Universal Meaning Representation Parsing

Daniel Hershcovich
Joint work with Omri Abend and Ari Rappoport

Seminar in Computational Linguistics
Uppsala University
November 29, 2019

What can we teach computers to do with language?

What can we teach computers to do with language?

Translate:

דניאל עבר לקופנהגן אחרי שסיים את הלימודים

After graduation, Daniel moved to Copenhagen

What can we teach computers to do with language?

Recognize entities:

After graduation, Daniel moved to Copenhagen

Daniel $\downarrow$ Person

Copenhagen $\downarrow$ Location

What can we teach computers to do with language?

Infer: $\boxed{\text{After graduation, Daniel moved to Copenhagen}}$

$$\downarrow$$

$$\boxed{\text{Daniel graduated.}}$$

What can we teach computers to do with language?

Simplify:

After graduation, Daniel moved to Copenhagen

Daniel graduated. Then Daniel moved to Copenhagen.

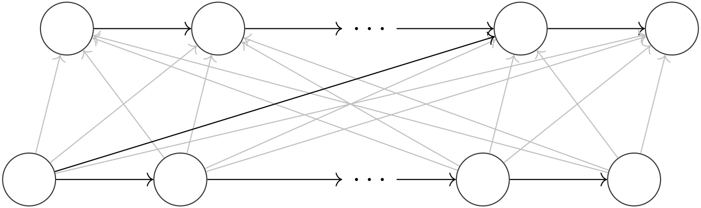
What can we teach computers to do with language?

דניאל עבר לקופנהגן אחרי שסיים את הלימודים

After graduation, Daniel moved to Copenhagen

Daniel graduated. Then Daniel moved to Copenhagen.

Neural models require the right *inductive bias*.



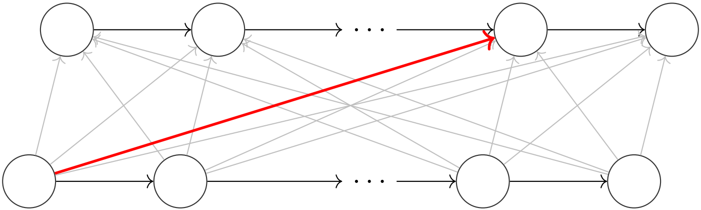
What can we teach computers to do with language?

דניאל עבר לקופנהגן אחרי שסיים את הלימודים

After graduation, Daniel moved to Copenhagen

Daniel graduated. Then Daniel moved to Copenhagen.

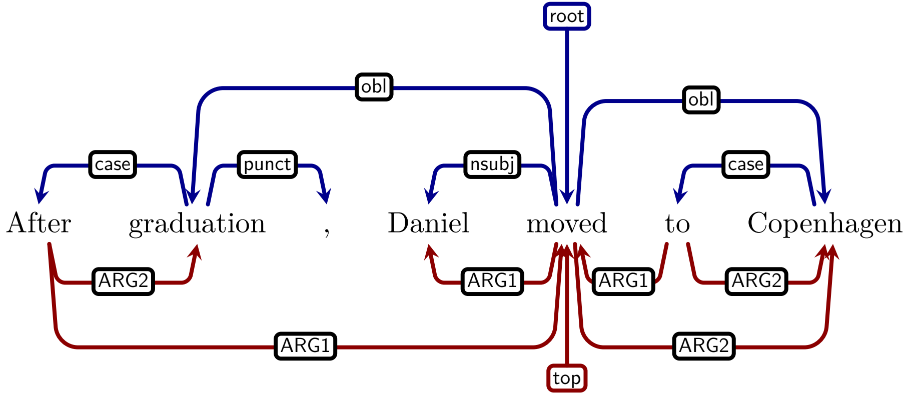
Neural models require the right *inductive bias*.



Symbolic Structure Representation

Relations between words or concepts.

Example: syntactic (UD)/semantic (DM) bi-lexical dependencies.



Meaning Representation

Abstract away from detail that does not affect meaning:

$\boxed{\text{graduation}} \approx \boxed{\text{graduated}} \approx$ סיים את הלימודים $\approx \boxed{\text{Abschluss}}$

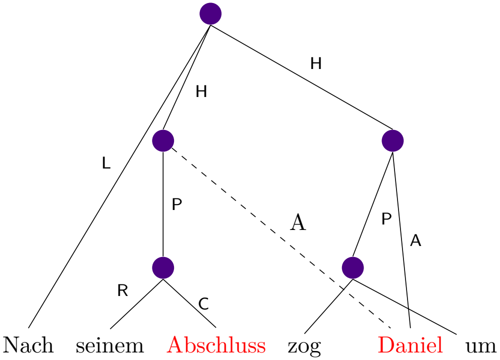
Meaning Representation

Abstract away from detail that does not affect meaning:

graduation ≈ graduated ≈ סיים את הלימודים ≈ Abschluss

But capture useful distinctions, such as:

- Scenes and participants
- Scene linkage
- Multi-word chunking



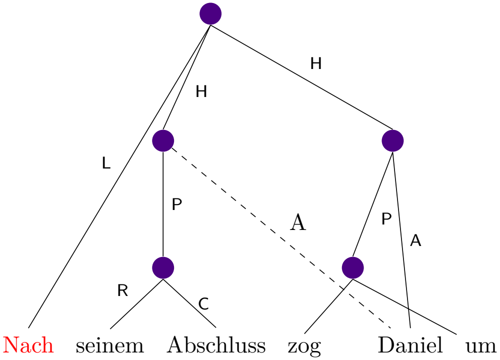
Meaning Representation

Abstract away from detail that does not affect meaning:

graduation ≈ graduated ≈ סיים את הלימודים ≈ Abschluss

But capture useful distinctions, such as:

- Scenes and participants
- Scene linkage
- Multi-word chunking



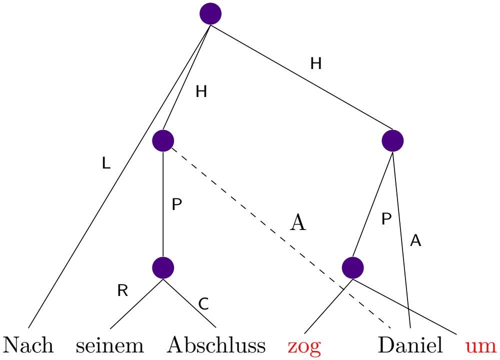
Meaning Representation

Abstract away from detail that does not affect meaning:

graduation ≈ graduated ≈ סיים את הלימודים ≈ Abschluss

But capture useful distinctions, such as:

- Scenes and participants
- Scene linkage
- Multi-word chunking

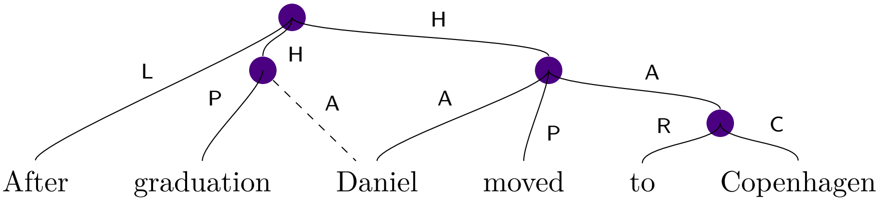


Outline

1. UCCA
2. Cross-lingual Parsing
3. Cross-framework Parsing
4. What Distinguishes Meaning Representations?

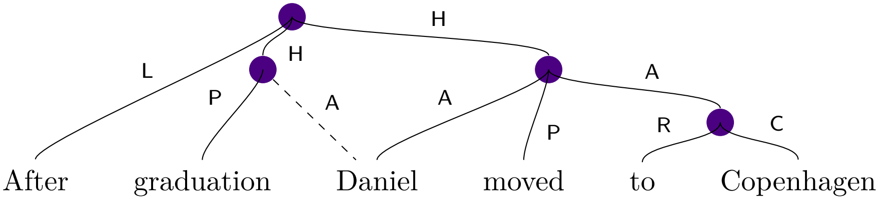
Universal Conceptual Cognitive Annotation (UCCA)

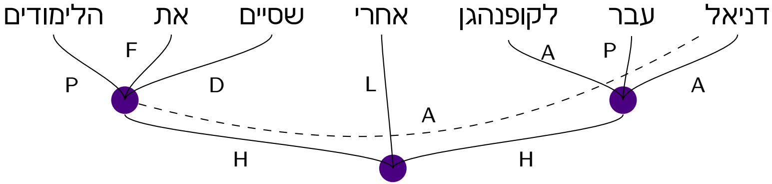
Supports rapid and intuitive annotation of linguistic semantic phenomena. [Abend and Rappoport, 2013]



Universal Conceptual Cognitive Annotation (UCCA)

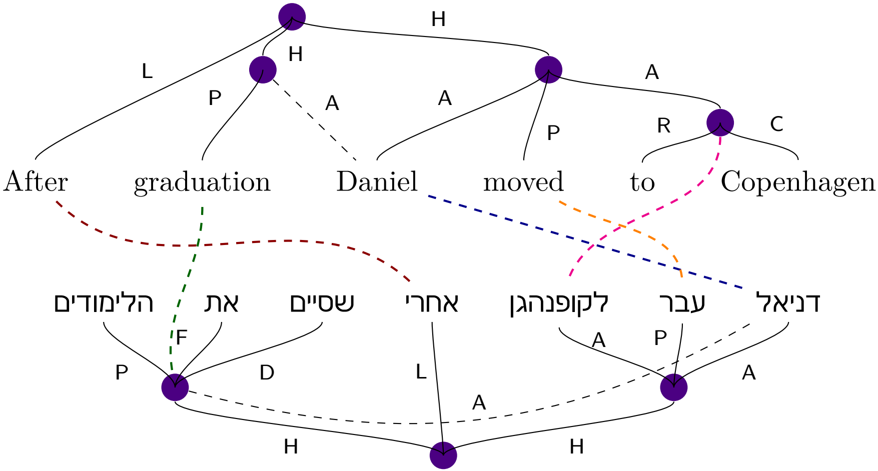
Supports rapid and intuitive annotation of linguistic semantic phenomena.
Cross-linguistically applicable and stable [Sulem et al., 2015].





Universal Conceptual Cognitive Annotation (UCCA)

Supports rapid and intuitive annotation of linguistic semantic phenomena.
Cross-linguistically applicable and stable [Sulem et al., 2015].



UCCA Applications

Semantics-based **evaluation** of

- Machine translation [Birch et al., 2016].
- Text simplification [Sulem et al., 2018a].
- Grammatical error correction [Choshen and Abend, 2018].

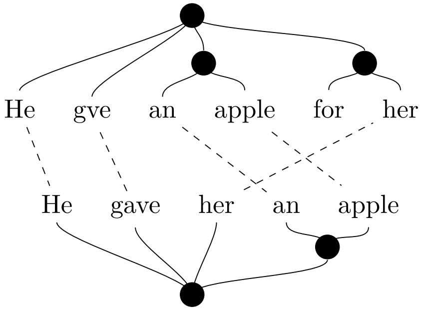
Sentence splitting for text simplification [Sulem et al., 2018b].

UCCA Applications

Semantics-based **evaluation** of

- Machine translation [Birch et al., 2016].
- Text simplification [Sulem et al., 2018a].
- Grammatical error correction [Choshen and Abend, 2018].

Sentence splitting for text simplification [Sulem et al., 2018b].

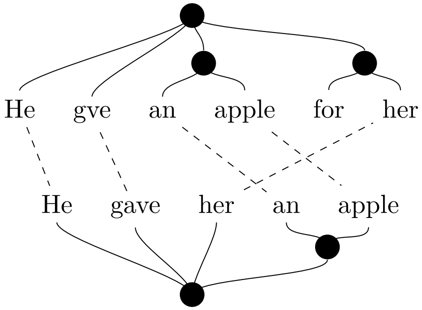


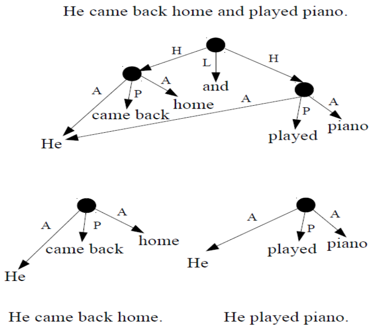
UCCA Applications

Semantics-based **evaluation** of

- Machine translation [Birch et al., 2016].
- Text simplification [Sulem et al., 2018a].
- Grammatical error correction [Choshen and Abend, 2018].

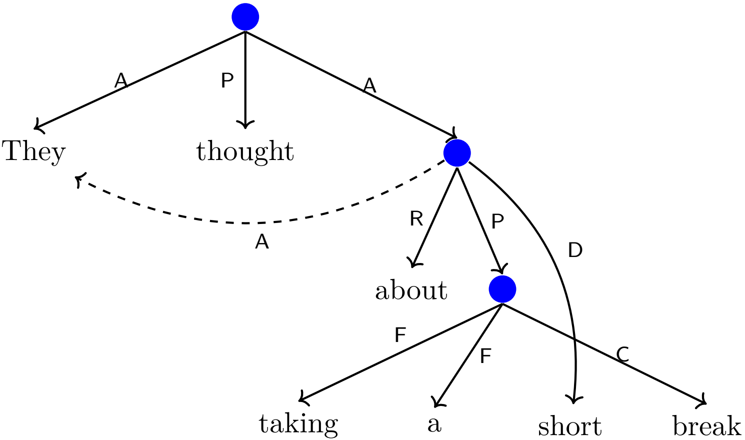
Sentence splitting for text simplification [Sulem et al., 2018b].





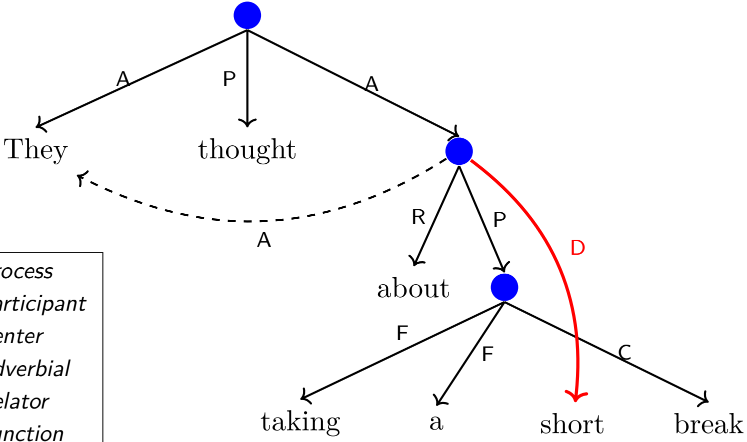
Graph Structure

UCCA structures are directed acyclic graphs (DAGs) with labeled edges.
Text tokens are terminals, complex units are non-terminal nodes.



