An AMR Aligner Tuned by Transition-based Parser

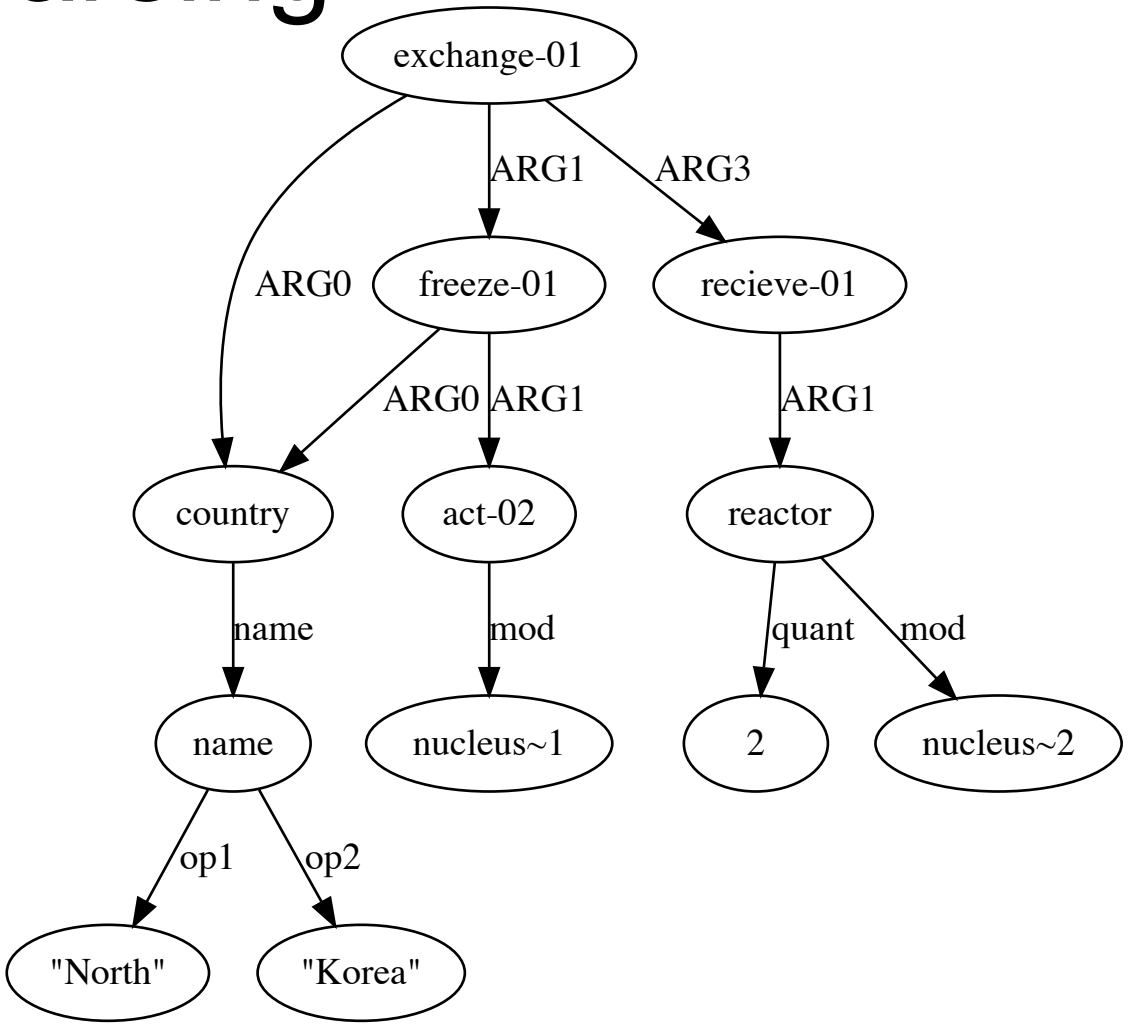
Yijia Liu*, Wanxiang Che, Bo Zheng, Bing Qin, Ting Liu

Research Center for Social Computing and Information Retrieval

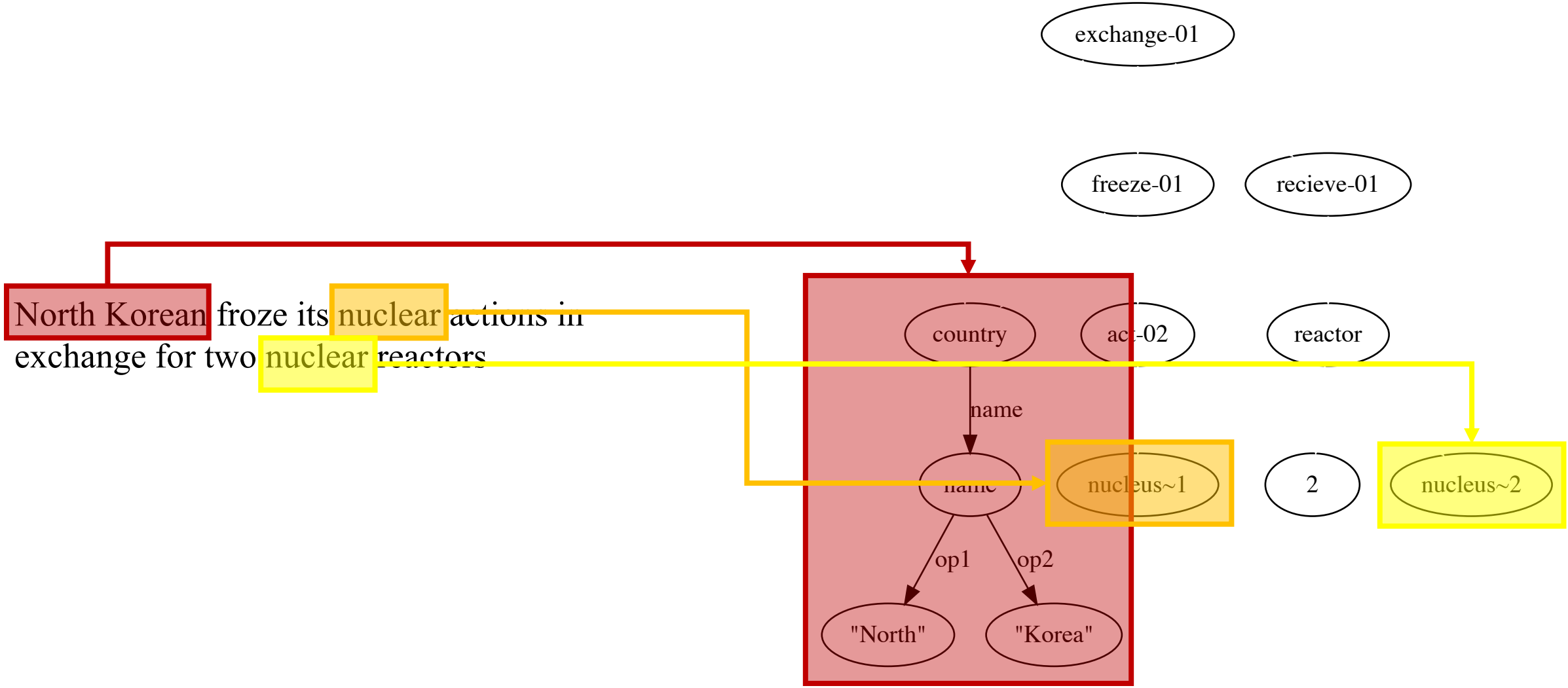
Harbin Institute of Technology

Abstract Meaning Representation (AMR) Parsing

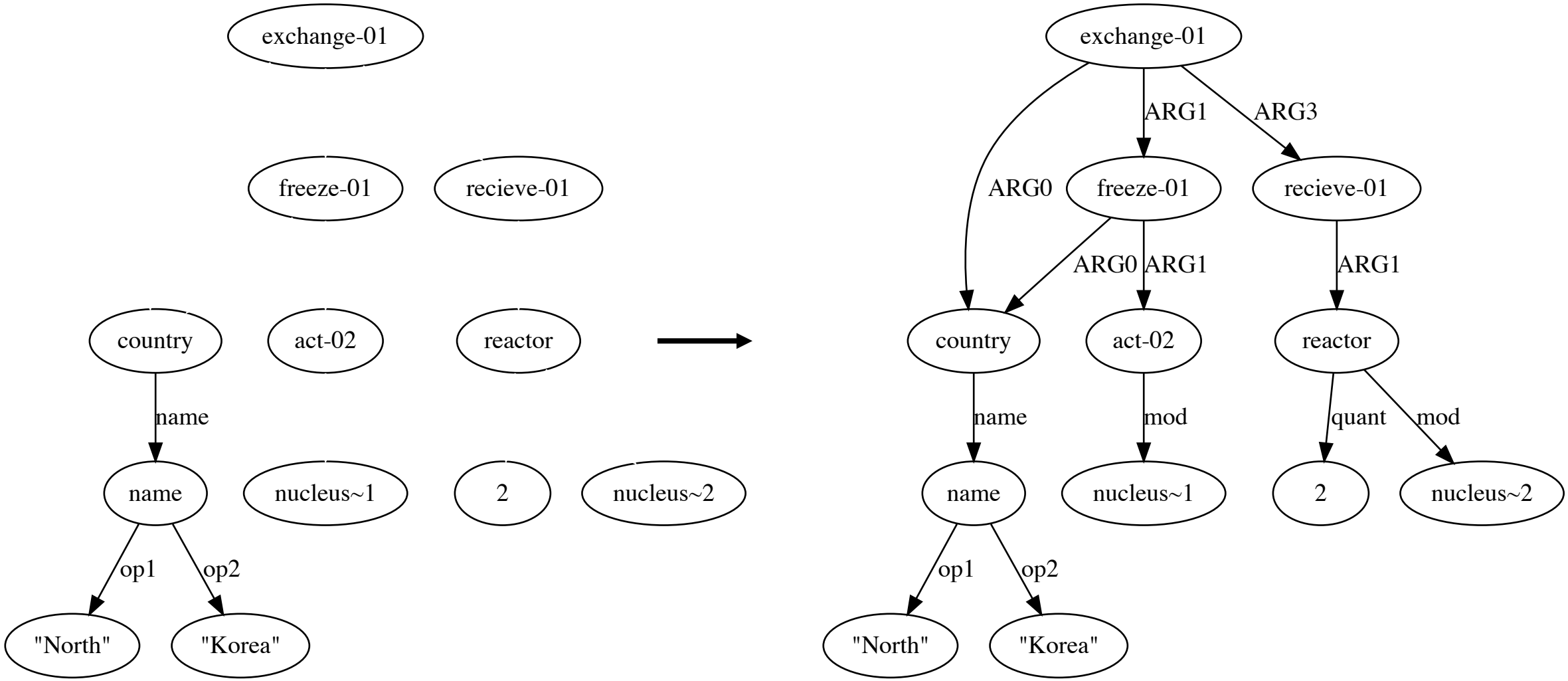
North Korean froze its nuclear actions in exchange for two nuclear reactors



