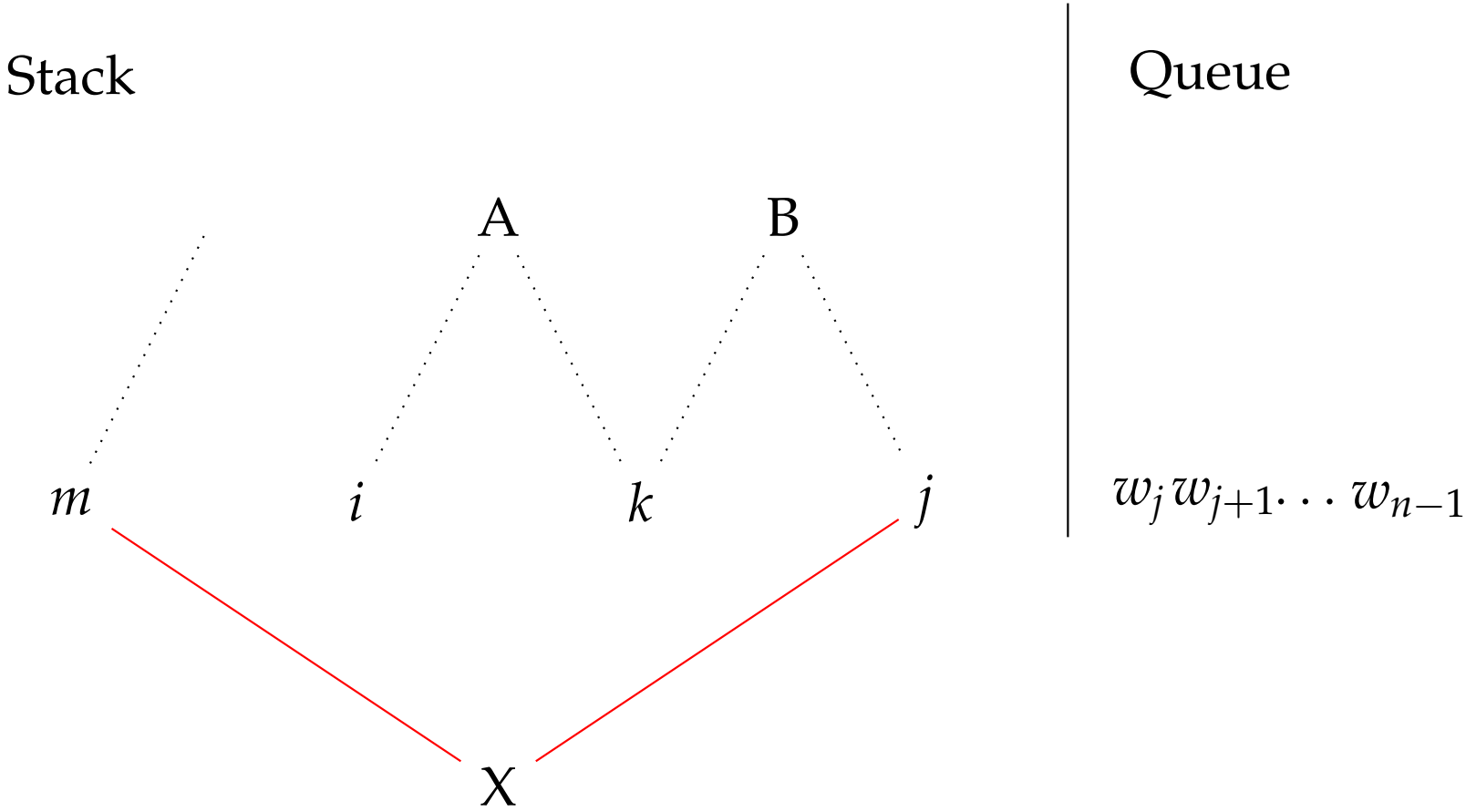
- If (X, k, n) is gold, SHIFT is the only action with cost 0
  - Intuition : (X, k, n) reachable from current configuration, not reachable after REDUCE
  - Shift $\rightarrow$ we lose reachability of $(\cdot, \cdot, j)$ constituents, incompatible with (X, k, n)