Graph Structure

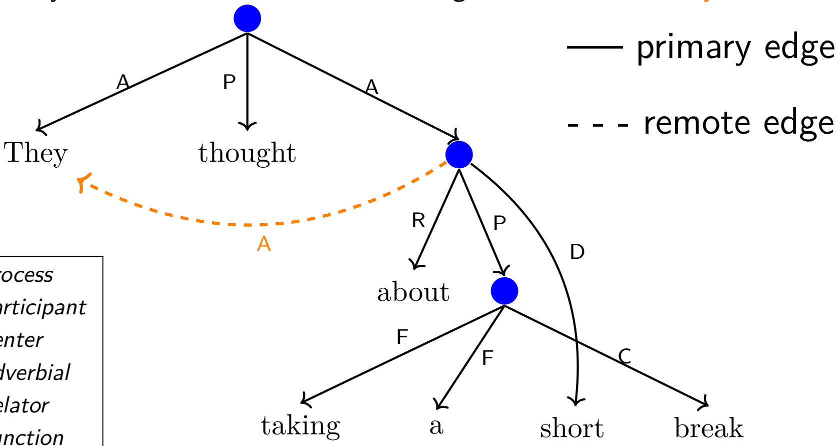
UCCA structures are directed acyclic graphs (DAGs) with labeled edges.
Text tokens are terminals, complex units are non-terminal nodes.
Phrases may be discontinuous.



P	*Process*
A	*Participant*
C	*Center*
D	*Adverbial*
R	*Relator*
F	*Function*

Graph Structure

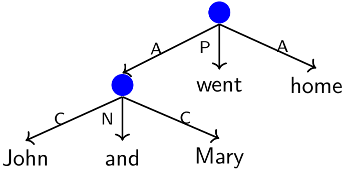
UCCA structures are directed acyclic graphs (DAGs) with labeled edges.
Text tokens are terminals, complex units are non-terminal nodes.
Phrases may be discontinuous. *Remote edges* enable reentrancy.



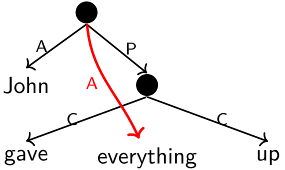
P	*Process*
A	*Participant*
C	*Center*
D	*Adverbial*
R	*Relator*
F	*Function*

Structural Properties

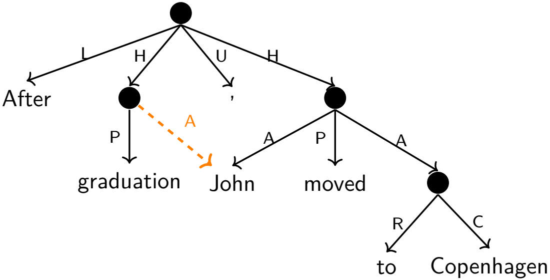
(1) non-terminal nodes



(2) discontinuity

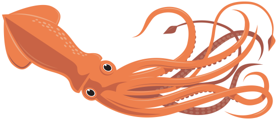


(3) reentrancy



UCCA Data

- English Wikipedia articles.
- English-French-German *Twenty Thousand Leagues Under the Sea*.
- English Web Treebank reviews.



Data Statistics

	Wiki	20K			EWT
	en	en	fr	de	en
# sentences	5,141	492	492	6,514	3,520
# tokens	158,739	12,638	13,021	144,529	51,042
# non-terminal nodes	62,002	4,699	5,110	51,934	18,156
% discontinuous	1.71	3.19	4.64	8.87	3.87
% reentrant	1.84	0.89	0.65	0.31	0.83
# edges	208,937	16,803	17,520	187,533	60,739
% primary	97.40	96.79	97.02	97.32	97.32
% remote	2.60	3.21	2.98	2.68	2.68

TUPA: Transition-based UCCA Parser

Parses text $w_1 \dots w_n$ to graph G incrementally by applying transitions to the parser state, consisting of: stack, buffer and constructed graph [Hershcovich, Abend, and Rappoport, 2017].

TUPA: Transition-based UCCA Parser

Parses text $w_1 \ldots w_n$ to graph G incrementally by applying transitions to the parser state, consisting of: stack, buffer and constructed graph [Hershcovich, Abend, and Rappoport, 2017].

Initial state:

stack | | | | | | | | buffer

stack	buffer						
●	They	thought	about	taking	a	short	break

TUPA: Transition-based UCCA Parser

Parses text $w_1 \dots w_n$ to graph G incrementally by applying transitions to the parser state, consisting of: stack, buffer and constructed graph [Hershcovich, Abend, and Rappoport, 2017].

Initial state:

stack

●

buffer

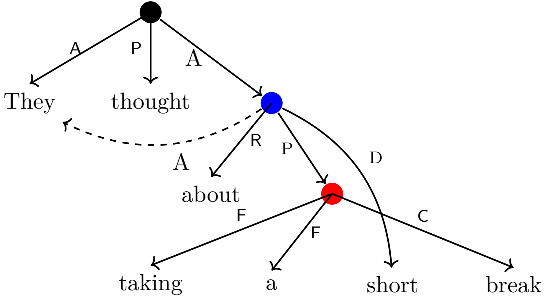
They	thought	about	taking	a	short	break

TUPA transitions:
{Shift, Reduce, Node$_X$, Left-Edge$_X$, Right-Edge$_X$,
Left-Remote$_X$, Right-Remote$_X$, Swap, Finish}

These transitions enable non-terminal nodes, reentrancy and discontinuity.

Training

An *oracle* provides the transition sequence given the correct graph:



$\Downarrow$

Shift, Right-Edge$_A$, Shift, Swap, Right-Edge$_P$, Reduce, Shift, Shift, Node$_R$, Reduce, Left-Remote$_A$, Shift, Shift, Node$_C$, Reduce, Shift, Right-Edge$_P$, Shift, Right-Edge$_F$, Reduce, Shift, Swap, Right-Edge$_D$, Reduce, Swap, Right-Edge$_A$, Reduce, Reduce, Shift, Reduce, Shift, Right-Edge$_C$, Finish