AMR Parsing: Concept Identification



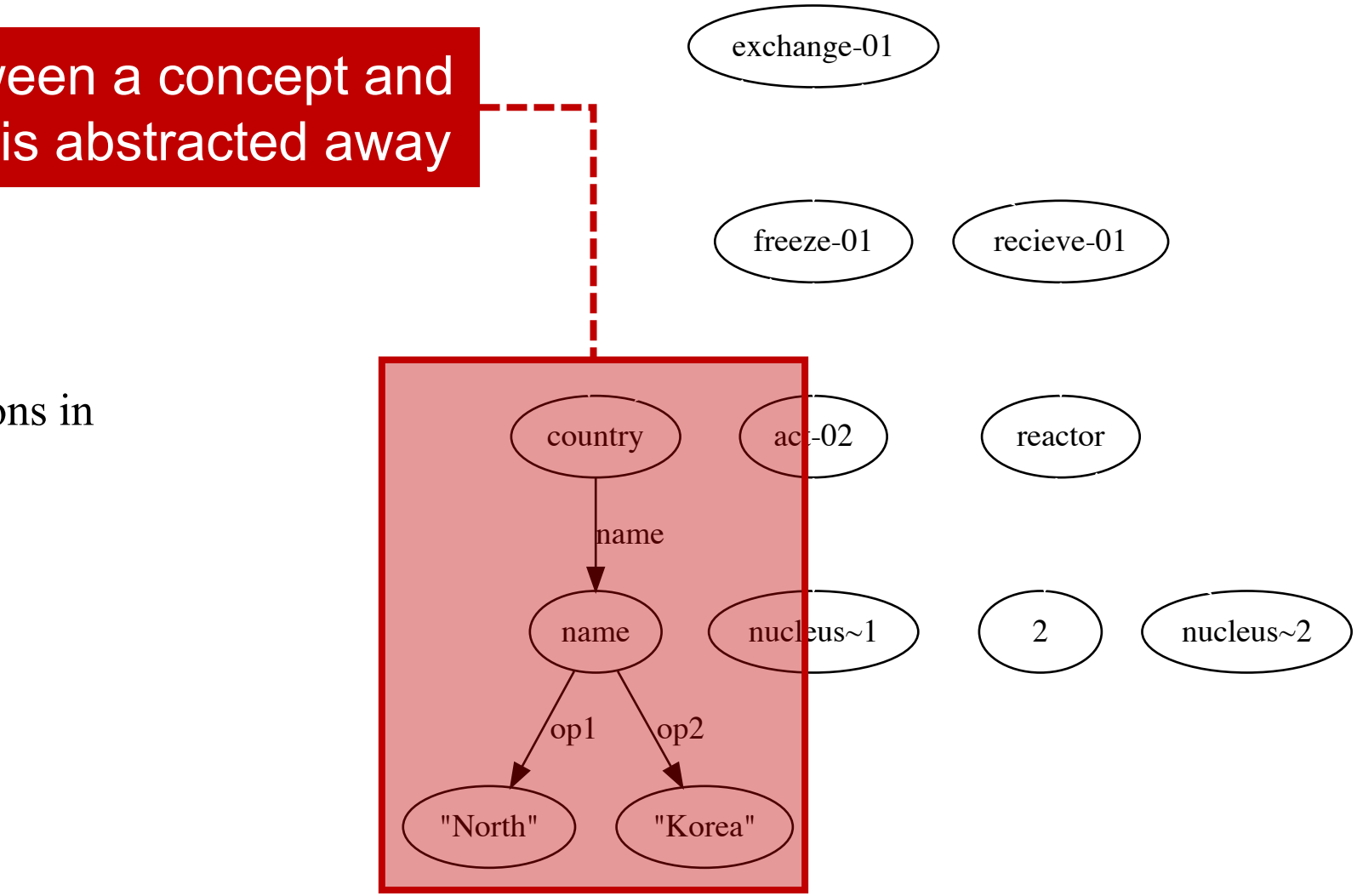
AMR Parsing: Relation Identification



AMR Parsing: Alignment Challenge

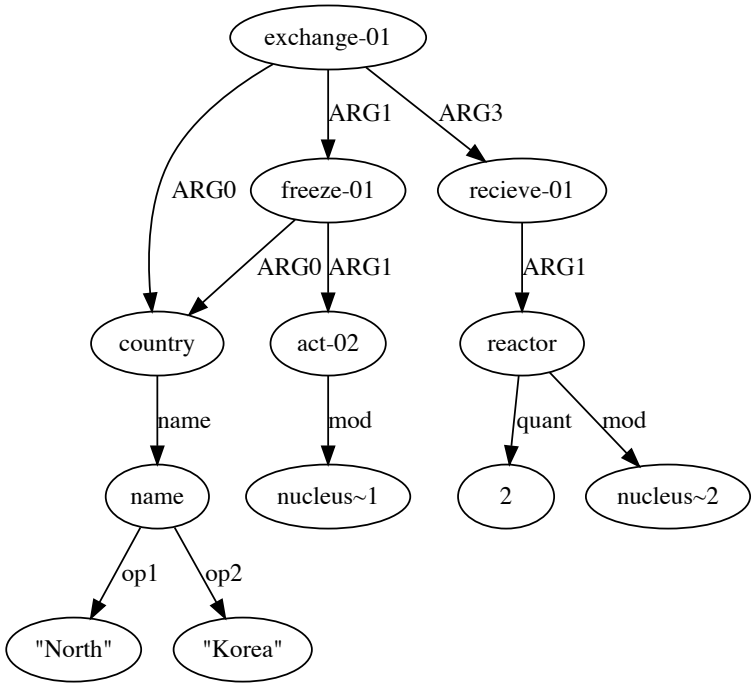
Association between a concept and a span of words is abstracted away

North Korean froze its nuclear actions in exchange for two nuclear reactors



Aligner for Training an AMR Parser: Guessing Alignment

North Korean froze its nuclear actions in exchange for two nuclear reactors

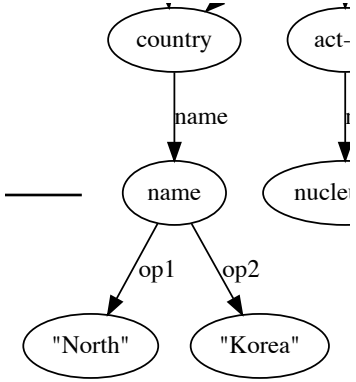


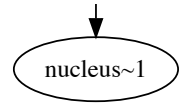
Aligner

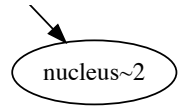
North Korean froze its

its *nuclear* actions

two *nuclear* reactors







JAMR Aligner (Flanigan et al. 2014)

A Set of Rules

1. **(Named Entity)** Applies to `name` concepts and their `opn` children. Matches a span that exactly matches its `opn` children in numerical order.

2. **(Fuzzy Named Entity)** Applies to `name` concepts and their `opn` children. Matches a span that matches the fuzzy match of each child in numerical order.

3. **(Date Entity)** Applies to `date-entity` concepts and their `day`, `month`, `year` children (if exist). Matches any permutation of day, month, year, (two digit or four digit years), with or without spaces.

4. **(Minus Polarity Tokens)** Applies to − concepts, and matches "no", "not", "non."

5. **(Single Concept)** Applies to any concept. Strips off trailing '-[0-9]+' from the concept (for example

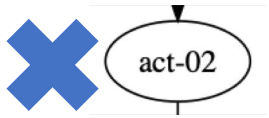
Greedy Decoding

- Manually define rule order
- Apply these rules one by one
- Alignments are mutually excluded, a span once aligned will never be aligned again.