I. FINDING 0-COST ACTIONS : CASE 2



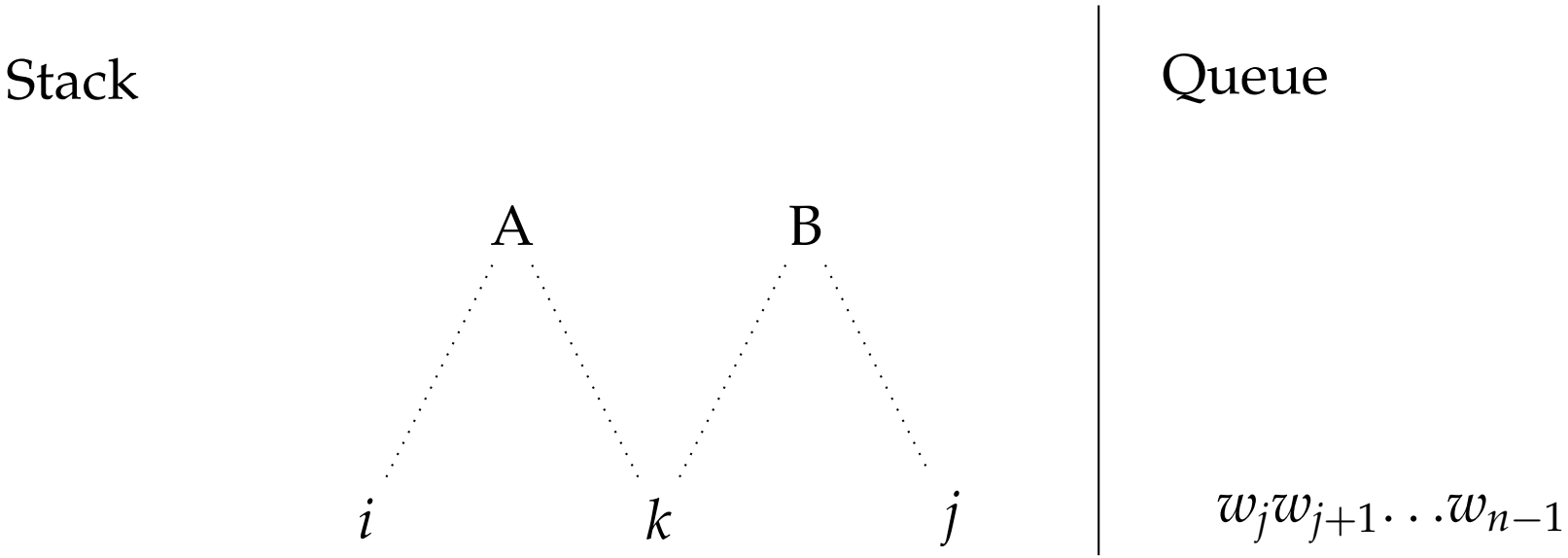
- If there is a gold constituent (X, i, j)
  - REDUCE(X) is the only 0-cost transition