Example: TUPA Transition Sequence

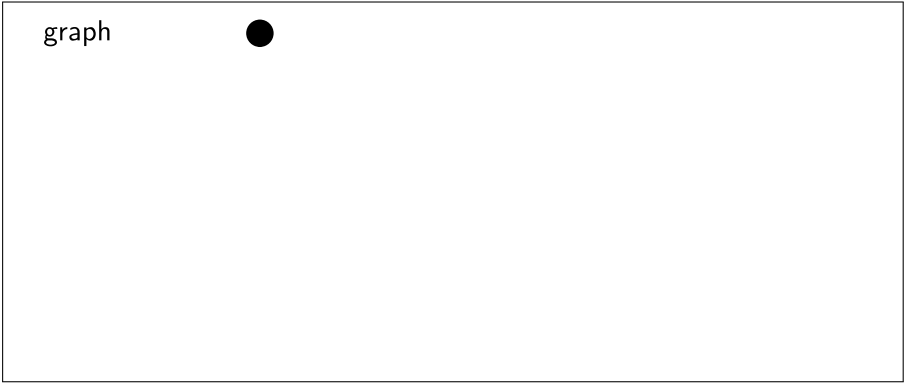
$\Rightarrow$ SHIFT

stack

●	They

buffer

thought	about	taking	a	short	break



Example: TUPA Transition Sequence

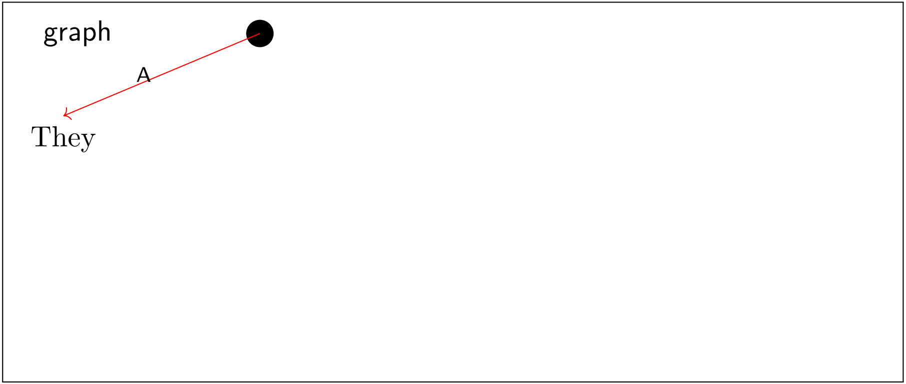
$\Rightarrow$ Right-Edge$_A$

stack

●	They

buffer

thought	about	taking	a	short	break



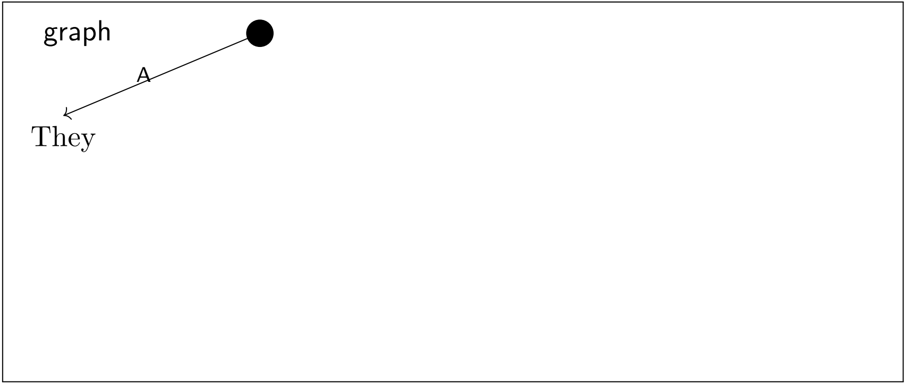
Example: TUPA Transition Sequence

⇒ Shift

stack | | | buffer

●	They	thought

about	taking	a	short	break



Example: TUPA Transition Sequence

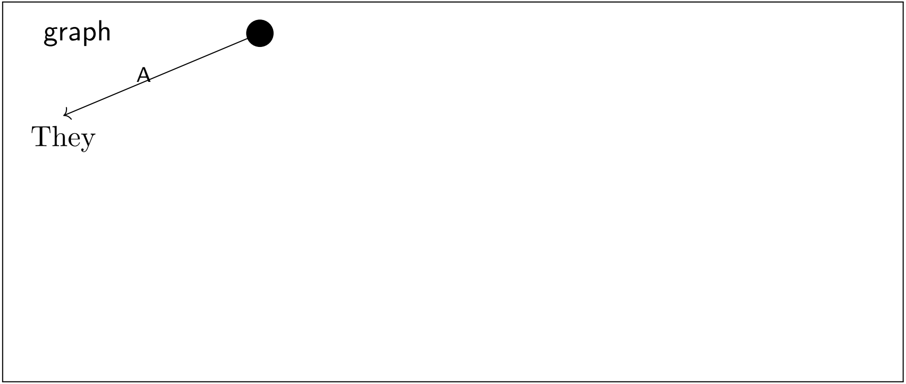
$\Rightarrow$ SWAP

stack

●	thought

buffer

They	about	taking	a	short	break



Example: TUPA Transition Sequence

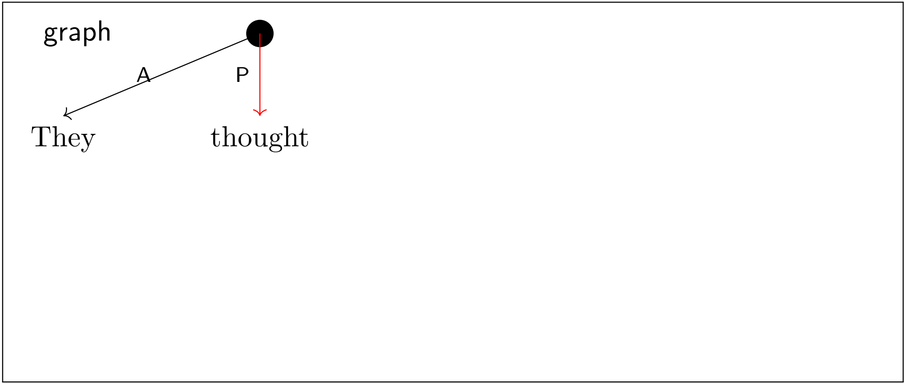
⇒ Right-Edge$_P$

stack

●	thought

buffer

They	about	taking	a	short	break



Example: TUPA Transition Sequence

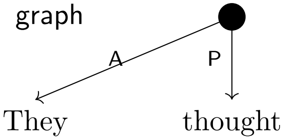
⇒ REDUCE

stack

They	about	taking	a	short	break

buffer

graph



Example: TUPA Transition Sequence

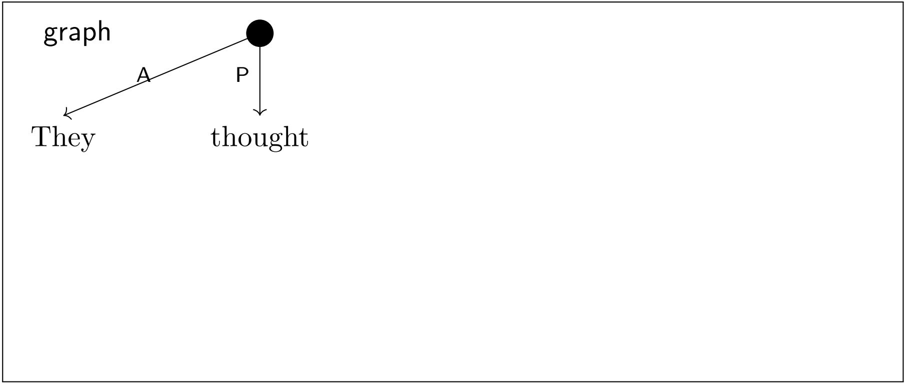
⇒ Shift

stack

	They

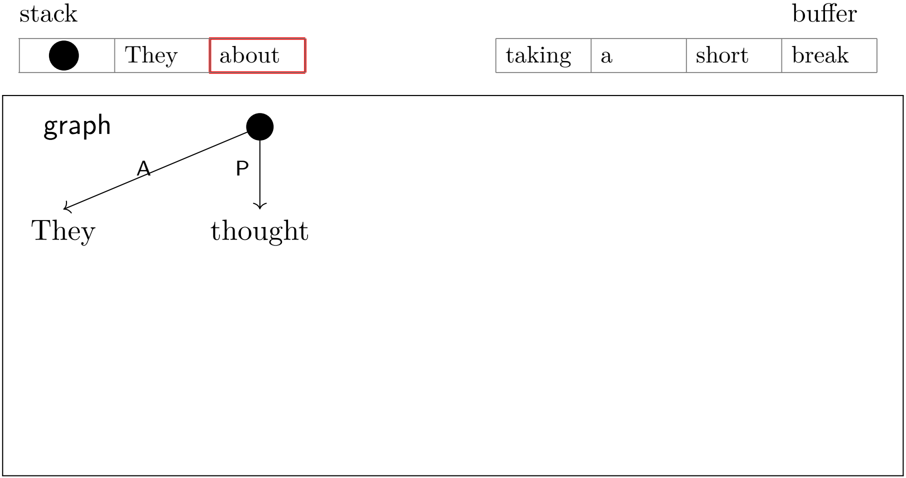
buffer

about	taking	a	short	break



Example: TUPA Transition Sequence

⇒ SHIFT

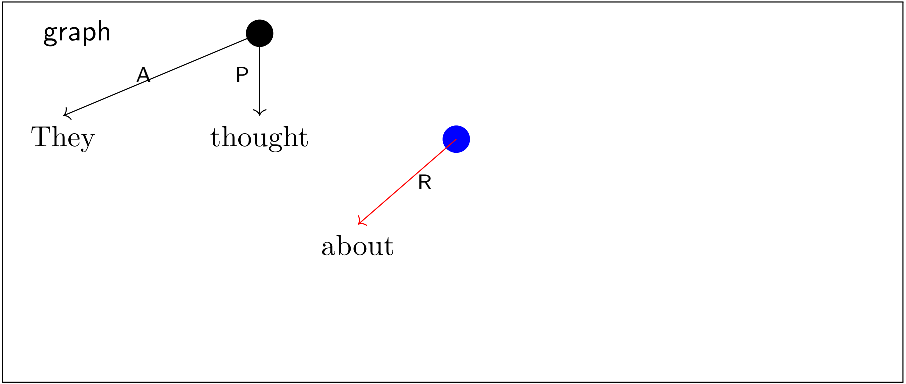


Example: TUPA Transition Sequence

$\Rightarrow$ NODE$_R$

stack		
●	They	about

				buffer
●	taking	a	short	break



Example: TUPA Transition Sequence

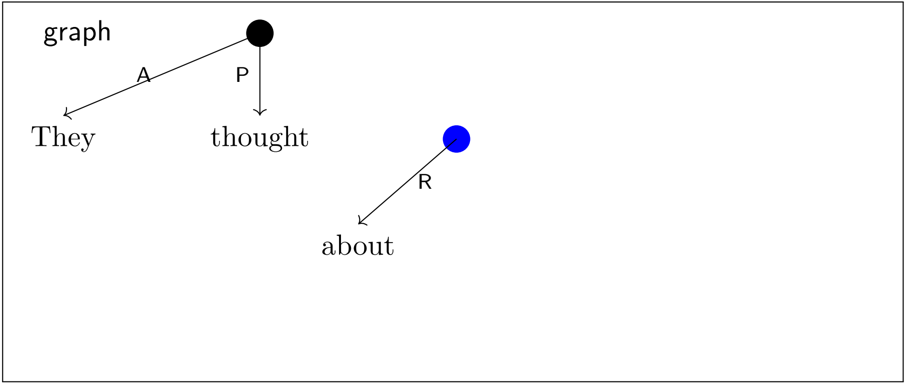
⇒ REDUCE

stack

●	They

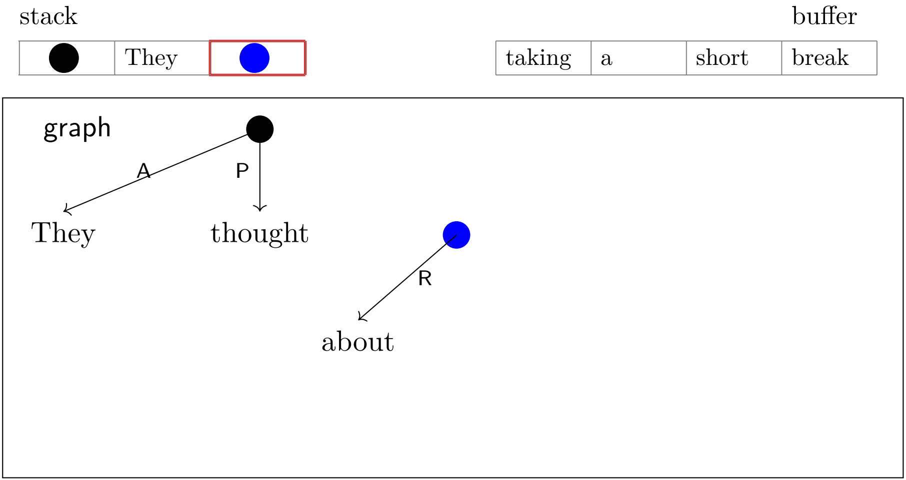
buffer

●	taking	a	short	break



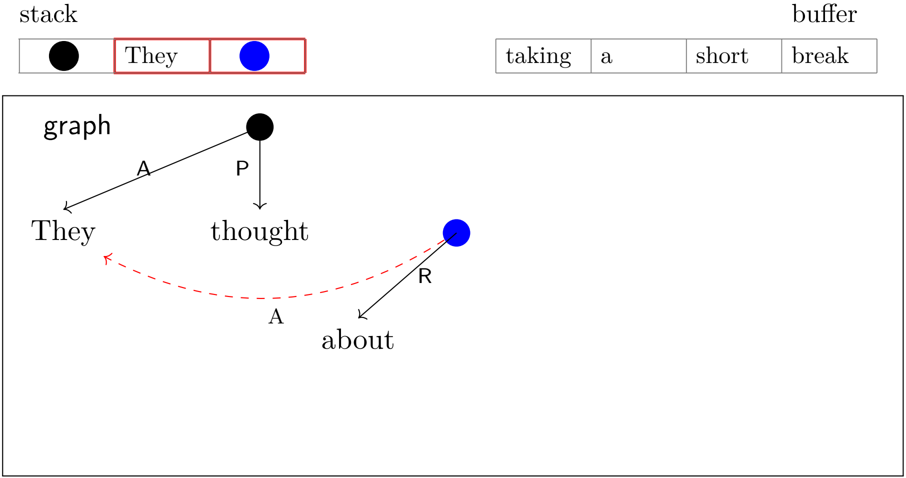
Example: TUPA Transition Sequence

⇒ Shift



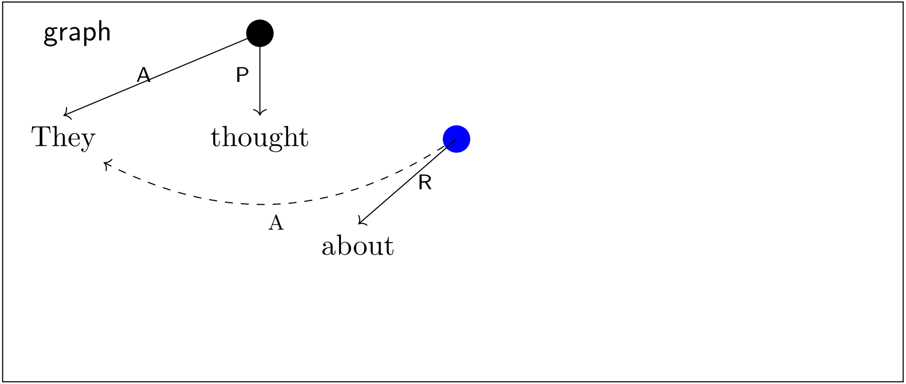
Example: TUPA Transition Sequence

⇒ Left-Remote$_A$



Example: TUPA Transition Sequence

⇒ Shift



Example: TUPA Transition Sequence

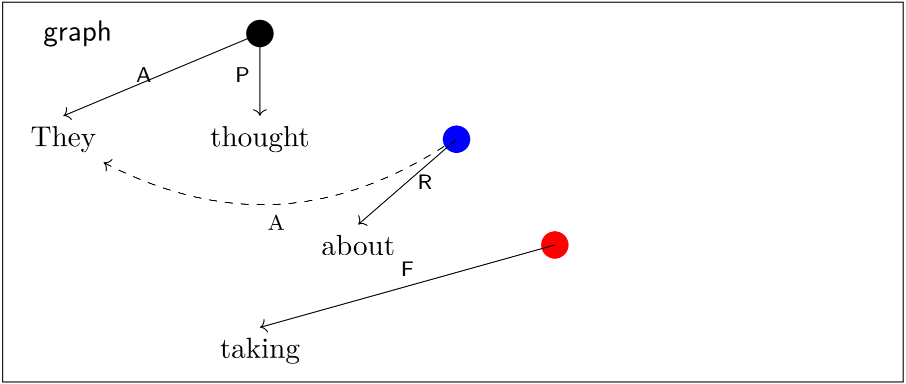
$\Rightarrow$ NODE$_C$

stack