Problems of JAMR

Limited semantic resources

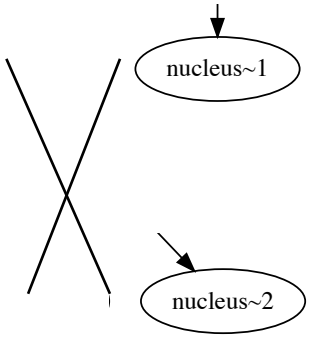
its nuclear *actions*



Ambiguities in matching results

its *nuclear* actions

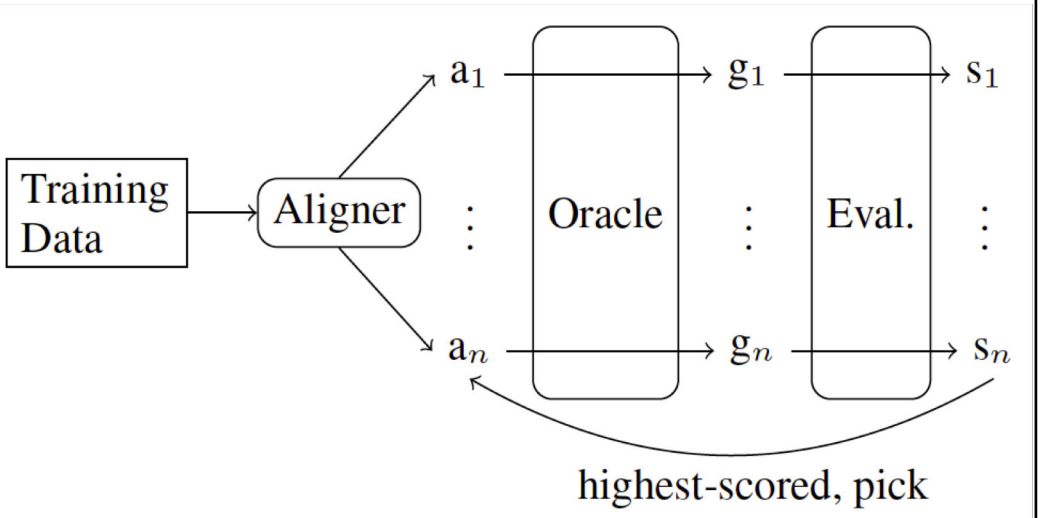
two *nuclear* reactors



Parser training does not feed back to alignment

Our Contributions

A novel aligner



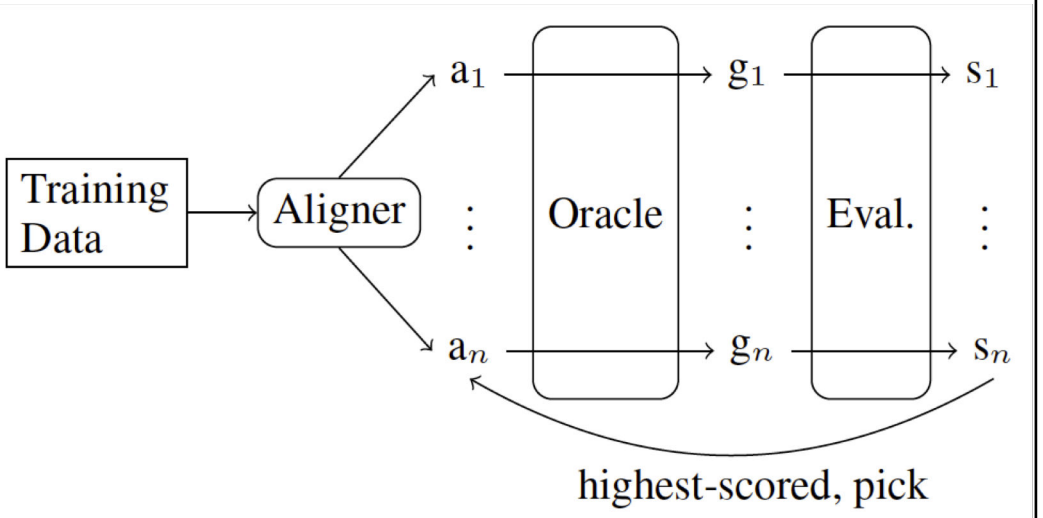
A transition-based parser

Transition	Current State	Resulting State	Description
DROP	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, \beta, A]$	pops out the word that doesn't convey any semantics (e.g., function words and punctuations).
MERGE	$[\sigma\|s_0, \delta, b_0\|b_1\|\beta, A]$	$[\sigma\|s_0, \delta, b_0\_b_1\|\beta, A]$	concatenates a sequence of words into a span, which can be derived as a named entity (`name`) or `date-entity`.
CONFIRM(c)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, c\|\beta, A]$	derives the first element of the buffer (a word or span) into a concept c.
ENTITY(c)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, c\|\beta, A \cup \text{relations}(c)]$	a special form of CONFIRM that derives the first element into an entity and builds the internal entity AMR fragment.
NEW(c)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, c\|b_0\|\beta, A]$	generates a new concept c and pushes it to the front of the buffer.
LEFT(r)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, b_0\|\beta, A \cup \{s_0 \xleftarrow{r} b_0\}]$	links a relation r between the top concepts on the stack and the buffer.
RIGHT(r)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, b_0\|\beta, A \cup \{s_0 \xrightarrow{r} b_0\}]$	
CACHE	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma, s_0\|\delta, b_0\|\beta, A]$	passes the top concept of the stack onto the deque.
SHIFT	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0\|\delta\|b_0, [], \beta, A]$	shifts the first concept of the buffer onto the stack along with those on the deque.
REDUCE	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma, \delta, b_0\|\beta, A]$	pops the top concept of the stack.

$$p(a|s) = \frac{\exp\{g_a \cdot \textsc{StackLSTM}(s) + b_a\}}{\sum_{a'} \exp\{g_{a'} \cdot \textsc{StackLSTM}(s) + b_{a'}\}}$$

Our Contributions

A novel aligner



A transition-based parser

Transition	Current State	Resulting State	Description
DROP	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, \beta, A]$	pops out the word that doesn't convey any semantics (e.g., function words and punctuations).
MERGE	$[\sigma\|s_0, \delta, b_0\|b_1\|\beta, A]$	$[\sigma\|s_0, \delta, b_0\_b_1\|\beta, A]$	concatenates a sequence of words into a span, which can be derived as a named entity (`name`) or `date-entity`.
CONFIRM(c)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, c\|\beta, A]$	derives the first element of the buffer (a word or span) into a concept `c`.
ENTITY(c)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, c\|\beta, A \cup \text{relations}(c)]$	a special form of CONFIRM that derives the first element into an entity and builds the internal entity AMR fragment.
NEW(c)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, c\|b_0\|\beta, A]$	generates a new concept `c` and pushes it to the front of the buffer.
LEFT(r)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, b_0\|\beta, A \cup \{s_0 \xleftarrow{r} b_0\}]$	links a relation r between the top concepts on the stack and the buffer.
RIGHT(r)	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0, \delta, b_0\|\beta, A \cup \{s_0 \xrightarrow{r} b_0\}]$	
CACHE	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma, s_0\|\delta, b_0\|\beta, A]$	passes the top concept of the stack onto the deque.
SHIFT	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma\|s_0\|\delta\|b_0, [], \beta, A]$	shifts the first concept of the buffer onto the stack along with those on the deque.
REDUCE	$[\sigma\|s_0, \delta, b_0\|\beta, A]$	$[\sigma, \delta, b_0\|\beta, A]$	pops the top concept of the stack.

$$p(a|s) = \frac{\exp\{g_a \cdot \textsc{StackLSTM}(s) + b_a\}}{\sum_{a'} \exp\{g_{a'} \cdot \textsc{StackLSTM}(s) + b_{a'}\}}$$

Our Aligner

Limited semantic resources

Recall more alignments with

- *Glove* embedding (Korean and `Korea`)
- *morphological links* (example and `exemplify`)

Ambiguities in matching results

Parser training does not feed back to alignment

Our Aligner

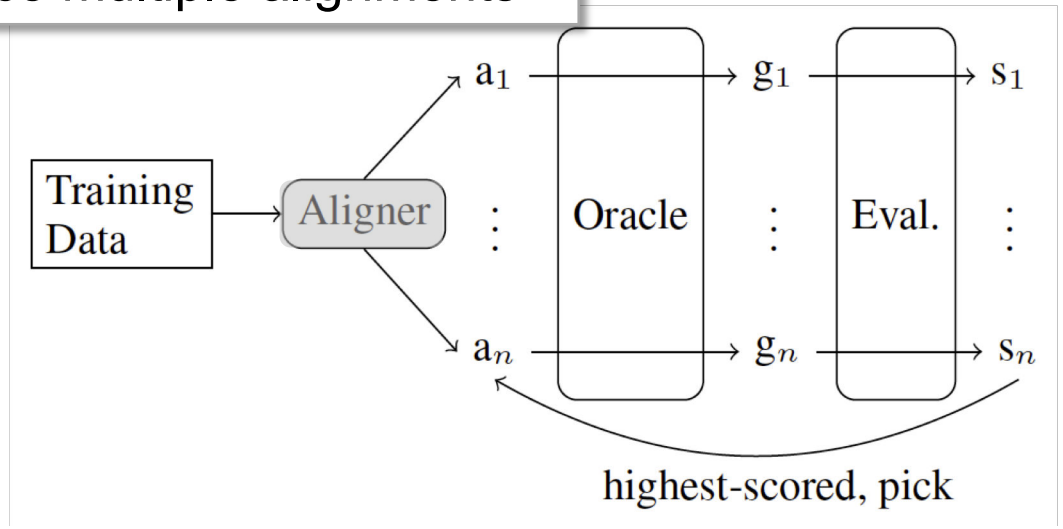
Limited semantic resources

Recall more alignments with

- *Glove* embedding (Korean and `Korea`)
- *morphological links* (example and `exemplify`)

Ambiguities in matching results

Produce multiple alignments



Parser training does not feed back to alignment

Our Aligner

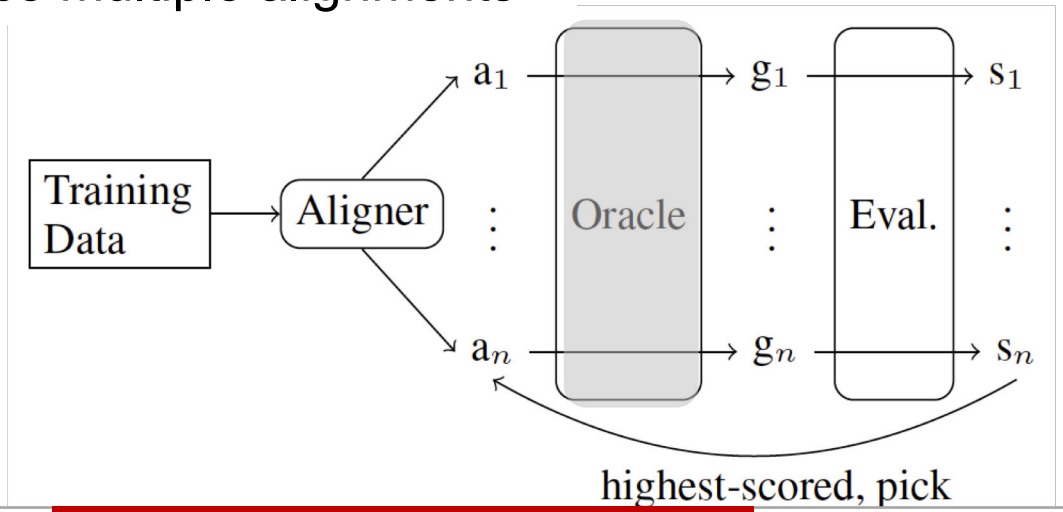
Limited semantic resources:

Recall more alignments with

- *Glove* embedding (Korean and `Korea`)
- *morphological links* (example and `exemplify`)

Ambiguities in matching results

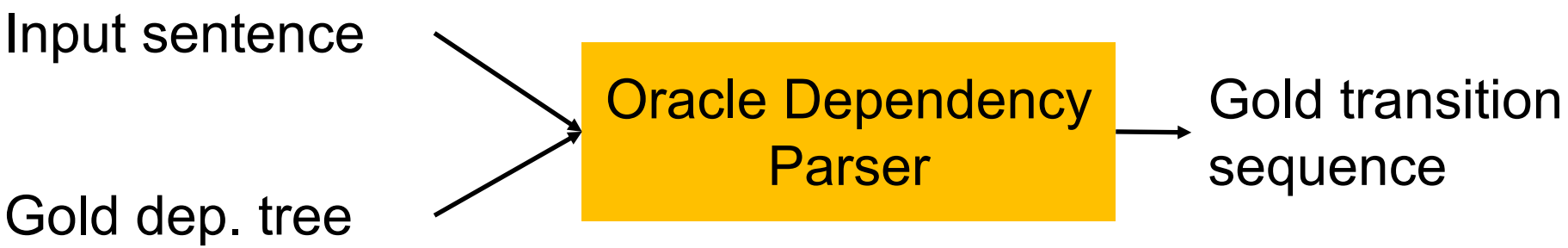
Produce multiple alignments

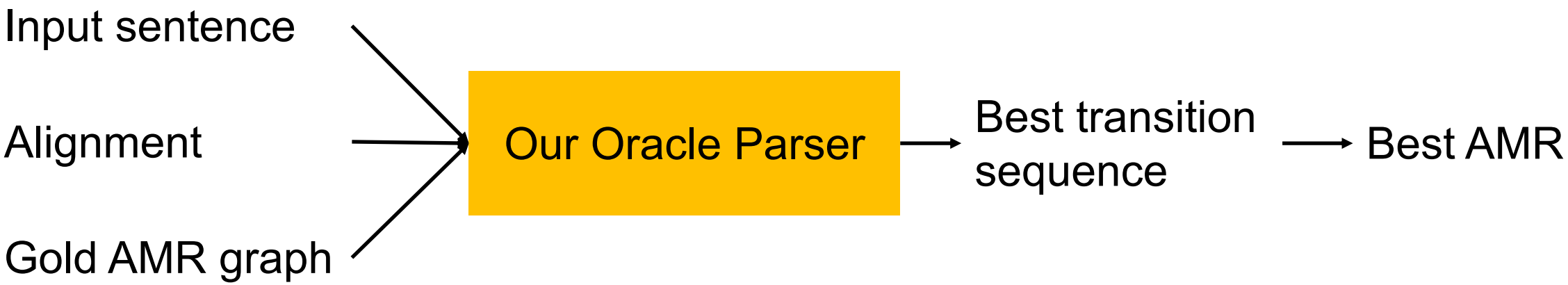


Parser training does not feed back to alignment

Use an oracle transition-based parser to determine the quality of an alignment

Oracle Transition-based Parser





Oracle Transition-based Parser

- State: $s = (\sigma|\mathrm{s}_0, \delta, \mathrm{b}_0|\mathrm{b}_1|\beta, A)$
- Actions:
  - extension of the list-based dependency parser
  - `CONFIRM`, `ENTITY`, and `NEW` for concept identification
    - *Oracle parser* use the alignment to derive these actions