I. FINDING 0-COST ACTIONS : CASE 3



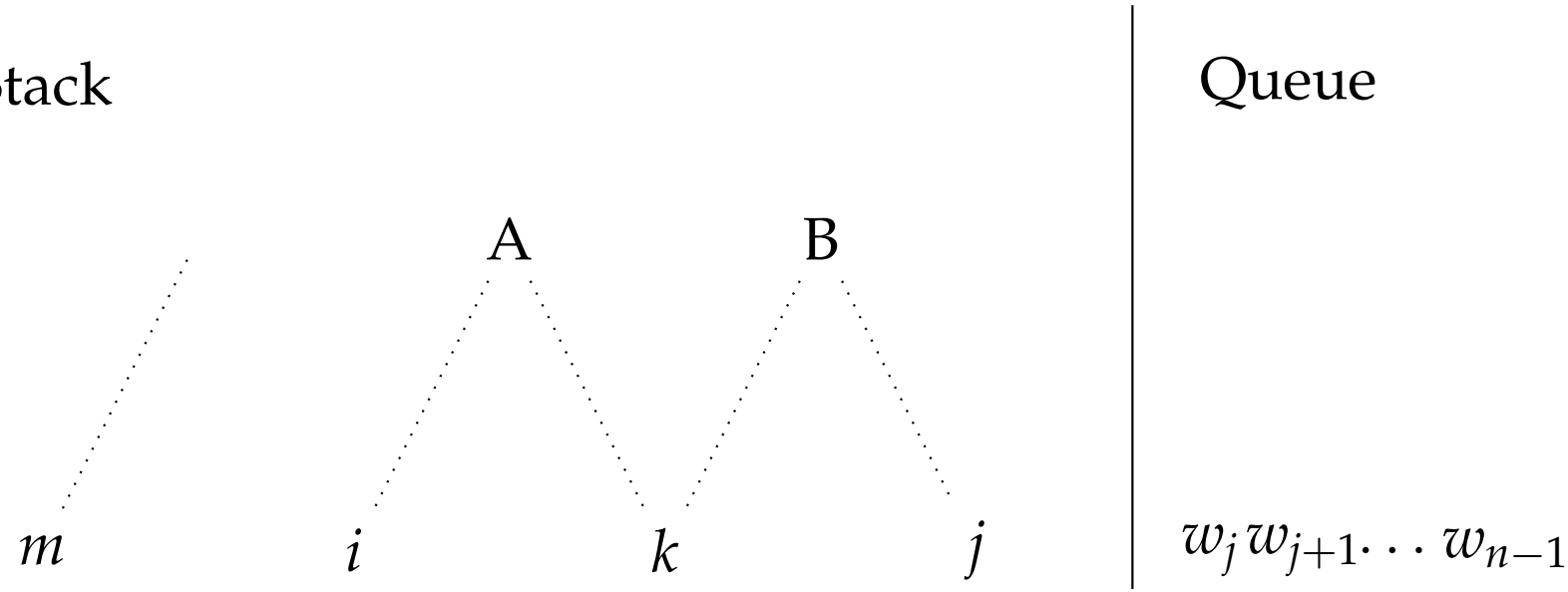
- If there is a reachable gold constituent (X, m, j)
  - $m < i$ is the span beginning of some symbol in the stack
  - Any REDUCE has cost 0 and SHIFT has non-0 cost