	They		taking

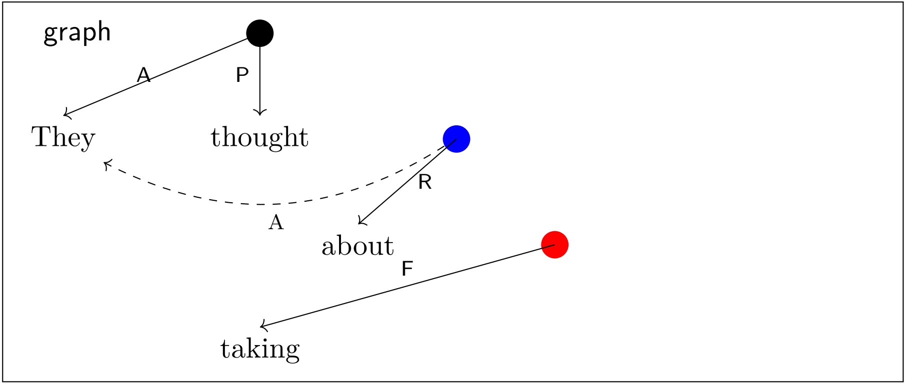
buffer

	a	short	break



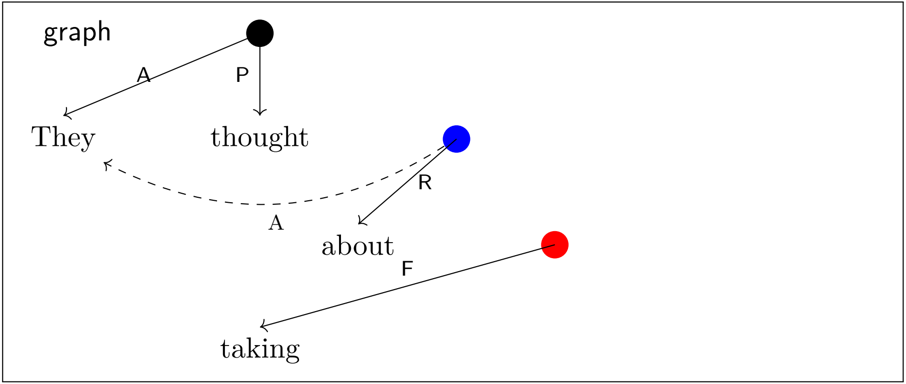
Example: TUPA Transition Sequence

⇒ Reduce



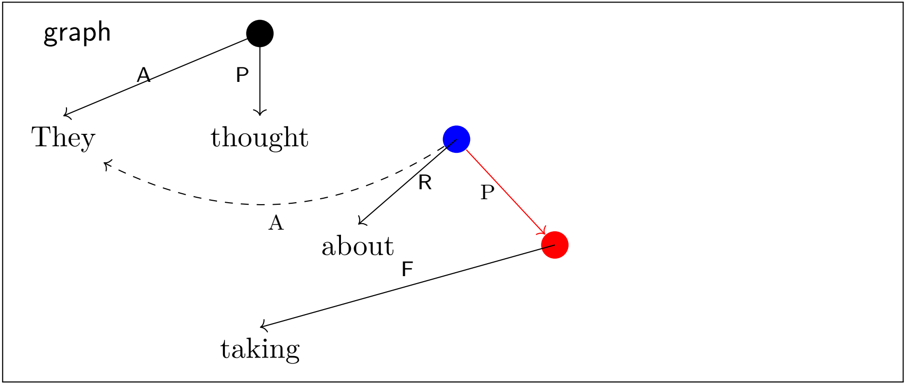
Example: TUPA Transition Sequence

⇒ Shift



Example: TUPA Transition Sequence

⇒ RIGHT-EDGE$_P$



Example: TUPA Transition Sequence

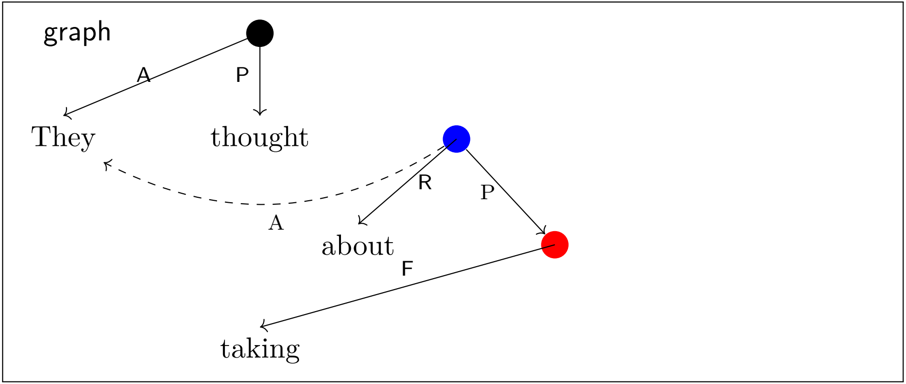
⇒ Shift

stack

●	They	●	●	a

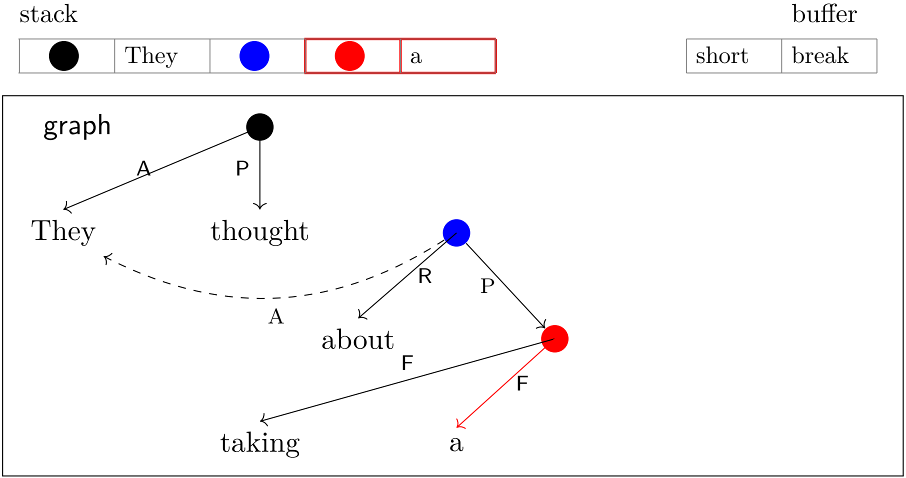
buffer

short	break



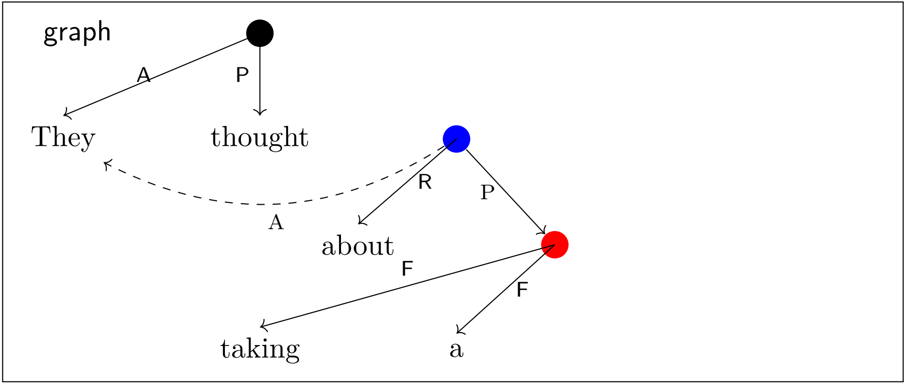
Example: TUPA Transition Sequence

⇒ RIGHT-EDGE$_F$



Example: TUPA Transition Sequence

⇒ Reduce



Example: TUPA Transition Sequence

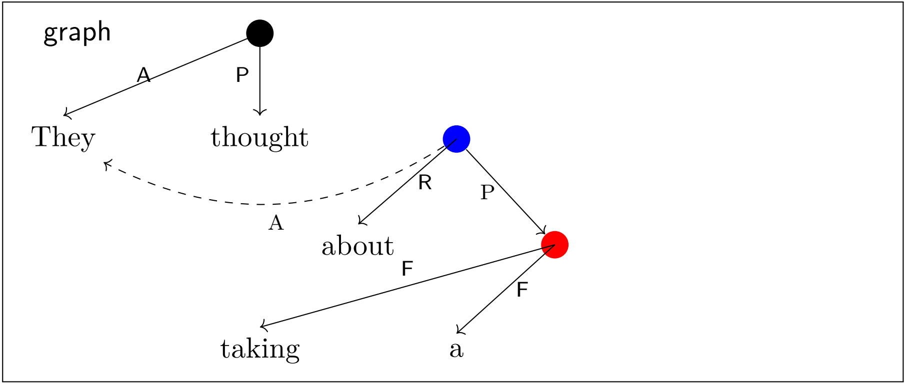
⇒ Shift

stack

●	They	●	●	short

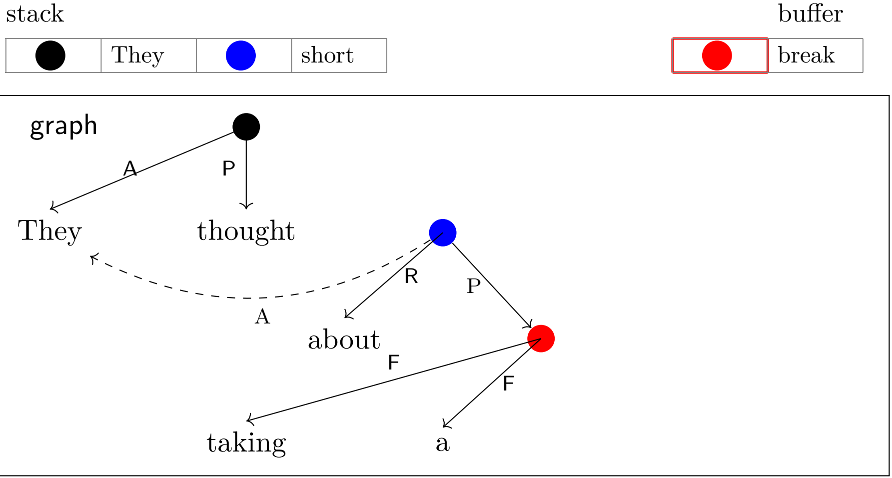
buffer

break



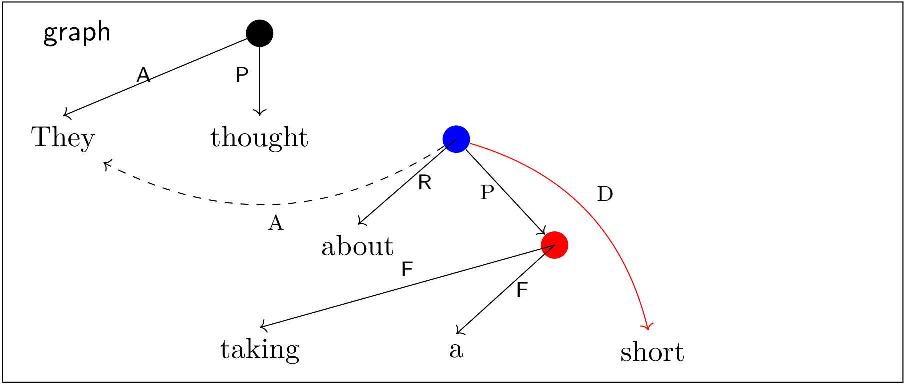
Example: TUPA Transition Sequence

⇒ Swap



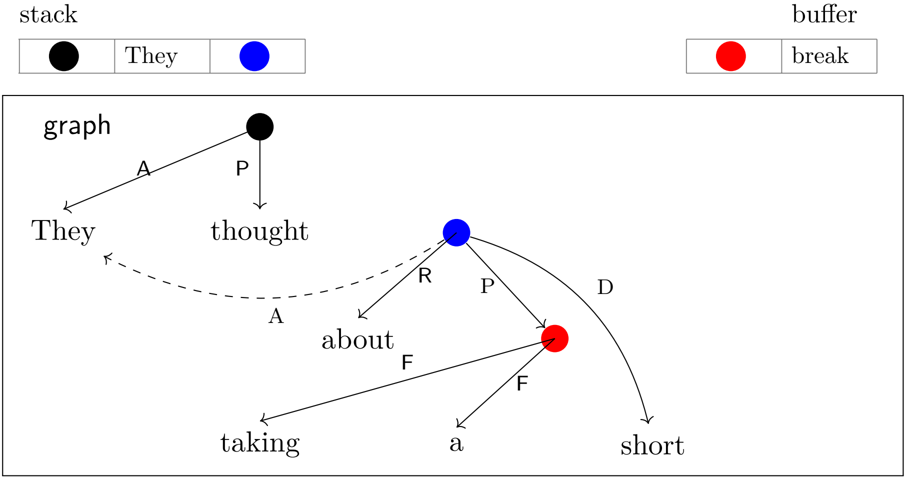
Example: TUPA Transition Sequence

$\Rightarrow$ RIGHT-EDGE$_D$



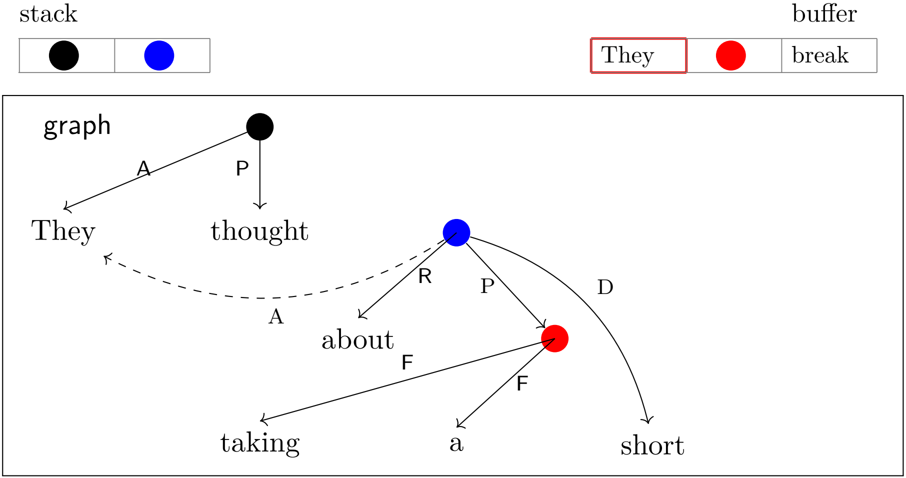
Example: TUPA Transition Sequence

⇒ Reduce



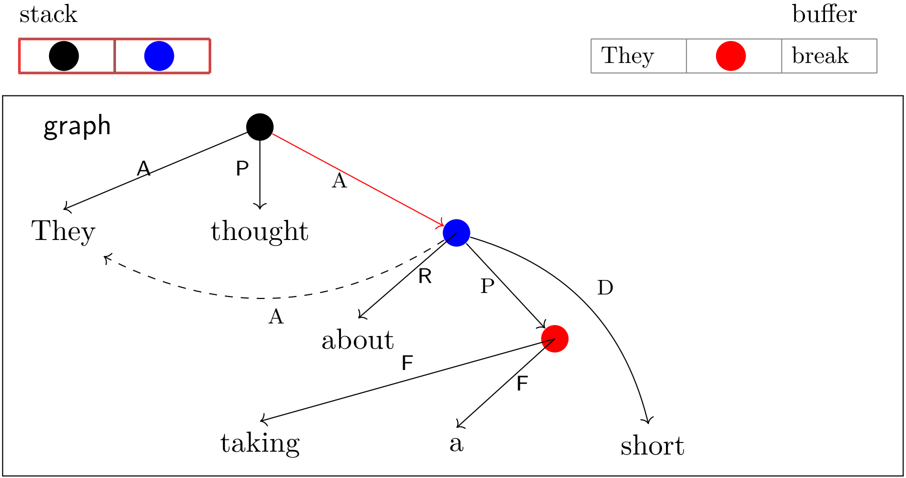
Example: TUPA Transition Sequence

⇒ Swap



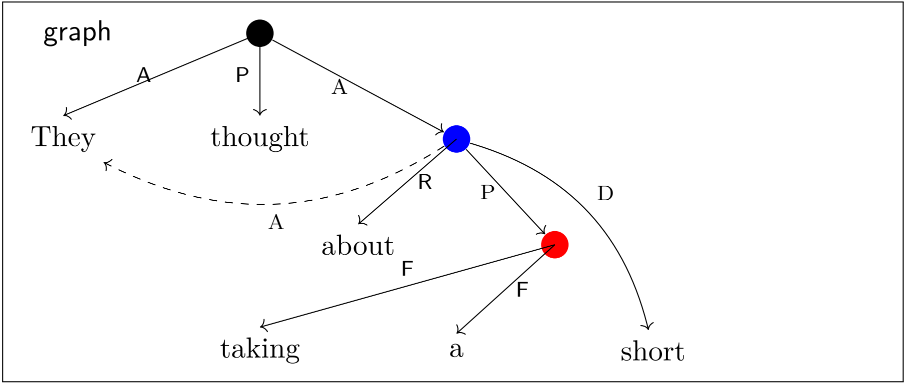
Example: TUPA Transition Sequence

⇒ RIGHT-EDGE$_A$



Example: TUPA Transition Sequence

⇒ REDUCE



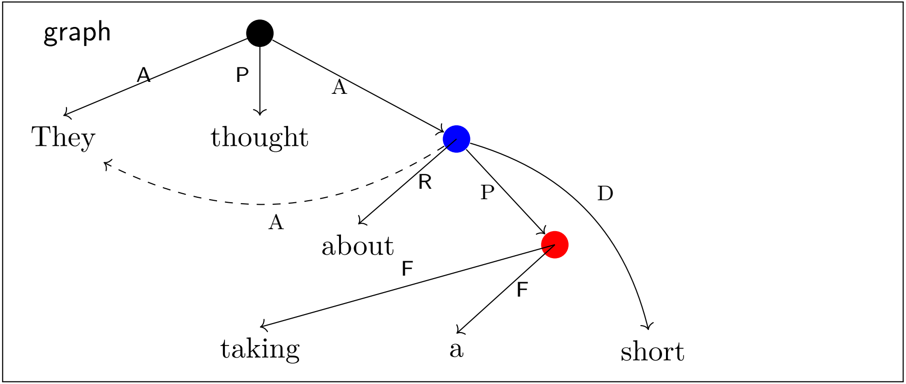
Example: TUPA Transition Sequence

⇒ REDUCE

stack

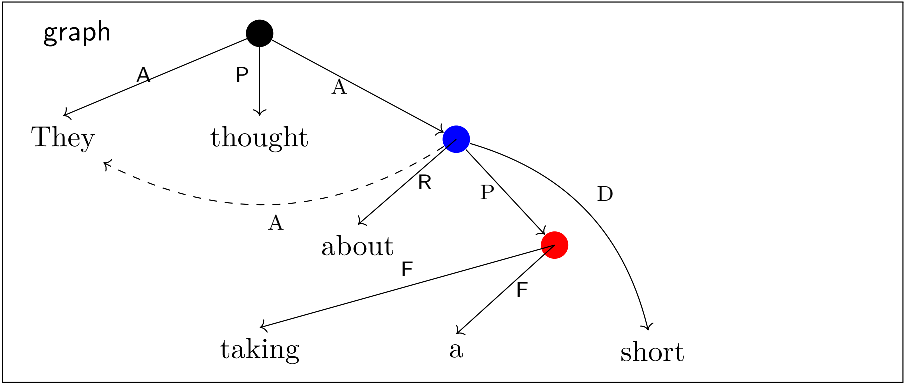
buffer

They		break



Example: TUPA Transition Sequence

⇒ SHIFT

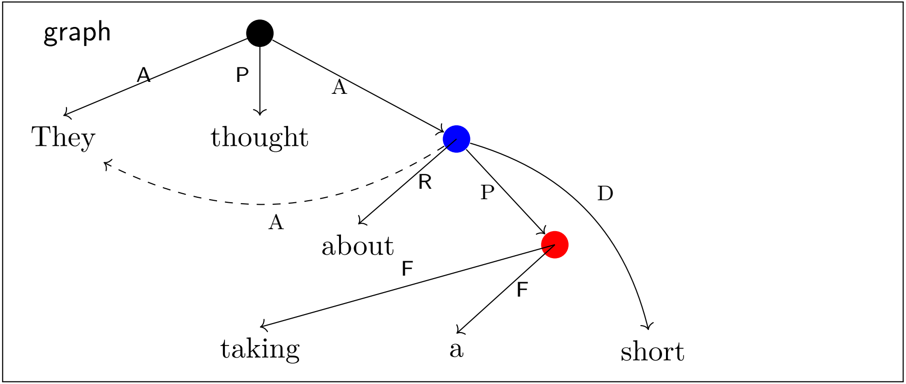


Example: TUPA Transition Sequence

⇒ Reduce

stack

buffer



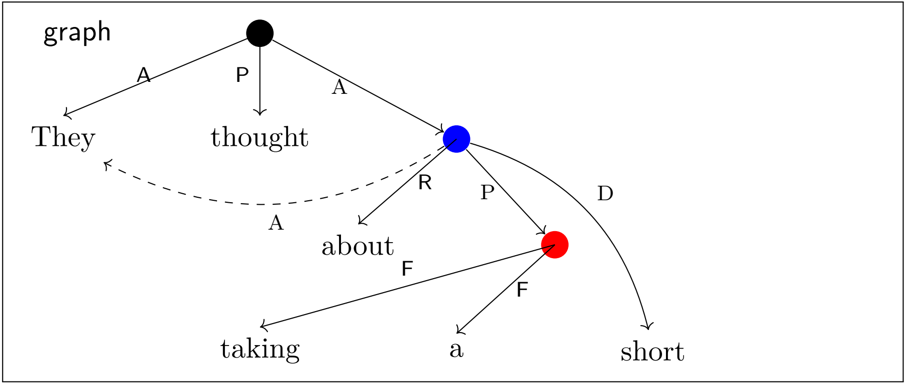
Example: TUPA Transition Sequence

$\Rightarrow$ SHIFT

stack

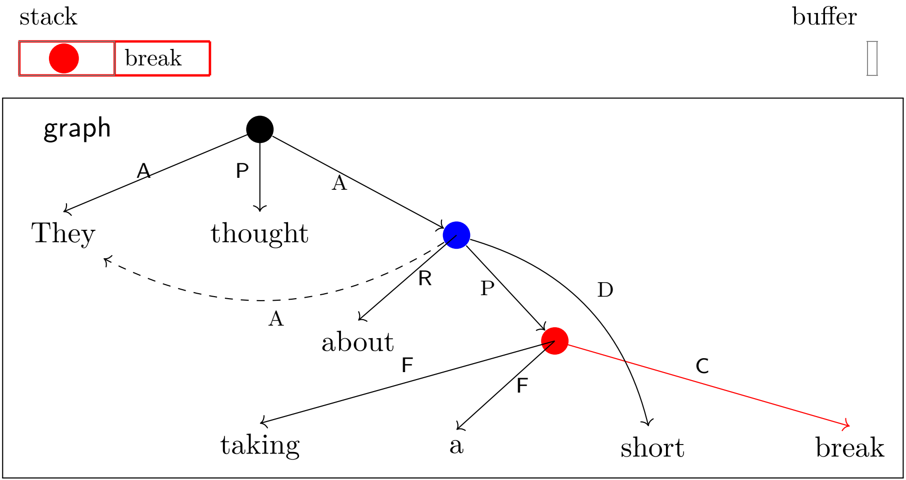
buffer

break



Example: TUPA Transition Sequence

⇒ RIGHT-EDGE$_C$