DROP	$[\sigma\|\mathrm{s}_0,\ \delta,\ b_0\|\beta,\ A]$	$[\sigma\|\mathrm{s}_0,\ \delta,\ \beta,\ A]$
MERGE	$[\sigma\|\mathrm{s}_0,\ \delta,\ b_0\|b_1\|\beta,\ A]$	$[\sigma\|\mathrm{s}_0,\ \delta,\ b_0\_b_1\|\beta,\ A]$
CONFIRM(c)	$[\sigma\|\mathrm{s}_0,\ \delta,\ b_0\|\beta,\ A]$	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{c}\|\beta,\ A]$
ENTITY(c)	$[\sigma\|\mathrm{s}_0,\ \delta,\ b_0\|\beta,\ A]$	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{c}\|\beta,\ A \cup \mathrm{relations}(c)]$
NEW(c)	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{b}_0\|\beta,\ A]$	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{c}\|\mathrm{b}_0\|\beta,\ A]$
LEFT(r)	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{b}_0\|\beta,\ A]$	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{b}_0\|\beta,\ A \cup \{\mathrm{s}_0 \overset{\mathrm{r}}{\leftarrow} \mathrm{b}_0\}]$
RIGHT(r)	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{b}_0\|\beta,\ A]$	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{b}_0\|\beta,\ A \cup \{\mathrm{s}_0 \overset{\mathrm{r}}{\rightarrow} \mathrm{b}_0\}]$
CACHE	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{b}_0\|\beta,\ A]$	$[\sigma,\ \mathrm{s}_0\|\delta,\ \mathrm{b}_0\|\beta,\ A]$
SHIFT	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{b}_0\|\beta,\ A]$	$[\sigma\|\mathrm{s}_0\|\delta\|\mathrm{b}_0,\ [\,],\ \beta,\ A]$
REDUCE	$[\sigma\|\mathrm{s}_0,\ \delta,\ \mathrm{b}_0\|\beta,\ A]$	$[\sigma,\ \delta,\ \mathrm{b}_0\|\beta,\ A]$

A Run on a Successful Alignment

stack:	deque:	buffer:
		North Korean froze its nuclear actions in exchange for two nuclear reactors

`ENTITY(country)`

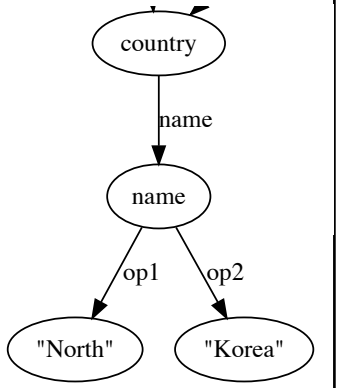
`SHIFT`

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~1`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~2`; reactors – `reactor`

A Run on a Successful Alignment

stack:



deque:

buffer:

froze its nuclear actions in exchange for two nuclear reactors

`CONCEPT(freeze)`

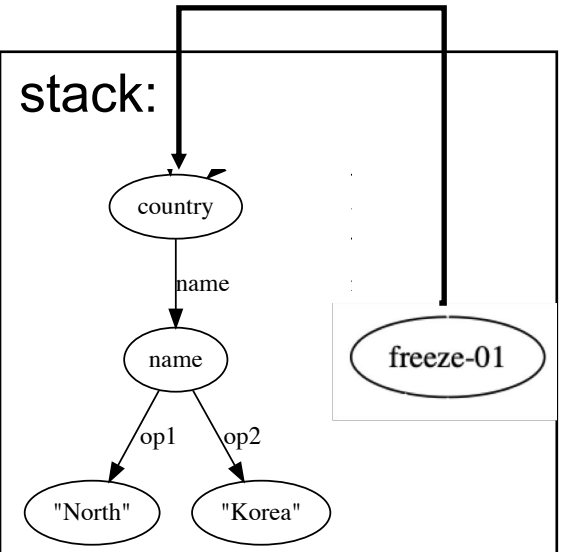
`LEFT(ARG0)`

`SHIFT`

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~1`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~2`; reactors – `reactor`

A Run on a Successful Alignment



stack:

deque:

buffer:

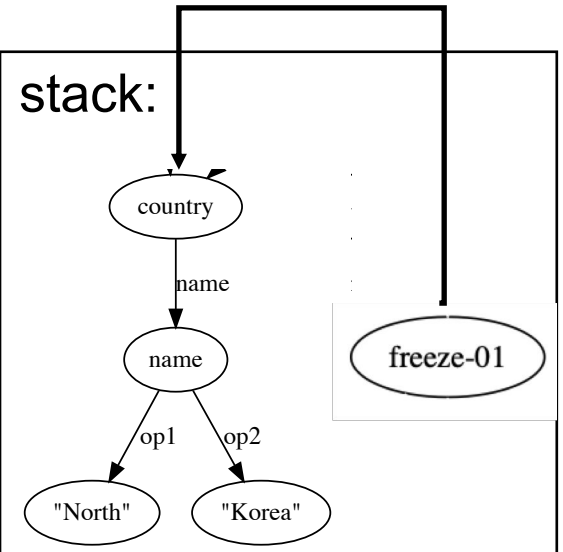
its nuclear actions in exchange for two nuclear reactors

DROP

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~1`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~2`; reactors – `reactor`

A Run on a Successful Alignment



deque:

buffer:

nuclear actions in exchange for two nuclear reactors

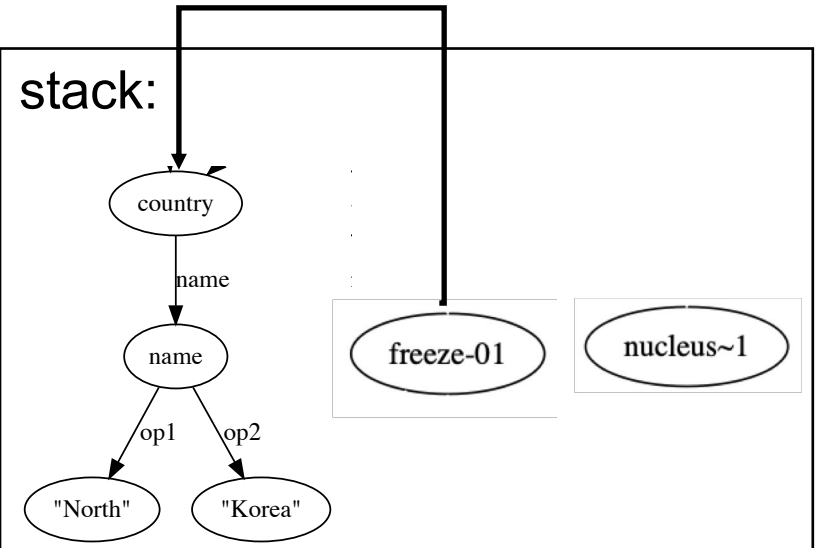
`CONCEPT(nucleus~1)`

`SHIFT`

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~1`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~2`; reactors – `reactor`

A Run on a Successful Alignment



deque:

buffer:

actions in exchange for two nuclear reactors

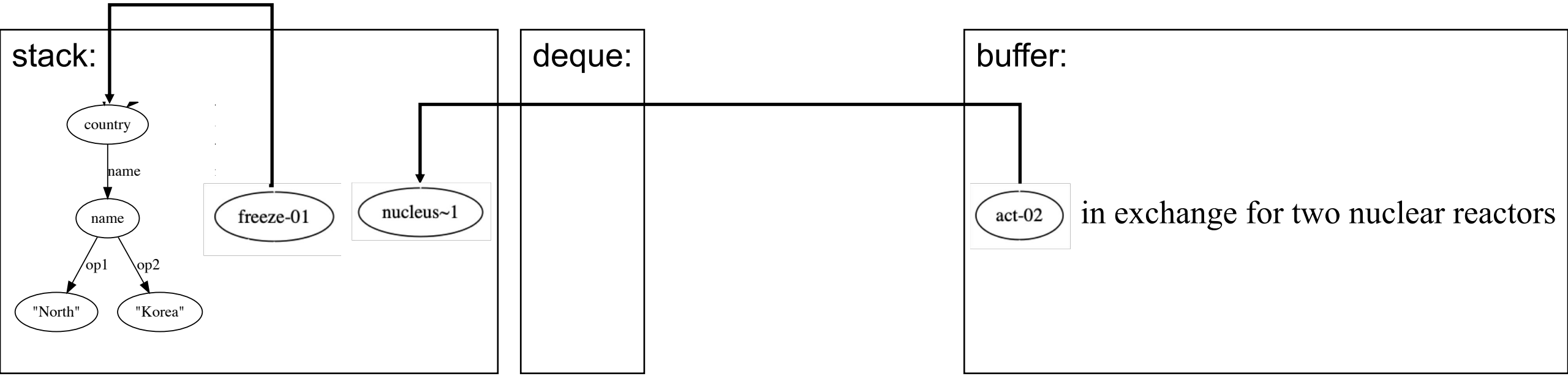
CONCEPT(act)

LEFT(mod)

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~1`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~2`; reactors – `reactor`

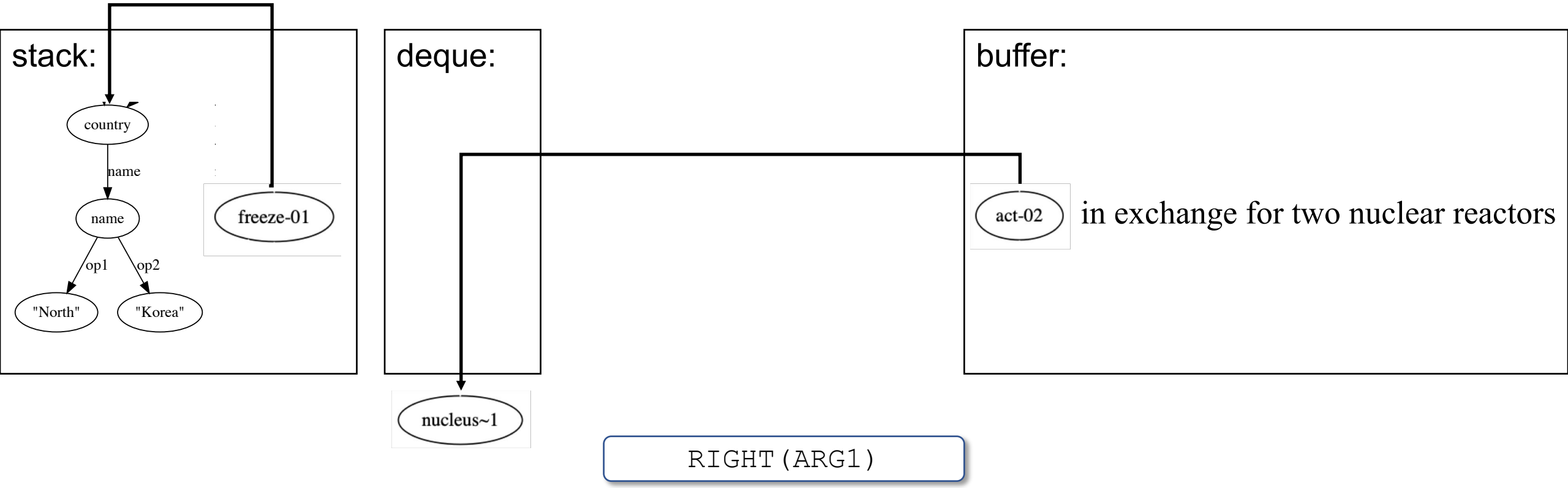
A Run on a Successful Alignment



Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~1`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~2`; reactors – `reactor`