I. FINDING 0-COST ACTIONS : CASE 4



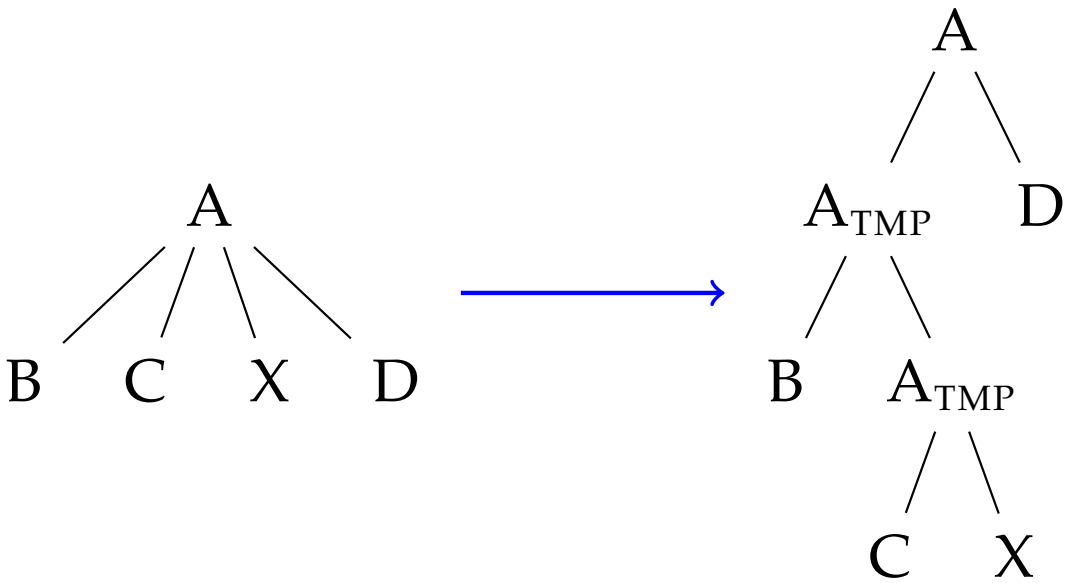
- Backoff case (none of the previous conditions is true) : any possible action has cost 0.

I. Algorithm : Summary

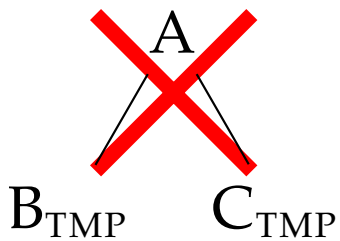


	Test	Cost-0 actions
Case 1	(X, k, n)	{ Shift }
Case 2	(X, i, j)	{ Reduce-X }
Case 3	(X, m, j)	{ Reduce-X, for any non terminal X}
Case 4	backoff	{ Any action }