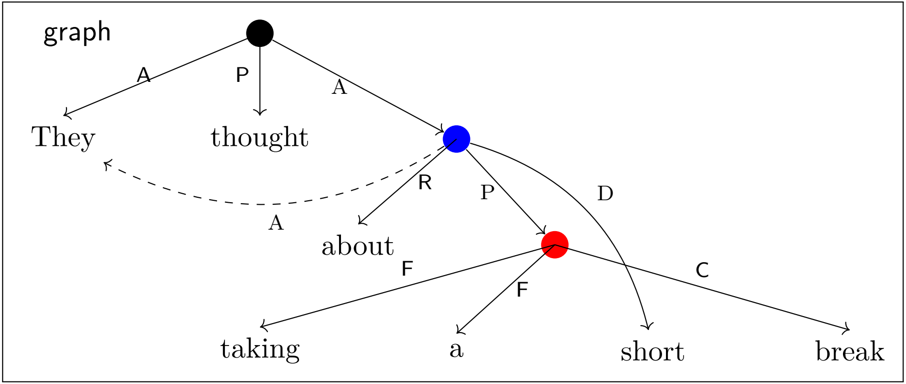
Example: TUPA Transition Sequence

⇒ FINISH

stack

●	break

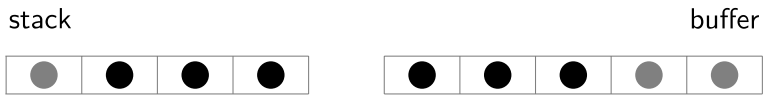
buffer



TUPA Model

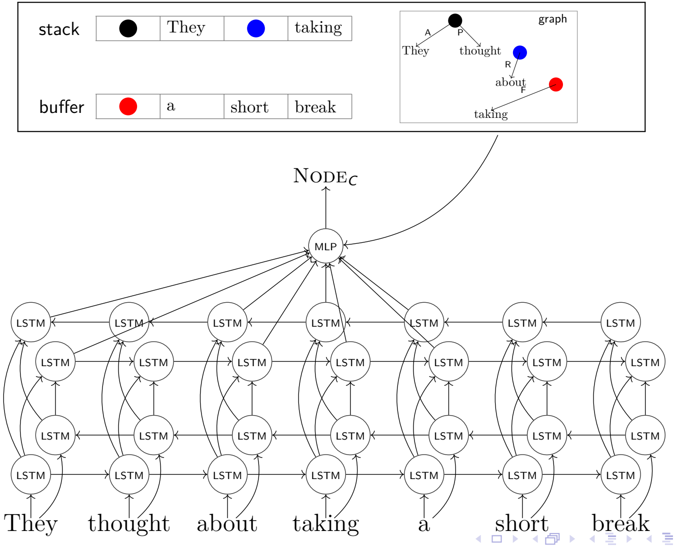
Learns to greedily predict transition based on current state.

Features include:
{words, parts of speech, syntactic dependencies, existing edge labels}
from the stack and buffer + parents, children, grandchildren.



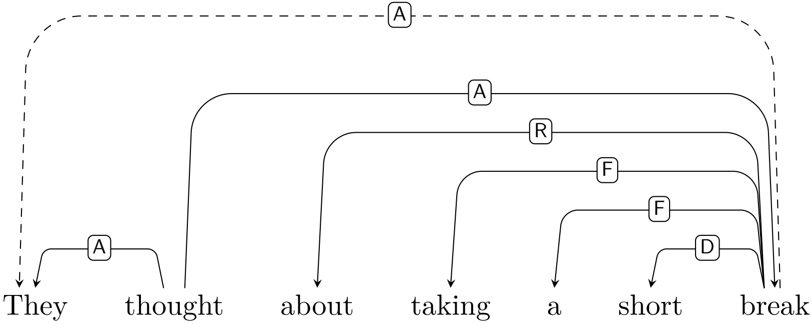
TUPA Model

Learns to greedily predict transition based on current state.



Comparing to Existing Methods

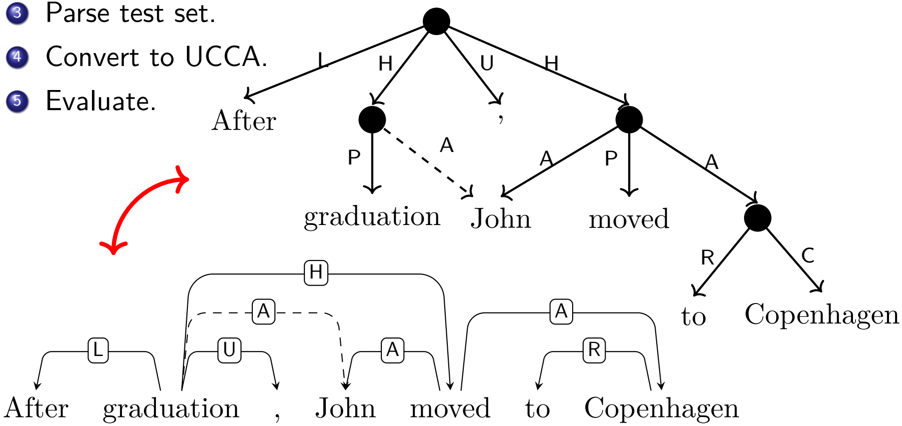
Using conversion-based approximation as baseline,
with bi-lexical DAG parsers and transition-based tree parsers.



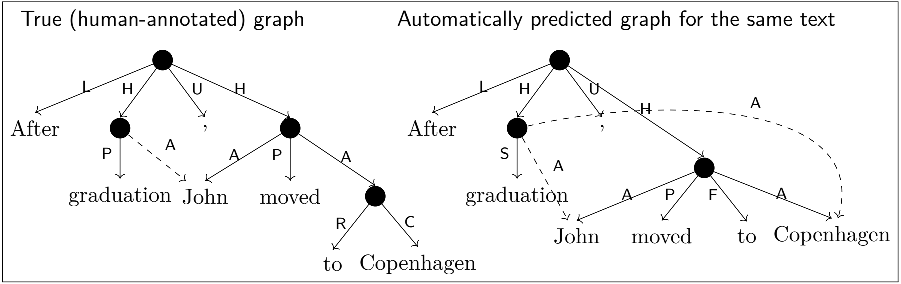
UCCA bi-lexical DAG approximation.

Bi-lexical Graph Approximation

1. Convert UCCA to bi-lexical DAGs.
2. Train bi-lexical parsers.
3. Parse test set.
4. Convert to UCCA.
5. Evaluate.

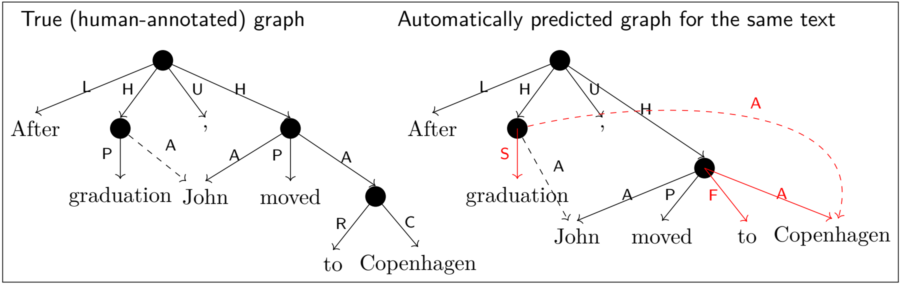


Evaluation



1. Match primary edges between the graphs by terminal yield and label.
2. Calculate **precision, recall and F1** scores.
3. Repeat for remote edges.

Evaluation



1. Match primary edges between the graphs by terminal yield and label.
2. Calculate **precision, recall and F1** scores.
3. Repeat for remote edges.

Primary

P	**R**	**F1**
$\frac{6}{9} = 67\%$	$\frac{6}{10} = 60\%$	64%

Remote

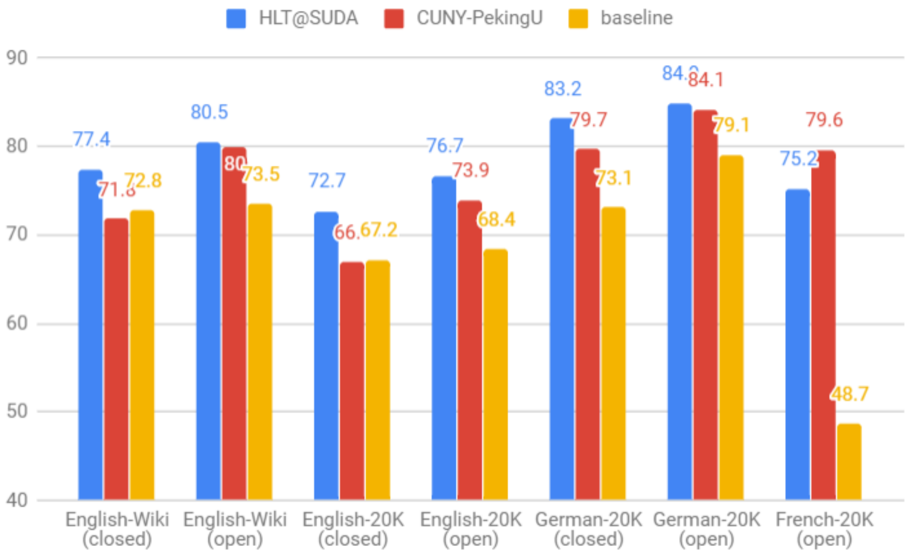
P	**R**	**F1**
$\frac{1}{2} = 50\%$	$\frac{1}{1} = 100\%$	67%

Outline

SemEval 2019: Cross-lingual UCCA Parsing

Shared task: parsing text to UCCA graphs [Hershcovich, Choshen, Sulem, Aizenbud, Rappoport, and Abend, 2019b].

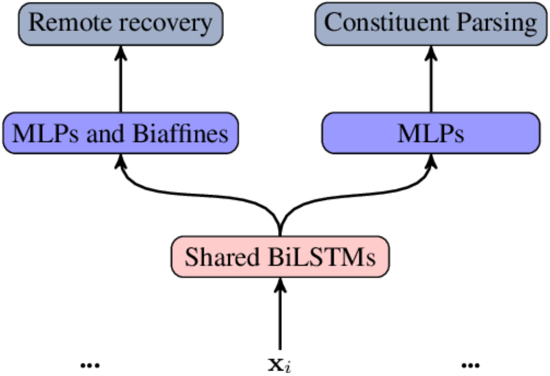
- Data: UCCA for English, French, German.
- Baseline: TUPA.
- Participants: 8 teams from 6 countries.