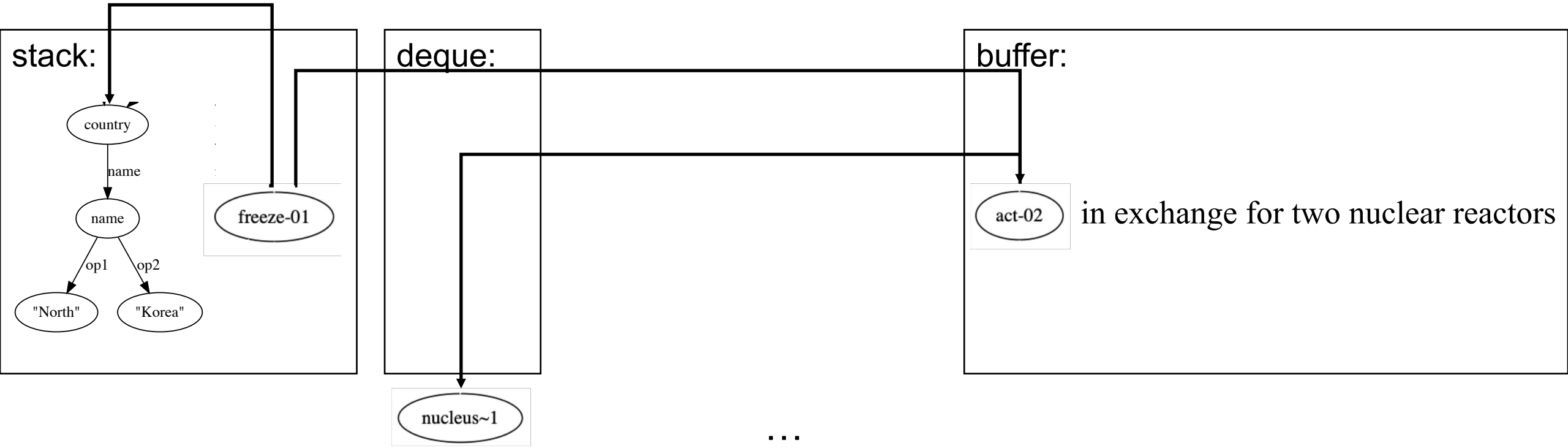
A Run on a Successful Alignment



Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~1`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~2`; reactors – `reactor`

A Run on a Successful Alignment



…

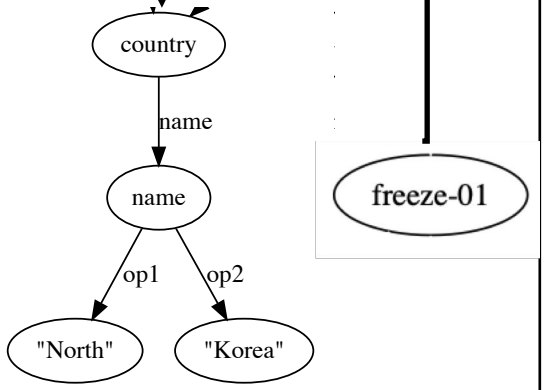
Ends up with **32** actions

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~1`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~2`; reactors – `reactor`

A Run on an Unsuccessful Alignment

stack:



deque:

buffer:

nuclear actions in exchange for two nuclear reactors

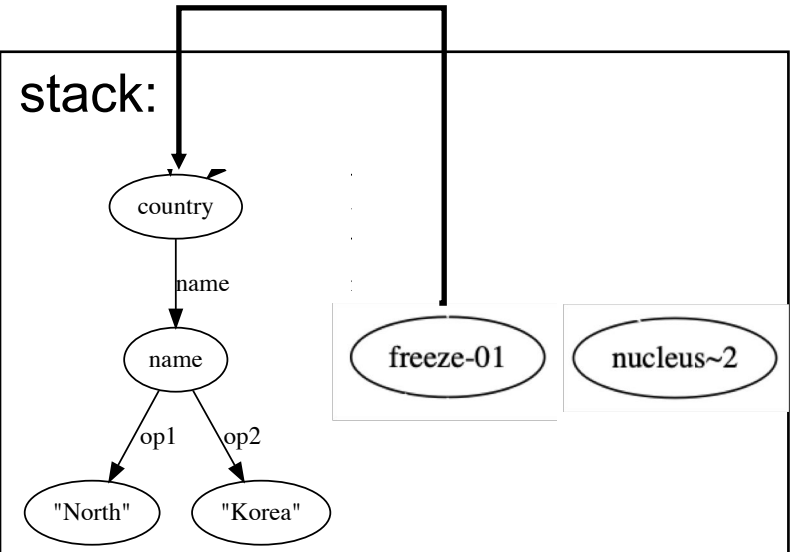
`CONCEPT(nucleus~2)`

`SHIFT`

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~2`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~1`; reactors – `reactor`

A Run on an Unsuccessful Alignment



deque:

buffer:

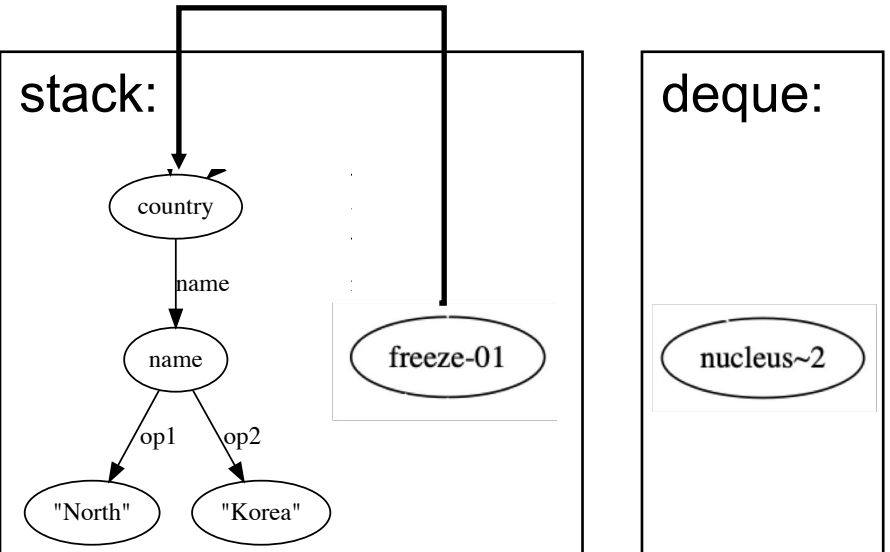
actions in exchange for two nuclear reactors

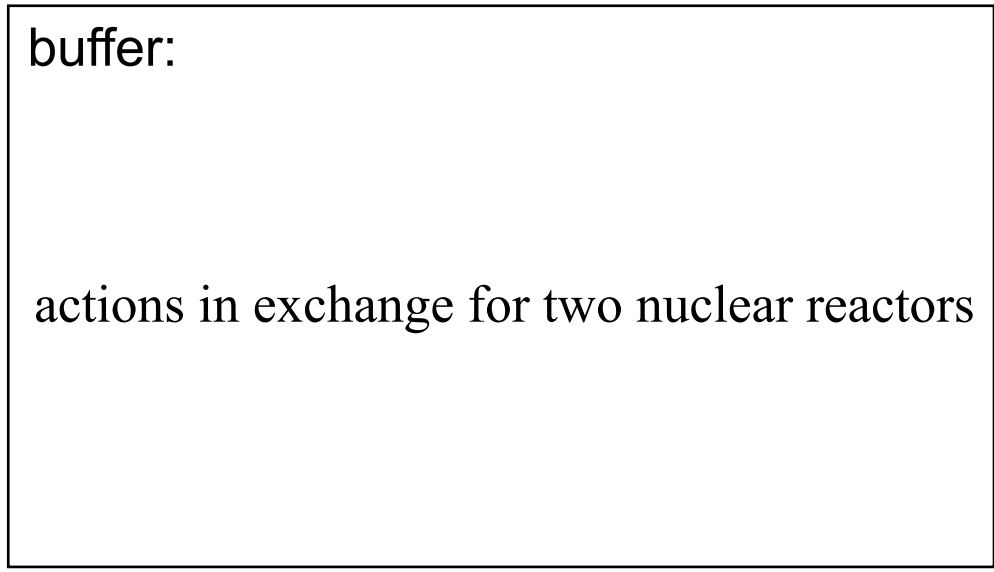
CACHE

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~2`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~1`; reactors – `reactor`

A Run on an Unsuccessful Alignment





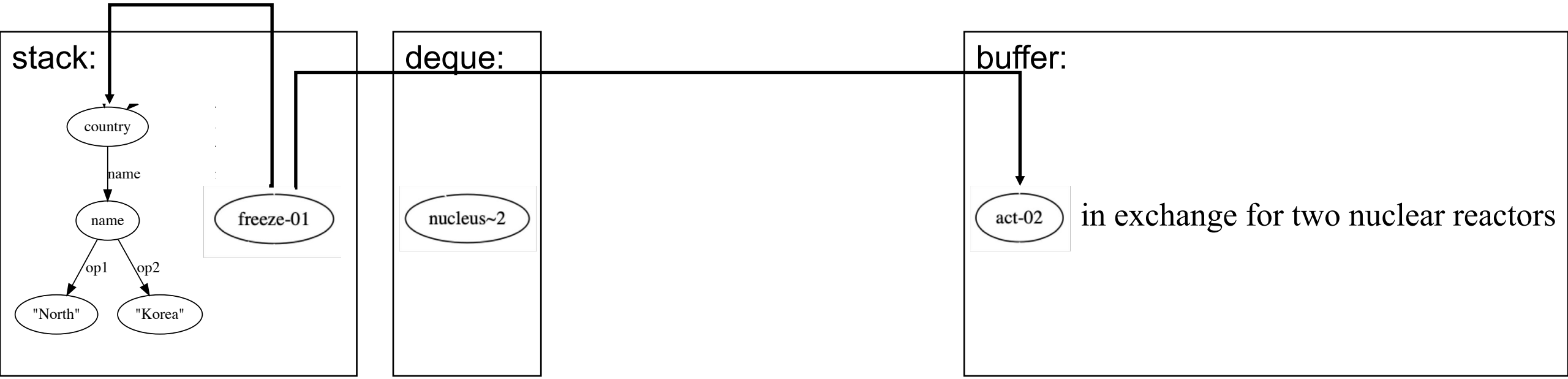
CONCEPT(act)