II. Dealing with Temporary Symbols

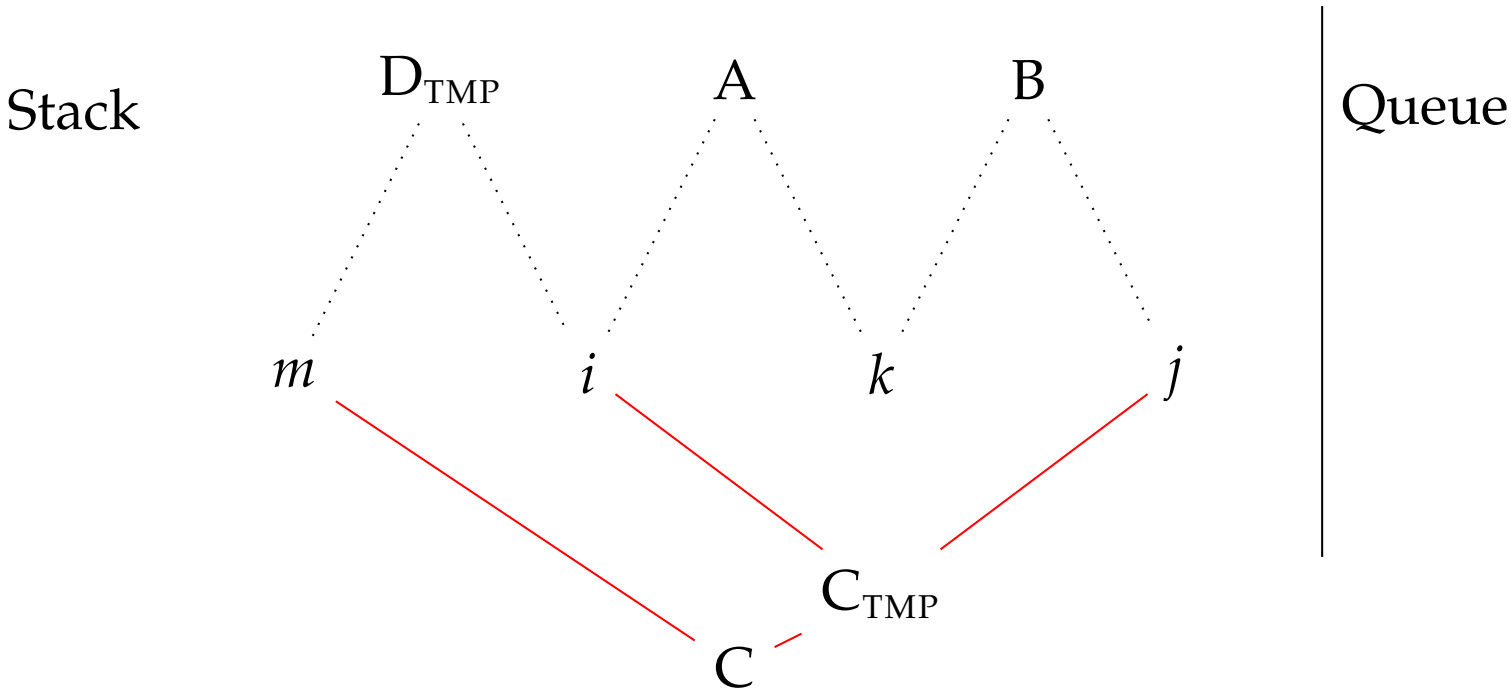


- Head-binarization introduces new temporary symbols
- Implicit grammatical constraints



II. Dealing with Temporary Symbols

- Impossible to compute cost based on individual constituent reachability



- Gold constituents C and C_{TMP} are both individually reachable.
  - C_{TMP} : Reduce(C_{TMP})
  - C : Reduce(C), Reduce(C)
- Reduce(C_{TMP}), ~~Reduce(C)~~ $\rightarrow$ C unreachable

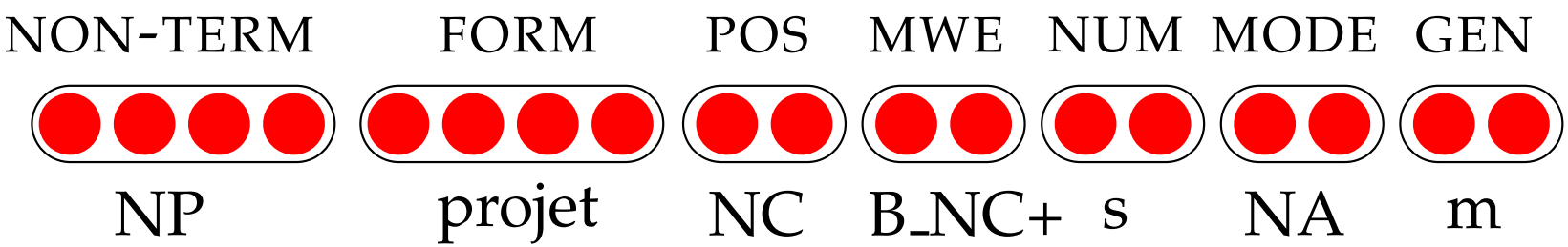
II. HEURISTICS

- Favour overall structure over correct labelling
  - Labelling errors are local, structure errors have global consequences (more costly)
  - If a reduction to a temporary symbol compromises the structure, it has a non-zero cost $\rightarrow$ use any non-temporary instead.
- **Reductions** (case 3 and backoff) : if temporary and non-temporary reductions have cost 0, learn to predict temporary label
  - Non-temporary symbols correspond to n-ary constituents after unbinarization
  - Avoid false positives

Plan

EXPERIMENTAL SETUP : FEATURES

- Non-terminals
- Tuple-structured tokens :
  [word-form, pos-tag, morph_1, ..., morph_n]
  - Morphological attribute e.g. case, gender, tense.
  - English : $n = 0$, Basque : $n = 9$
- e.g. lexicalised non-terminal French 'NP[projet]' (project)



Experimental Setup

Training :