HLT@SUDA
CUNY-PekingU
baseline
90
80
70
60
50
40
77.4
72.8
80.5
73.5
72.7
67.2
76.7
73.9
68.4
83.2
79.7
73.1
84.1
79.1
75.2
79.6
48.7
English-Wiki (closed)
English-Wiki (open)
English-20K (closed)
English-20K (open)
German-20K (closed)
German-20K (open)
French-20K (open)

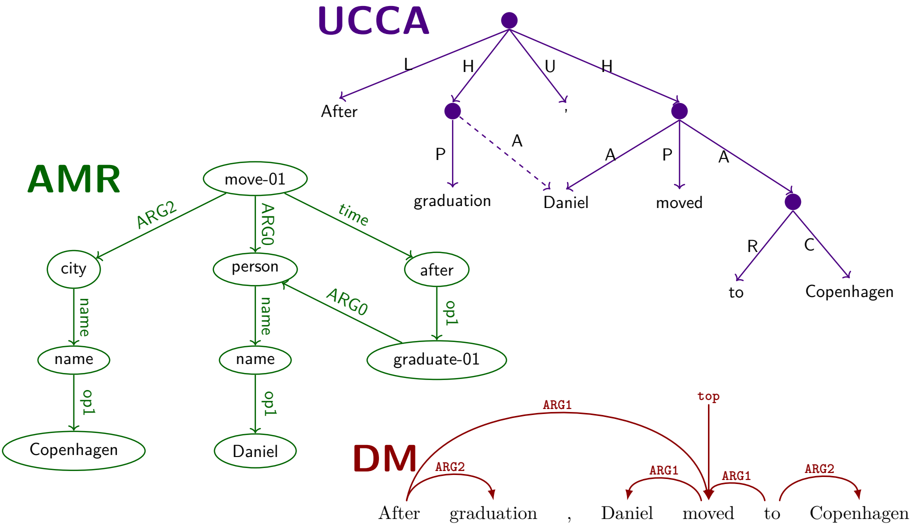
UCCA Graph Parsing as Constituent Tree Parsing

Winning system: HLT@SUDA (Suzhou, China).
Neural constituency parser + multilingual BERT.



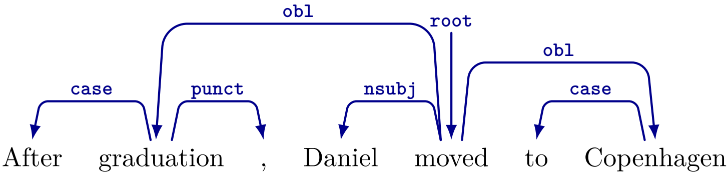
Outline

Meaning Representations



Syntactic Representations

UD (Universal Dependencies)



Data

UCCA training data is scarce

(English)

Data

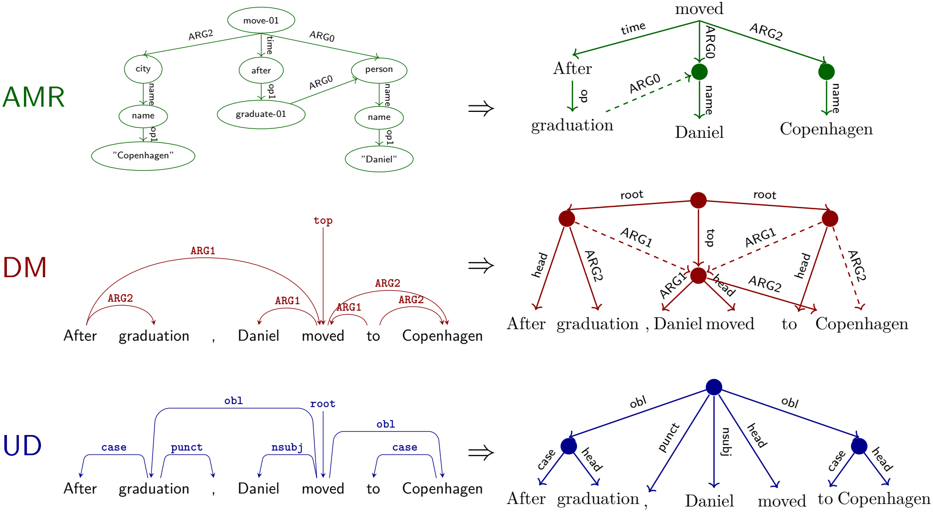
UCCA training data is scarce

(English)

and domains are limited.

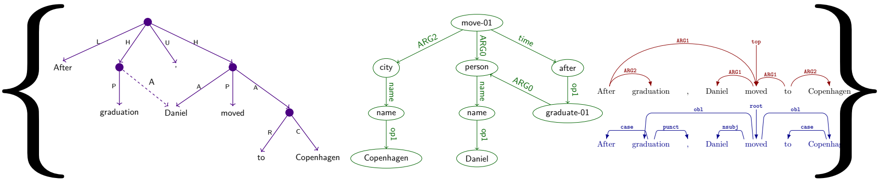
UCCA	AMR	DM	UD
Wikipedia	blogs	news	blogs
books	news		news
reviews	emails		emails
	reviews		reviews
			Q&A

Conversion



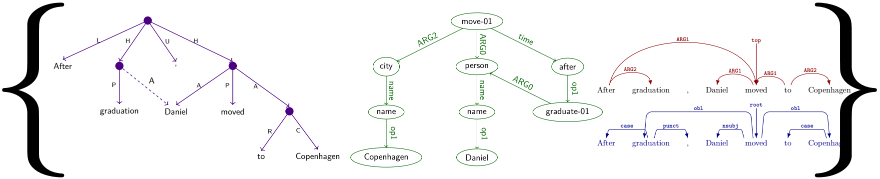
AMR
move-01
ARG2
time
ARG0
city
after
person
name
op1
ARG0
name
graduate-01
name
op1
op1
"Copenhagen"
"Daniel"
⇒
moved
time
ARG0
ARG2
After
op
ARG0
name
name
graduation
Daniel
Copenhagen
DM
top
ARG1
ARG2
ARG1
ARG2
ARG1
ARG2
After graduation , Daniel moved to Copenhagen
⇒
root
root
head
ARG2
ARG1
top
ARG1
head
ARG2
ARG1
head
ARG2
After graduation , Daniel moved to Copenhagen
UD
obl
root
obl
case
punct
nsubj
case
After graduation , Daniel moved to Copenhagen
⇒
obl
obl
punct
nsubj
head
case
head
case
head
After graduation , Daniel moved to Copenhagen

Multi-task

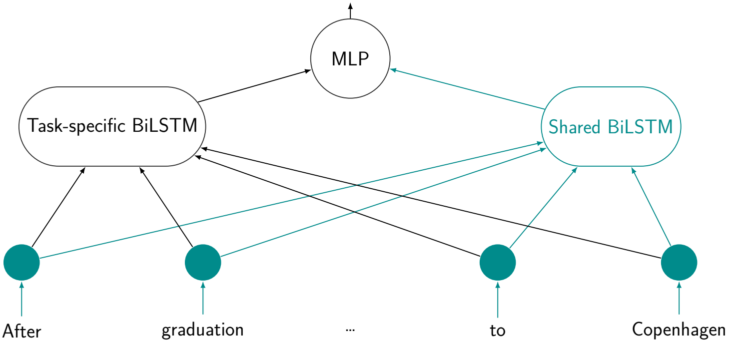


After
graduation
Daniel
moved
to
Copenhagen
L
H
U
H
P
A
A
P
A
R
C
,
move-01
ARG2
ARG0
time
city
person
after
name
name
ARG0
op1
graduate-01
op1
op1
Copenhagen
Daniel
ARG1
top
ARG2
ARG1
ARG1
ARG2
obl
root
obl
case
punct
nsubj
case

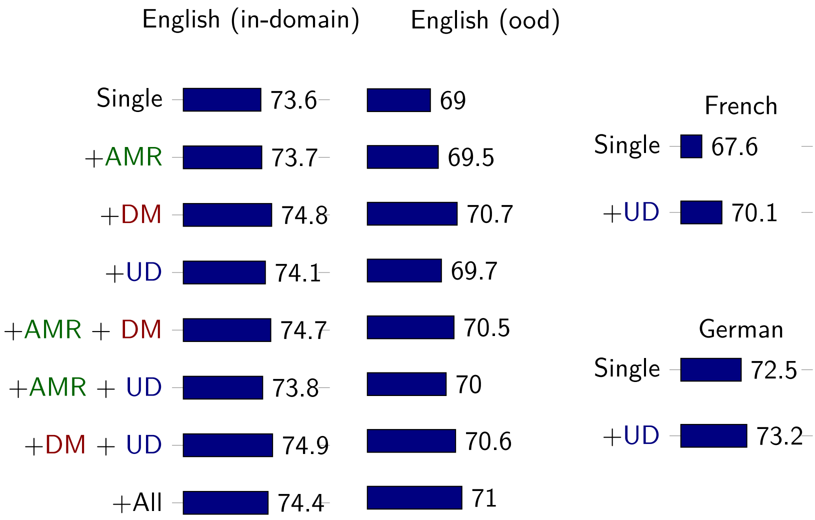
Multi-task



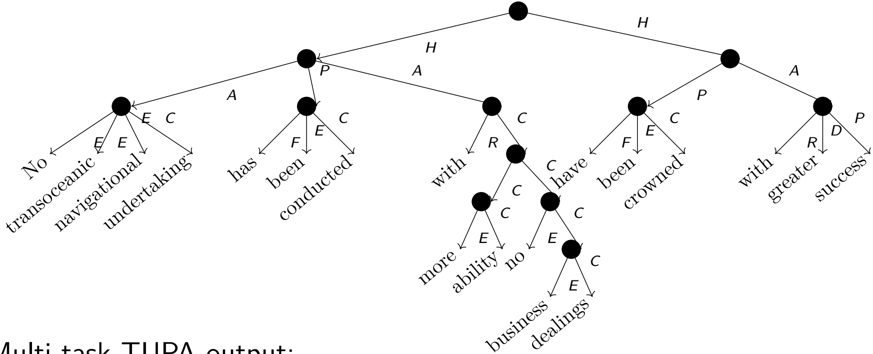
Multi-task TUPA model [Hershcovich, Abend, and Rappoport, 2018]



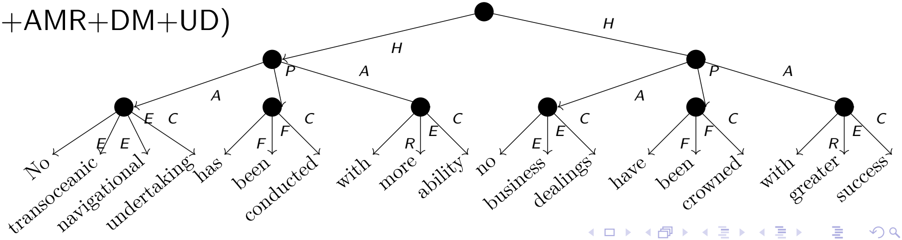
Results


English (in-domain)
English (ood)
Single 73.6 69
+AMR 73.7 69.5
+DM 74.8 70.7
+UD 74.1 69.7
+AMR + DM 74.7 70.5
+AMR + UD 73.8 70
+DM + UD 74.9 70.6
+All 74.4 71
French
Single 67.6
+UD 70.1
German
Single 72.5
+UD 73.2

TUPA output:



Multi-task TUPA output:
(+AMR+DM+UD)

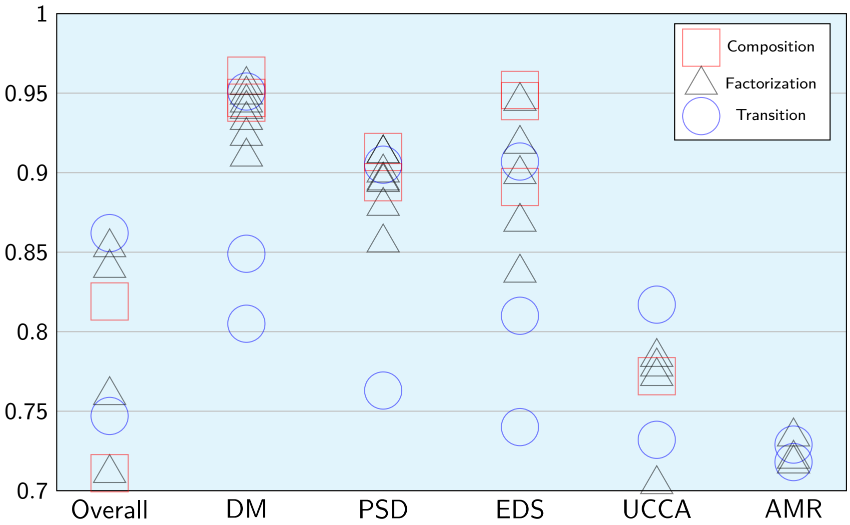


CoNLL 2019: Cross-framework MRP

Shared task: parsing text to graphs in five frameworks [Oepen, Abend, Hajič, Hershcovich, Kuhlmann, O'Gorman, Xue, Chun, Straka, and Urešová, 2019].

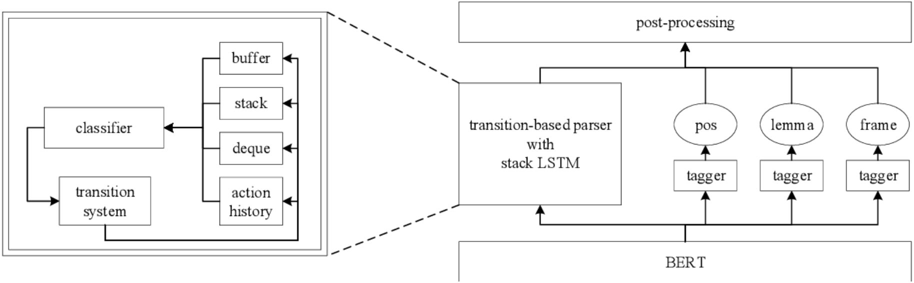
- Data: DM, PSD, EDS, UCCA and AMR for English.
- Baseline: TUPA.
- Participants: 18 teams from 8 countries.