RIGHT(ARG1)

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~2`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~1`; reactors – `reactor`

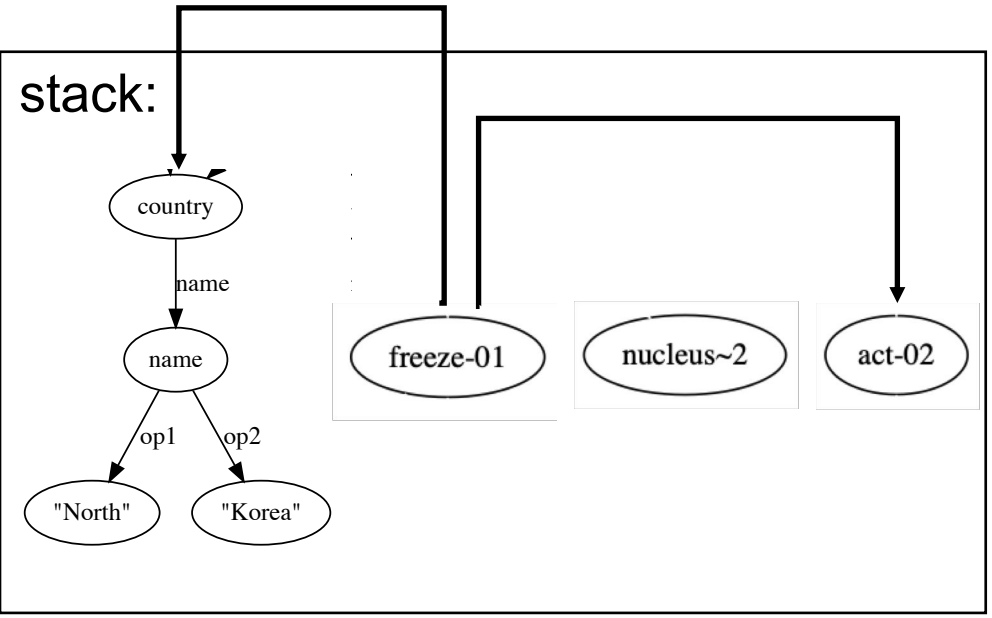
A Run on an Unsuccessful Alignment



Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~2`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~1`; reactors – `reactor`

A Run on an Unsuccessful Alignment



deque:

buffer:

in exchange for two nuclear reactors

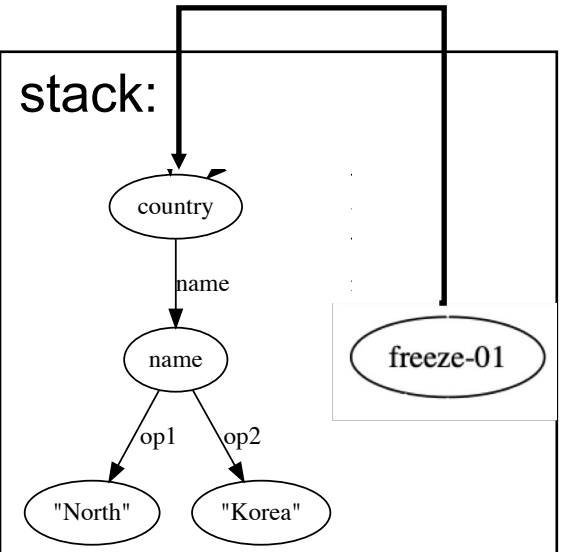
Keep `CACHE`ing

Ends up with **44** actions

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; nuclear[1] – `nucleus~2`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear[2] – `nucleus~1`; reactors – `reactor`

A Run on another Unsuccessful Alignment



deque:

buffer:

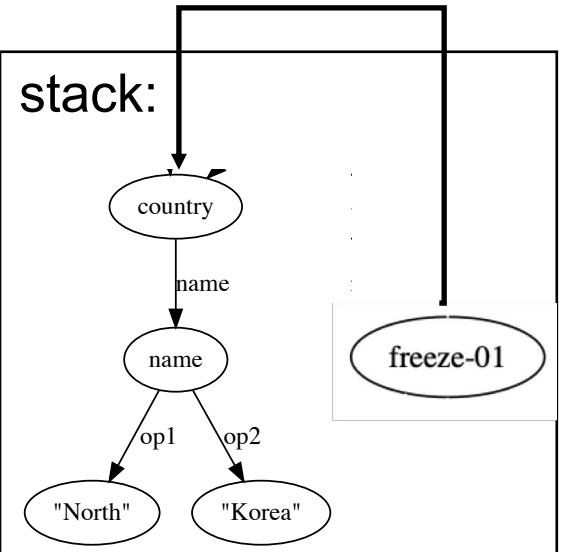
nuclear actions in exchange for two nuclear reactors

DROP

Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear^2 – `nucleus~2`; reactors – `reactor`

A Run on another Unsuccessful Alignment



deque:

buffer:

actions in exchange for two nuclear reactors

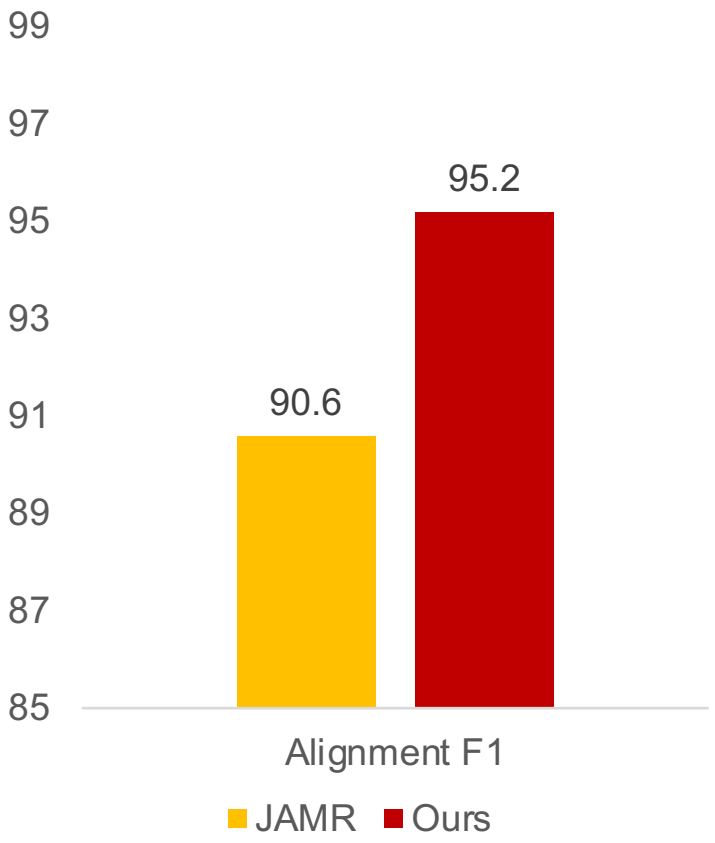
Ends up with a graph with Smatch of

0.91

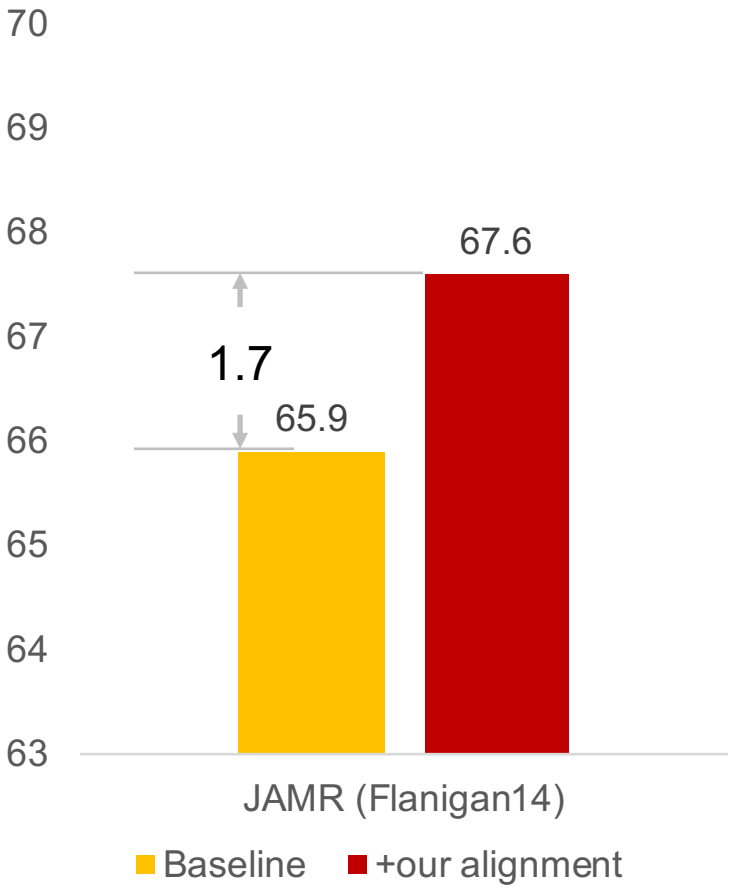
Input sentence: North Korean froze its nuclear actions in exchange for two nuclear reactors

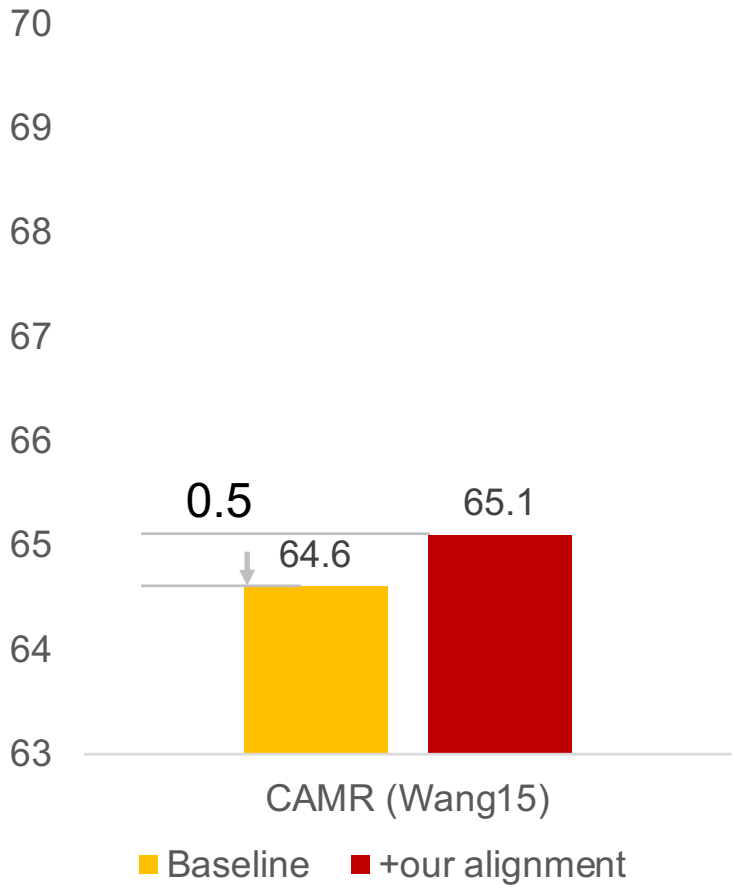
Alignment: North Korean – `country entity: "North Korean"`; froze – `freeze`; actions – `act`; exchange – `exchange`; for – `receive`; two – 2; nuclear^2 – `nucleus~2`; reactors – `reactor`