- Averaged SGD
- Strictly supervised (no pretrained embeddings)
- *Training with exploration* [Goldberg & Nivre 2013] (slightly modified)
  - When training online : follow the oracle's prediction with probability $1 - p$ or the parser's prediction (p)

Datasets :

- 9 languages of SPMRL, English : PTB-WSJ
- Predicted tags and morphology (MARMOT, Müller et al. 2013)

RESULTS

	Arabic	Basque	French	German	Hebrew	Hungarian	Korean	Polish	Swedish	Avg	English
	Test F1 (EVALBSPMRL)										Test F1 (EVALB)
static	79.77	85.91	79.62	79.20	88.64	90.54	84.53	92.69	81.45	84.71	88.0
dynamic	80.71	86.24	79.91	80.15	88.69	90.51	85.10	92.96	81.74	85.11	88.6
Δ	**+0.9**	**+0.3**	**+0.3**	**+1.0**	+0.1	0	**+0.6**	**+0.3**	**+0.3**	**+0.4**	**+0.6**

Absolute improvement

- $\Delta \in [0, 0.6]$
- Average improvement on SPMRL dataset : $\Delta = +0.4$

RESULTS

		Arabic	Basque	French	German	Hebrew	Hungarian	Korean	Polish	Swedish	Avg
Test F1 (EVALBSPMRL)											
static	greedy	79.77	85.91	79.62	79.20	88.64	90.54	84.53	92.69	81.45	84.71
dynamic	greedy	80.71	**86.24**	79.91	80.15	88.69	90.51	**85.10**	**92.96**	81.74	**85.11**
dynamic	beam=2	81.14	86.45	80.32	80.68	89.06	90.74	85.17	**93.15**	82.65	85.48
dynamic	beam=4	81.59	86.45	80.48	80.69	89.18	90.73	85.31	93.13	82.77	85.59
dynamic	beam=8	**81.80**	**86.48**	80.56	80.74	89.24	**90.76**	**85.33**	93.13	82.80	**85.64**

- Gains of beam search and dynamic oracle add up
- Ceiling effect with small beam size
  - +0.4 with beamsize=2
  - +0.5 with beamsize=8

RESULTS

		Arabic	Basque	French	German	Hebrew	Hungarian	Korean	Polish	Swedish	Avg
Test F1 (EVALBSPMRL)											
Björkelund 2014		81.32*	**88.24**	**82.53**	**81.66**	**89.80**	**91.72**	83.81	90.50	**85.50**	**86.12**
Durrett & Klein 2015	CKY	80.24	85.41	**81.25**	**80.95**	88.61	**90.66**	82.23	**92.97**	**83.45**	**85.09**
Crabbé 2015	beam=8	**81.31**	84.94	80.84	79.26	**89.65**	90.14	82.65	92.66	83.24	84.97
static	greedy	79.77	85.91	79.62	79.20	88.64	90.54	84.53	92.69	81.45	84.71
dynamic	greedy	80.71	**86.24**	79.91	80.15	88.69	90.51	**85.10**	**92.96**	81.74	**85.11**
dynamic	beam=2	81.14	86.45	80.32	80.68	89.06	90.74	85.17	**93.15**	82.65	85.48
dynamic	beam=4	81.59	86.45	80.48	80.69	89.18	90.73	85.31	93.13	82.77	85.59
dynamic	beam=8	**81.80**	**86.48**	80.56	80.74	89.24	**90.76**	**85.33**	93.13	82.80	**85.64**

- MRL : greedy parser competitive with other single parsers with global learning and more complex inference (CKY, beam-search)

Conclusion

- **Dynamic oracle** for transition-based constituent parsing
  - + feedforward NN
  - + morphological feature embeddings
  - = accurate greedy parser for MRL languages

Conclusion

- **Dynamic oracle** for transition-based constituent parsing
  - + feedforward NN
  - + morphological feature embeddings
  - = accurate greedy parser for MRL languages

Thanks !