Results


1
0.95
0.9
0.85
0.8
0.75
0.7
Composition
Factorization
Transition
Overall
DM
PSD
EDS
UCCA
AMR

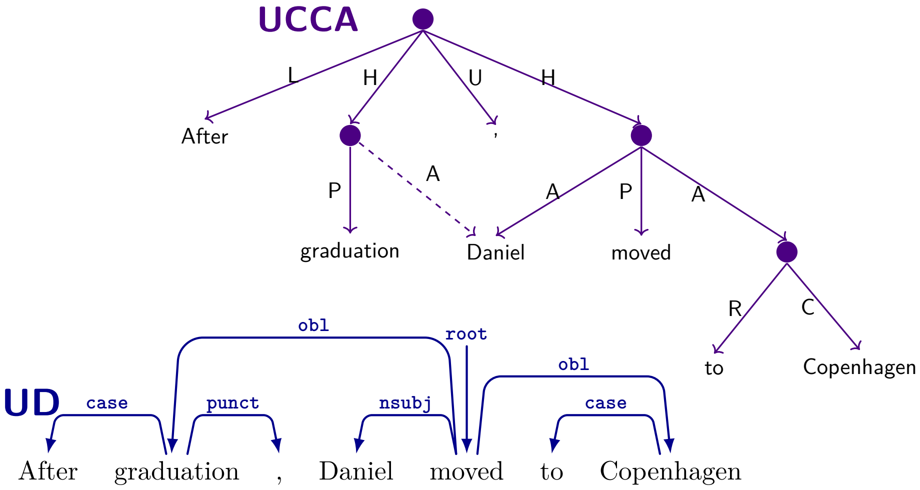
Unified Pipeline for Meaning Representation Parsing

Winning system: HIT-SCIR (Harbin, China).
Transition-based parser (similar to TUPA) + efficient training + BERT.



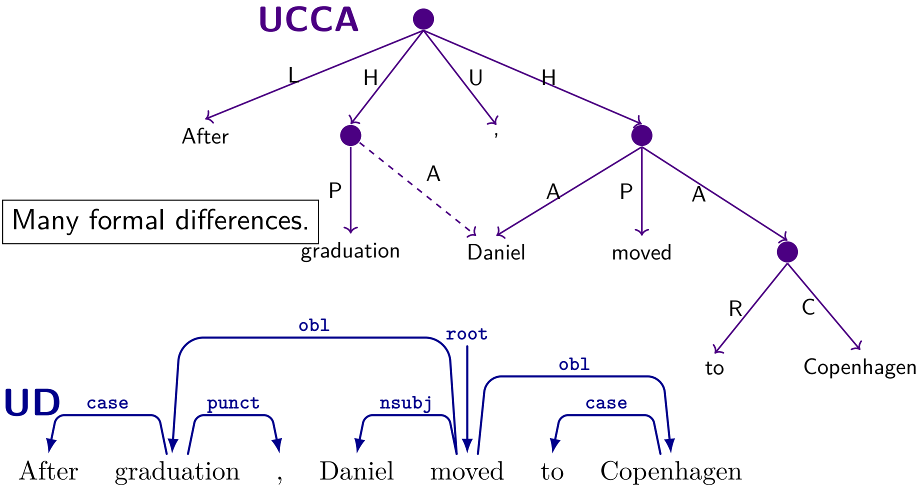
Outline

UCCA vs. UD


UCCA
L
H
U
H
After
,
P
A
A
P
A
graduation
Daniel
moved
R
C
to
Copenhagen
UD
case
obl
punct
nsubj
root
obl
case
After
graduation
,
Daniel
moved
to
Copenhagen

UCCA vs. UD

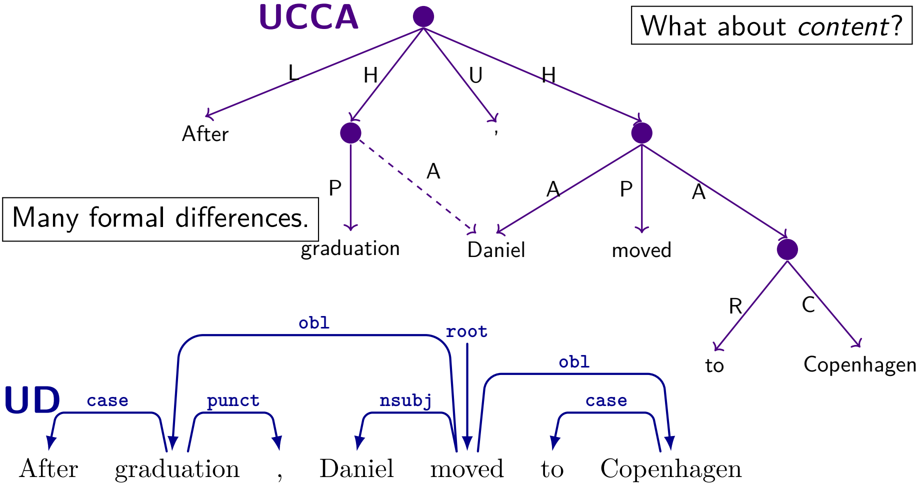
Many formal differences.



UCCA vs. UD

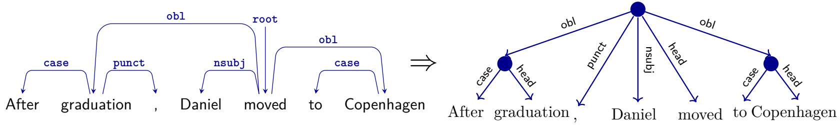
What about *content*?

Many formal differences.

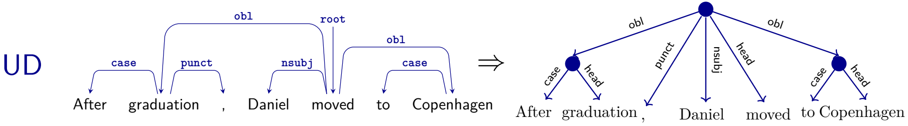


Assimilating the Graph Structures

UD



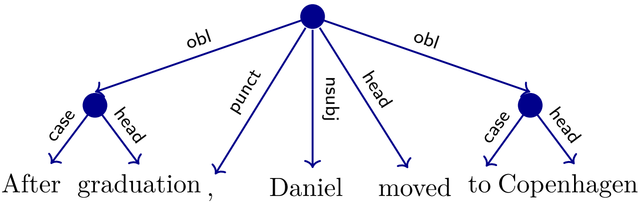
Assimilating the Graph Structures



Evaluate by matching edges [Hershcovich, Abend, and Rappoport, 2019a].

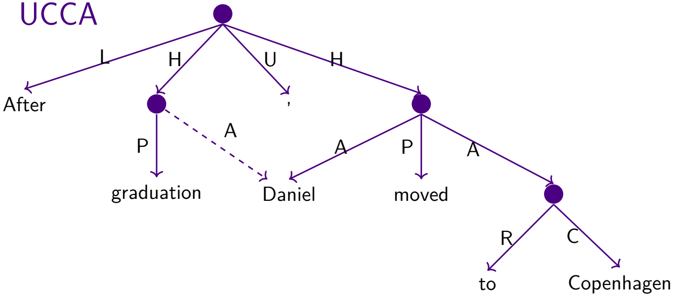
Assimilating the Graph Structures

UD



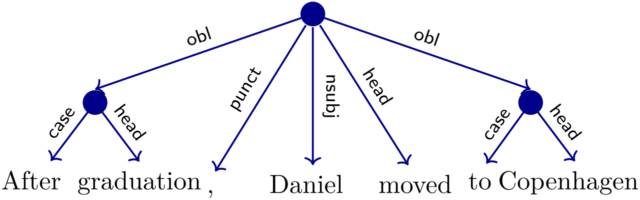
Evaluate by matching edges [Hershcovich, Abend, and Rappoport, 2019a].

UCCA



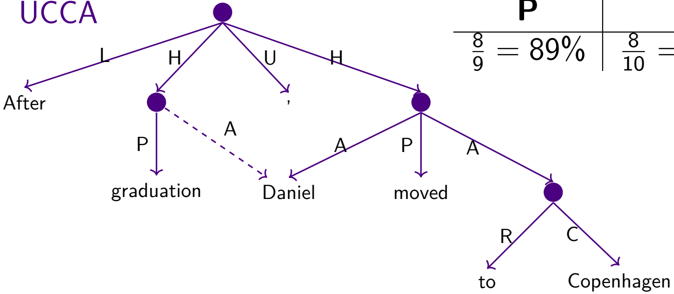
Assimilating the Graph Structures

UD



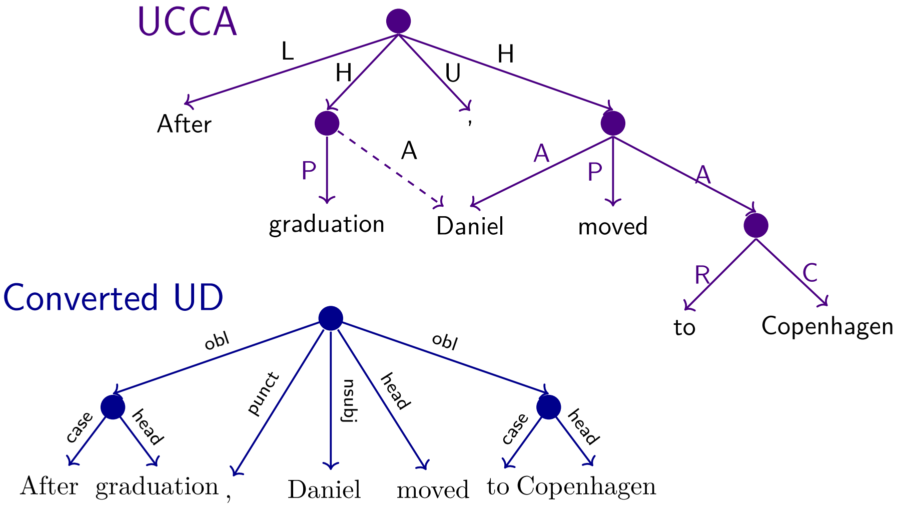
Evaluate by matching edges [Hershcovich, Abend, and Rappoport, 2019a].

UCCA

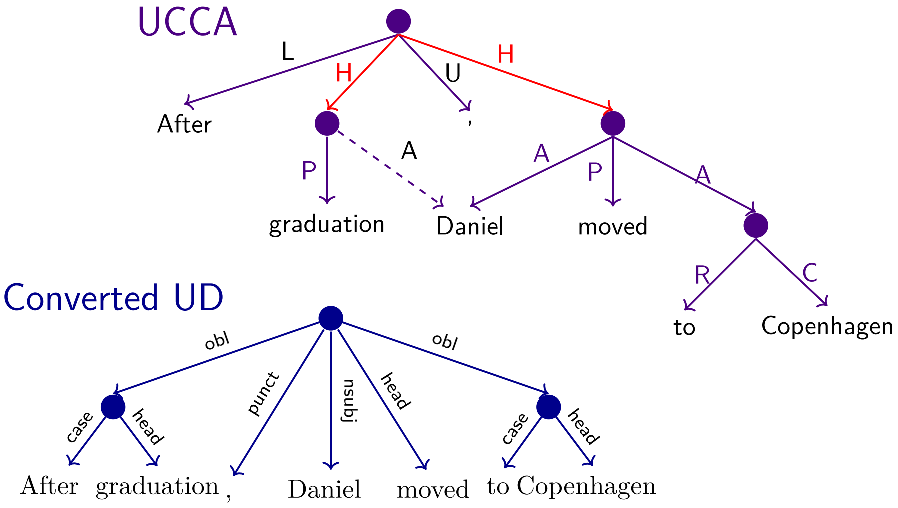


P	R	F1
$\frac{8}{9} = 89\%$	$\frac{8}{10} = 80\%$	84%

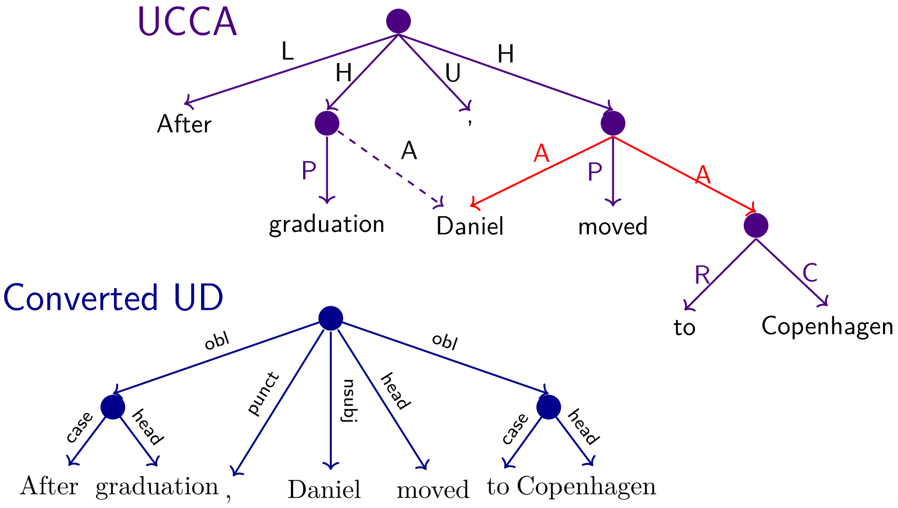
Scenes and non-Scenes, Relations and Participants



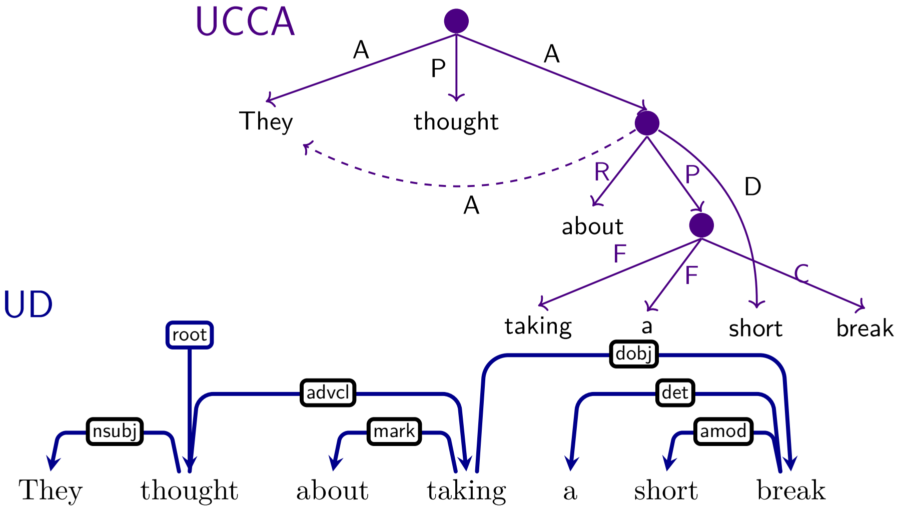
Scenes and non-Scenes, Relations and Participants