Aligner Experiments (intrinsic): Alignment Evaluation



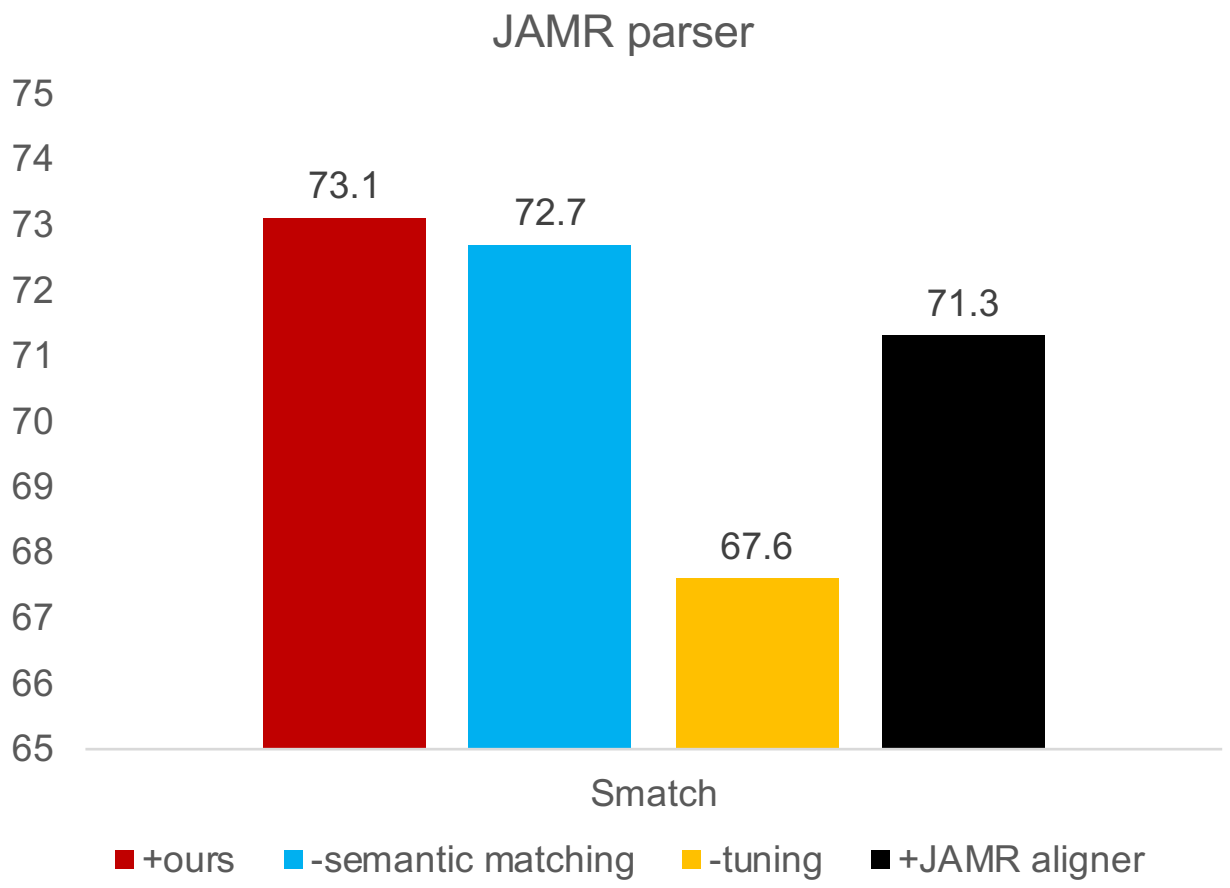
Aligner Experiments (extrinsic): Two Open-sourced AMR Parsers





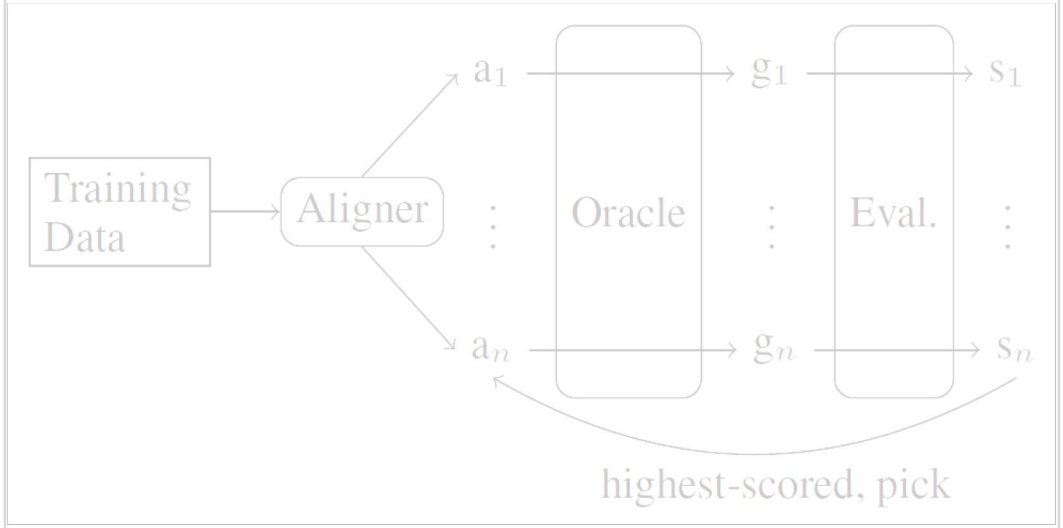
On LDC2014T12

Aligner Experiments: Ablations



Our Contributions

A novel aligner



A transition-based parser

Transition	Current State	Resulting State	Description
DROP	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma \mid s_0, \delta, \beta, A]$	pops out the word that doesn't convey any semantics (e.g., function words and punctuations).
MERGE	$[\sigma \mid s_0, \delta, b_0 \mid b_1 \mid \beta, A]$	$[\sigma \mid s_0, \delta, b_0\_b_1 \mid \beta, A]$	concatenates a sequence of words into a span, which can be derived as a named entity (`name`) or `date-entity`.
CONFIRM(c)	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma \mid s_0, \delta, c \mid \beta, A]$	derives the first element of the buffer (a word or span) into a concept c.
ENTITY(c)	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma \mid s_0, \delta, c \mid \beta, A \cup \text{relations}(c)]$	a special form of CONFIRM that derives the first element into an entity and builds the internal entity AMR fragment.
NEW(c)	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma \mid s_0, \delta, c \mid b_0 \mid \beta, A]$	generates a new concept c and pushes it to the front of the buffer.
LEFT(r)	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A \cup \{s_0 \xleftarrow{r} b_0\}]$	links a relation r between the top concepts on the stack and the buffer.
RIGHT(r)	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A \cup \{s_0 \xrightarrow{r} b_0\}]$	
CACHE	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma, s_0 \mid \delta, b_0 \mid \beta, A]$	passes the top concept of the stack onto the deque.
SHIFT	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma \mid s_0 \mid \delta \mid b_0, [], \beta, A]$	shifts the first concept of the buffer onto the stack along with those on the deque.
REDUCE	$[\sigma \mid s_0, \delta, b_0 \mid \beta, A]$	$[\sigma, \delta, b_0 \mid \beta, A]$	pops the top concept of the stack.

$$p(a|s) = \frac{\exp\{g_a \cdot \textsc{StackLSTM}(s) + b_a\}}{\sum_{a'} \exp\{g_{a'} \cdot \textsc{StackLSTM}(s) + b_{a'}\}}$$

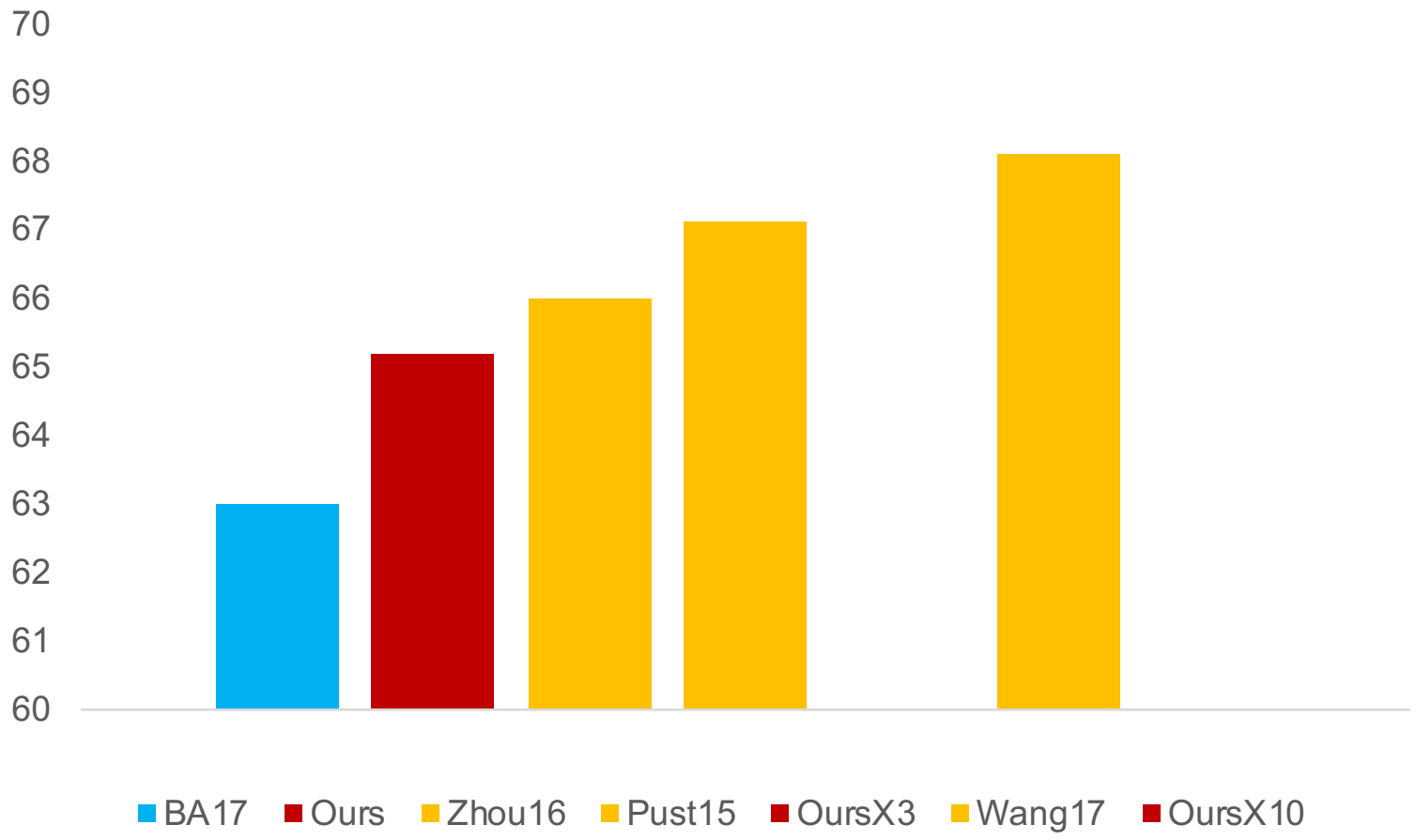
Transition-based AMR Parser on Our Transition System