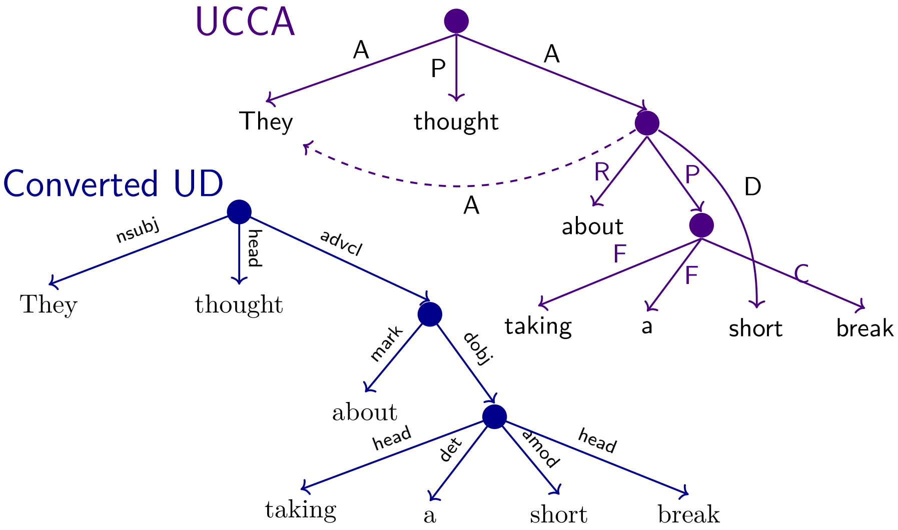
Scenes and non-Scenes, Relations and Participants



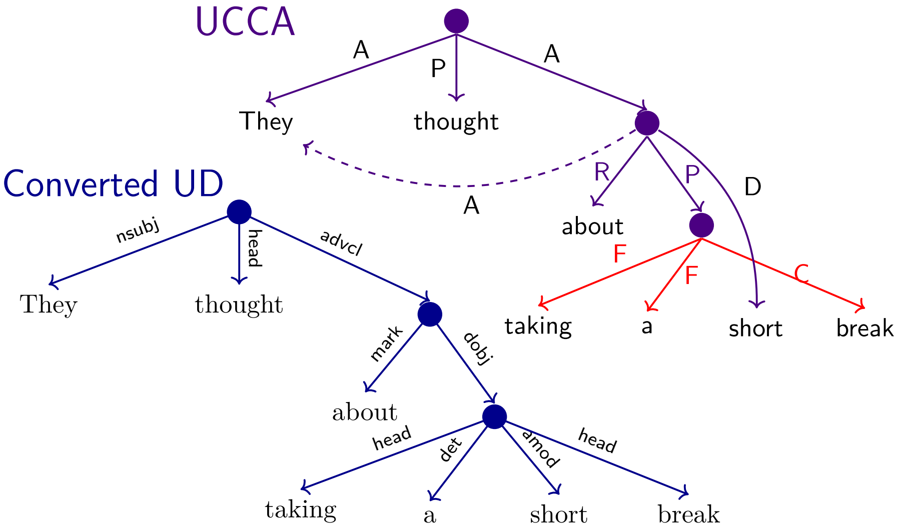
Multi-word Expressions



Multi-word Expressions

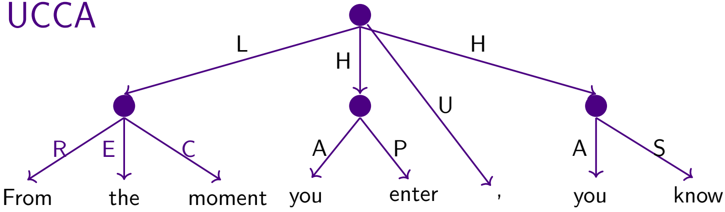


Multi-word Expressions

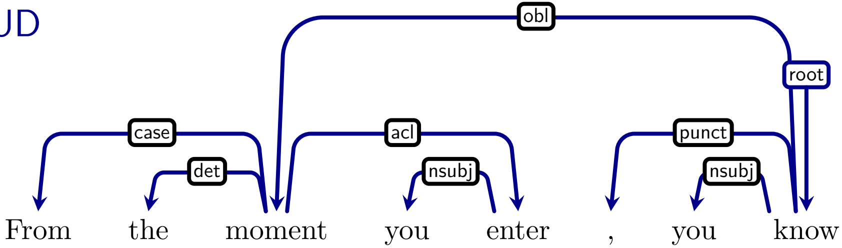


Linkage between Scenes

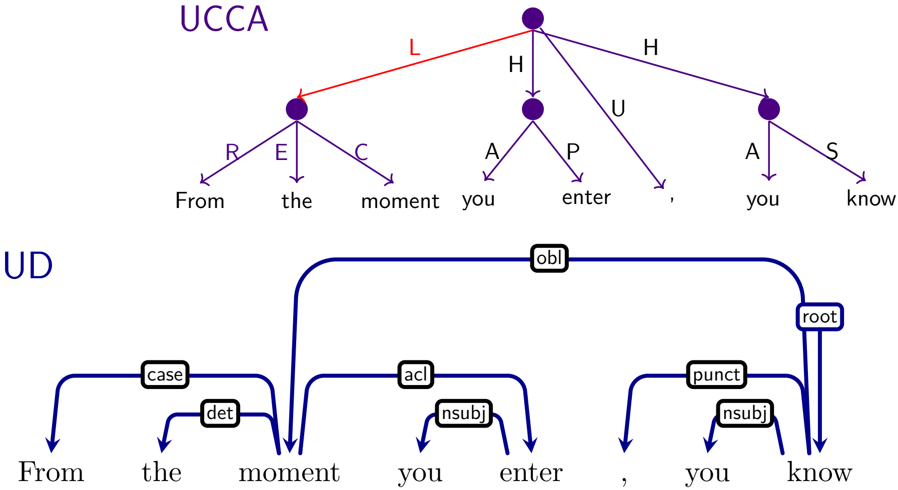
UCCA



UD



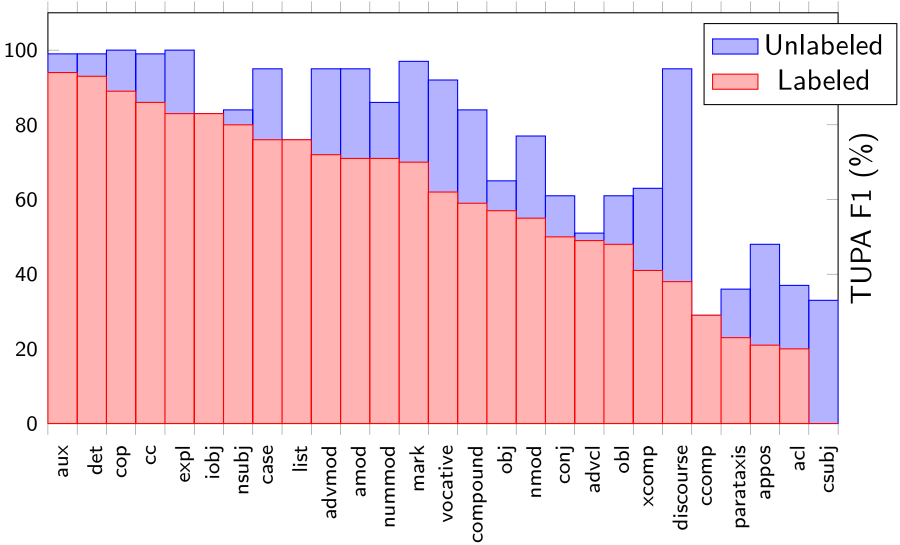
Linkage between Scenes



Confusion Matrix: EWT Reviews Gold Data

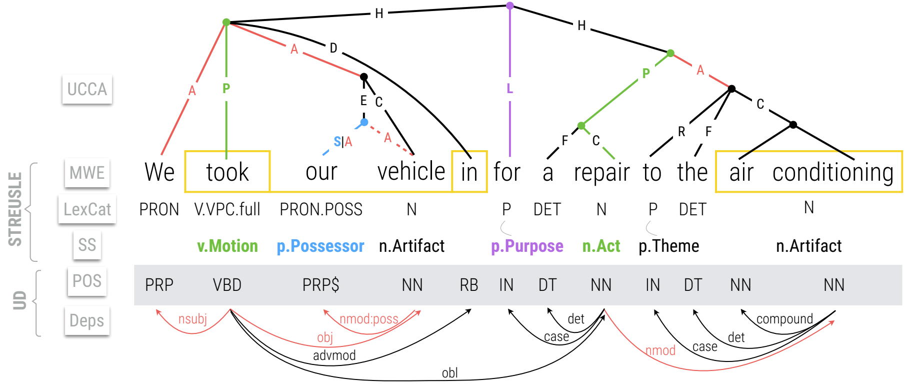
	A	A|P	A|S	C	D	E	F	G	H	L	N	P	Q	R	S	T	No Match
acl	58			1	4	249	1		48			6			1	1	409
advcl	14			12	2	2		6	512	4		11					423
advmod	225		1	69	1778	332	27	135	14	258	2	2	15	44	9	368	273
amod	25			134	647	837		1	28			7	130	3	269	25	176
appos	21			39	2	34			18						8		33
aux					384	2	1335			2		1		1			17
case	11			31	27	25	123			213	26	11	1	2629	154	1	262
cc				8	4	1	4	1	1	1567	381		6	12			52
ccomp	345			1		1			36			2			1	1	166
compound	225			116	67	586	21		2			32	19	1	12	24	683
conj	10			449	4	5		1	1262	1		6	2		10		497
cop				1			1312			1		9		10	178		7
csubj	13								3								46
det	10			17	119	440	2963				1		129	16	1		124
discourse	1			2	1		25	29	27	16					5		19
expl	21			1			98								17		3
iobj	131			1			1										10
list	3			7	2	1			27						1		6
mark				9	7	1	531	1		654				407	1	5	143
nmod	844	1	1	20	9	786	8	4	12	1	1	20	2	2	11	27	488
nsubj	4296	7	21	25	3	2	55	1	5	61		58	1	80	14	4	247
nummod	2			33	12	17		4		4			334				64
obj	1845		1	54	21	6	11	1	4	23		52	1	23	3	11	583
obl	1195			19	115	41	1	17	39	34		6	6	26	7	302	611
parataxis	6		1	5		4		6	285						3		180
vocative	17							8									
xcomp	121			4	25				8			38			38		526
head	445	48	159	6388	717	142	564	83	2462	42	1	4163	120	52	1547	32	2235
No Match	1421	37	58	640	417	291	14	33	2291	146	6	802	94	52	369	96	

Fine-grained UCCA Parsing Evaluation



Ongoing Work

Complement syntax with *lexical* semantics to make up for differences.



Conclusion

- Meaning representation is valuable for language understanding.

Conclusion

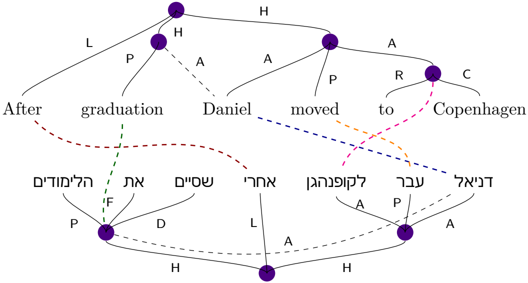
- Meaning representation is valuable for language understanding.
- Transition-based parsers excel across frameworks and languages.

Conclusion

- Meaning representation is valuable for language understanding.
- Transition-based parsers excel across frameworks and languages.
- Cross-framework unification by multi-task and linguistic analysis.

Conclusion

- Meaning representation is valuable for language understanding.
- Transition-based parsers excel across frameworks and languages.
- Cross-framework unification by multi-task and linguistic analysis.



Thanks!

References I

Omri Abend and Ari Rappoport. Universal Conceptual Cognitive Annotation (UCCA). In *Proc. of ACL*, pages 228–238, August 2013. URL http://aclweb.org/anthology/P13-1023.

Alexandra Birch, Omri Abend, Ondřej Bojar, and Barry Haddow. HUME: Human UCCA-based evaluation of machine translation. In *Proc. of EMNLP*, pages 1264–1274, November 2016. URL http://aclweb.org/anthology/D16-1134.

Leshem Choshen and Omri Abend. Reference-less measure of faithfulness for grammatical error correction. In *Proc. of NAACL-HLT*, 2018. URL http://aclweb.org/anthology/N18-2020.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. A transition-based directed acyclic graph parser for ucca. In *Proc. of ACL*, pages 1127–1138, 2017. URL http://aclweb.org/anthology/P17-1104.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. Multitask parsing across semantic representations. In *Proc. of ACL*, pages 373–385, 2018. URL http://aclweb.org/anthology/P18-1035.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. Content differences in syntactic and semantic representation. In *Proc. of NAACL-HLT*, pages 478–488, June 2019a. URL https://aclweb.org/anthology/N19-1047.

Daniel Hershcovich, Leshem Choshen, Elior Sulem, Zohar Aizenbud, Ari Rappoport, and Omri Abend. SemEval 2019 task 1: Cross-lingual semantic parsing with UCCA. In *Proc. of SemEval*, 2019b. URL https://aclweb.org/anthology/S19-2001.

Stephan Oepen, Omri Abend, Jan Hajič, Daniel Hershcovich, Marco Kuhlmann, Tim O'Gorman, Nianwen Xue, Jayeol Chun, Milan Straka, and Zdeňka Urešová. MRP 2019: Cross-framework Meaning Representation Parsing. In *Proc. of CoNLL MRP Shared Task*, pages 1 – 27, 2019. URL https://aclweb.org/anthology/K19-2001.pdf.

Elior Sulem, Omri Abend, and Ari Rappoport. Conceptual annotations preserve structure across translations: A French-English case study. In *Proc. of S2MT*, pages 11–22, 2015. URL http://aclweb.org/anthology/W15-3502.

Elior Sulem, Omri Abend, and Ari Rappoport. Semantic structural annotation for text simplification. In *Proc. of NAACL*, 2018a.

Elior Sulem, Omri Abend, and Ari Rappoport. Simple and effective text simplification using semantic and neural methods. In *Proc. of ACL*, 2018b.