- Use StackLSTM [Dyer et al. 2015] to represent state

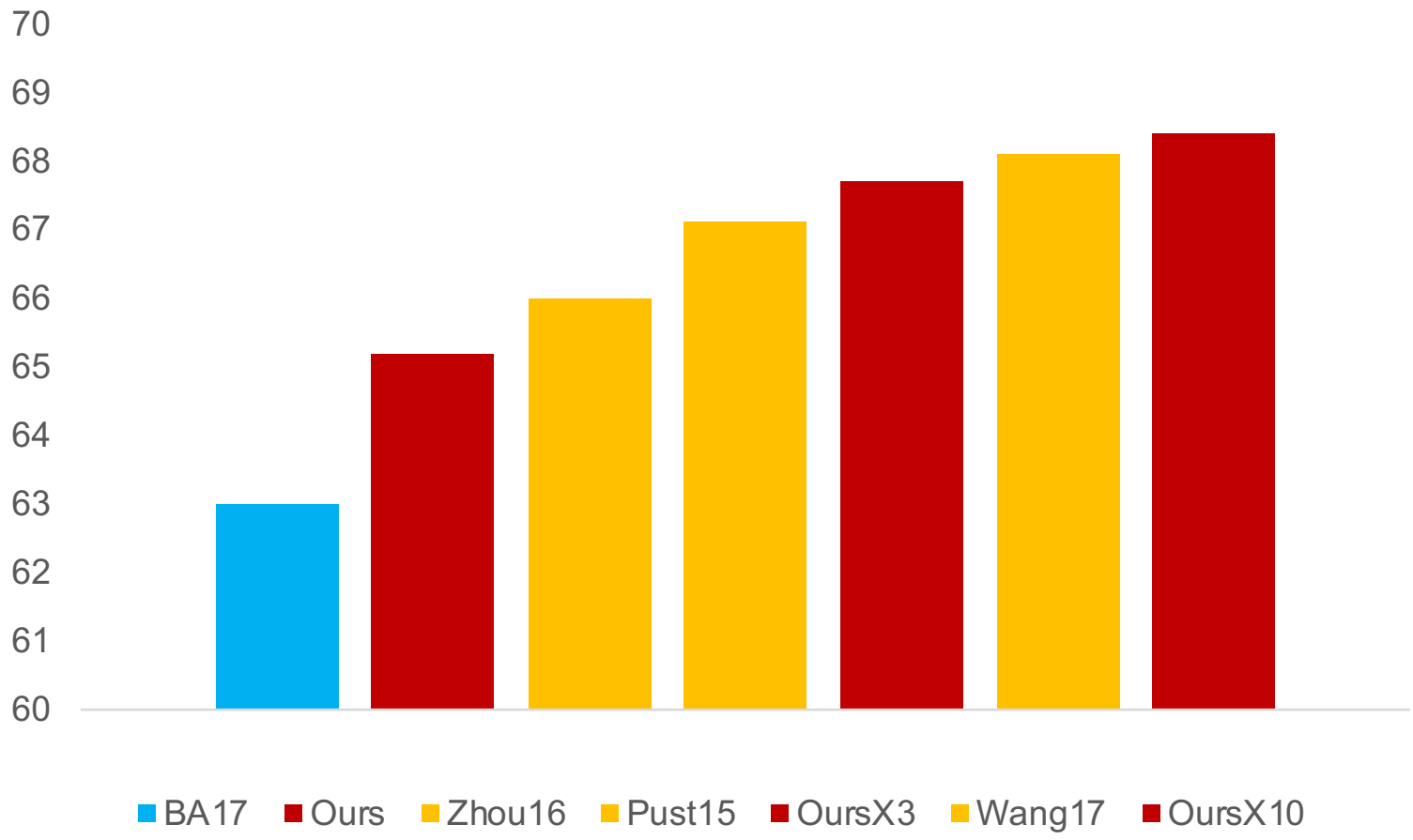
$$p(a|s) = \frac{\exp\{g_a \cdot \textsc{StackLSTM}(s) + b_a\}}{\sum_{a'} \exp\{g_{a'} \cdot \textsc{StackLSTM}(s) + b_{a'}\}}$$

- Call a unique subroutine for span-to-concept classification

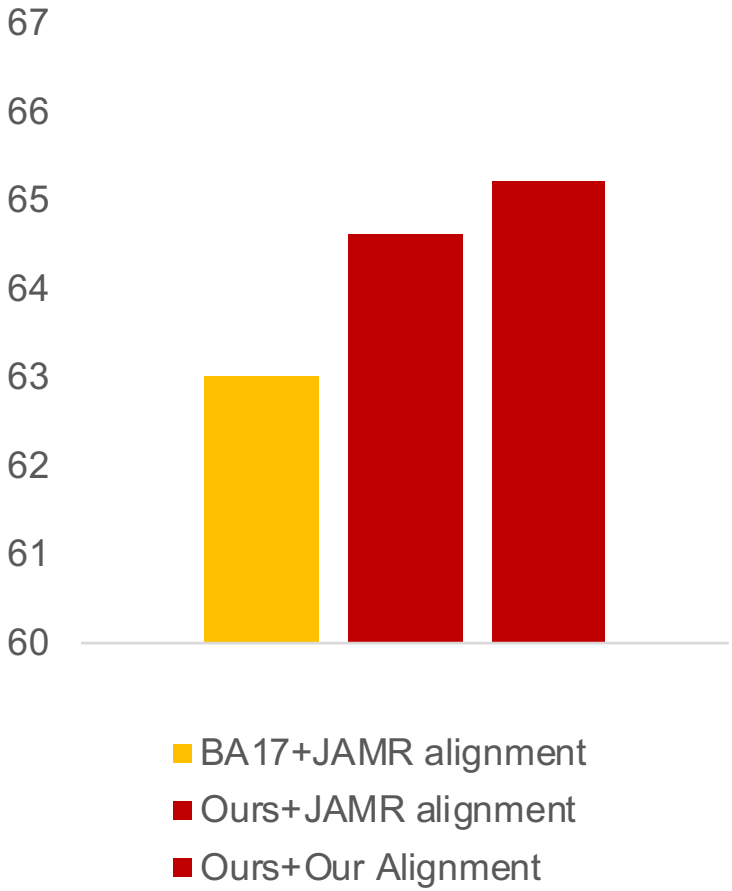
Parser Experiments

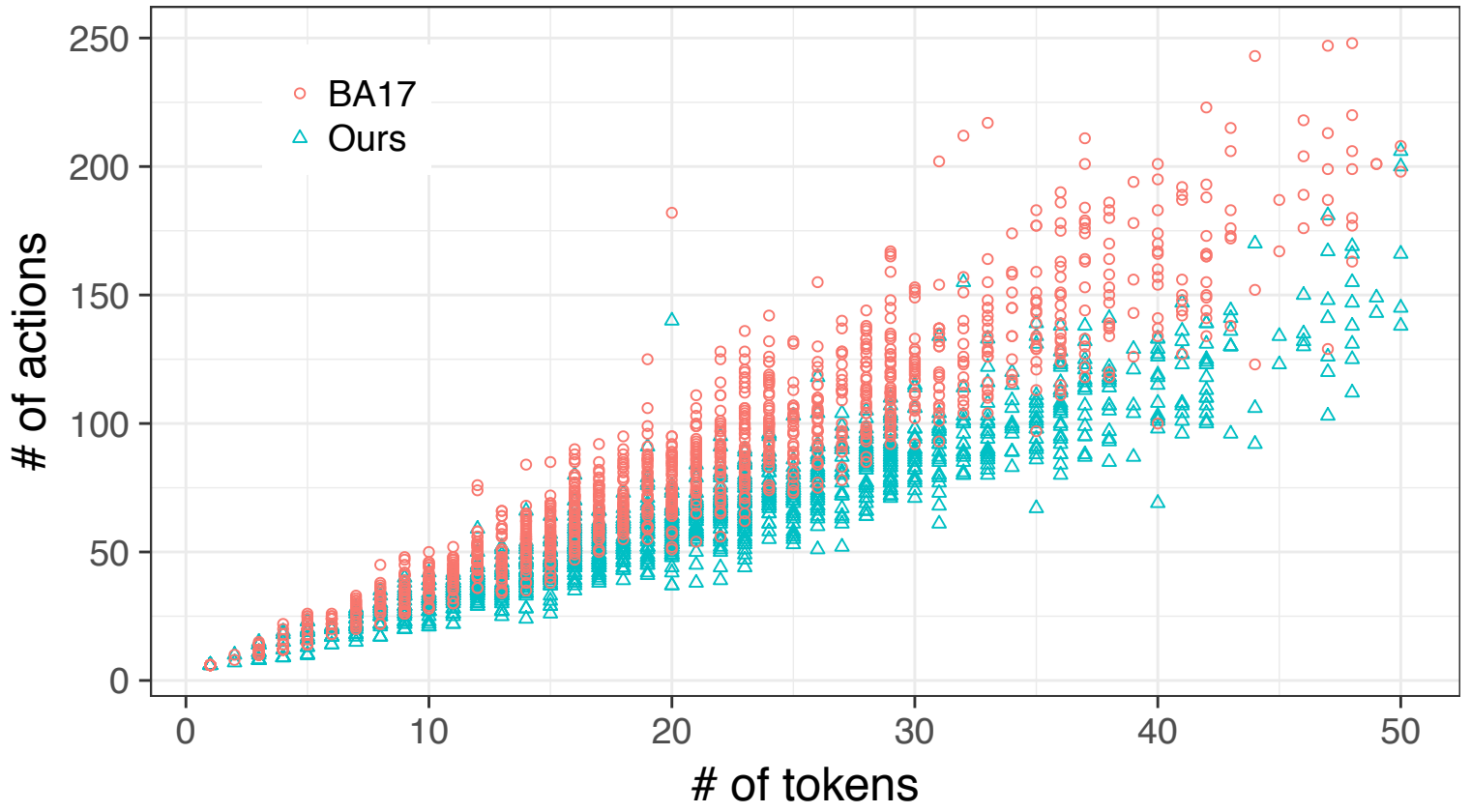


Parser Experiments



Comparison to Ballesteros and Al-Onaizan (2017)





Parser Experiments: Speed Comparison

Model	Tokens/s
JAMR	7
CAMR	24
Wang17 (Extension of CAMR)	<24
Our (X10)	**43**

Conclusion

- A new AMR aligner that improves two open-sourced parsers
- A transition-based parser that is accurate and fast
- Code and generated alignments (LDC2014T12 and LDC2017T10) are available at: https://github.com/Oneplus/tamr

HIT-SCIR

Thanks and Q/A