Glue semantics for Universal Dependencies

Matthew Gotham and Dag Haug

University of Oslo

The 23rd International Lexical-Functional Grammar Conference
17 July 2018

Introduction

- Universal Dependencies (UD) is a *de facto* annotation standard for cross-linguistic annotation of syntactic structure

Introduction

- Universal Dependencies (UD) is a *de facto* annotation standard for cross-linguistic annotation of syntactic structure
- $\rightarrow$ interest in deriving semantic representations from UD structures, ideally in a language-independent way

Introduction

- Universal Dependencies (UD) is a *de facto* annotation standard for cross-linguistic annotation of syntactic structure
- $\rightarrow$ interest in deriving semantic representations from UD structures, ideally in a language-independent way
- Our approach: adapt and exploit techniques from LFG + Glue semantics
  - dependency structures $\approx$ f-structures
  - LFG inheritance in UD (via Stanford dependencies)
  - Glue offers a syntax-semantics interace where syntax can underspecify semantics

Introduction

- Universal Dependencies (UD) is a *de facto* annotation standard for cross-linguistic annotation of syntactic structure
- $\rightarrow$ interest in deriving semantic representations from UD structures, ideally in a language-independent way
- Our approach: adapt and exploit techniques from LFG + Glue semantics
  - dependency structures $\approx$ f-structures
  - LFG inheritance in UD (via Stanford dependencies)
  - Glue offers a syntax-semantics interace where syntax can underspecify semantics
- Postpone the need for language-specific, lexical resources

Outline

Plan

1. Target representations
2. Universal Dependencies
3. Our pipeline
4. Evaluation and discussion

Target representations

- Our target representations for sentence meanings are DRSs.
- The format of these DRSs is inspired by Boxer (Bos, 2008).

Target representations

- Our target representations for sentence meanings are DRSs.
- The format of these DRSs is inspired by Boxer (Bos, 2008).
- We assume discourse referents (drefs) of three sorts: entities (x_n), eventualities (e_n) and propositions (p_n).

Target representations

- Our target representations for sentence meanings are DRSs.
- The format of these DRSs is inspired by Boxer (Bos, 2008).
- We assume discourse referents (drefs) of three sorts: entities (x_n), eventualities (e_n) and propositions (p_n).
- The predicate *ant* means that its argument has an antecedent (it's a presupposed dref).

  $\rightarrow$ Also applies to the predicates beginning *pron.*_
- The connective ∂ marks presupposed conditions—it maps TRUE to TRUE and is otherwise undefined.

  $\rightarrow$ Unlike Boxer, which has separate DRSs for presupposed and asserted material.

An example

(1) Abrams persuaded the dog to bark.

Boxer:

$$\left(\begin{array}{|l|}\hline x_2 \\ \hline dog(x_2) \\ \hline \end{array} + \begin{array}{|l|}\hline x_1 \;\; e_1 \;\; p_1 \\ \hline named(x_1, abrams) \\ persuade(e_1) \\ agent(e_1, x_1) \\ theme(e_1, x_2) \\ content(e_1, p_1) \\ p_1 : \begin{array}{|l|}\hline e_2 \\ \hline bark(e_2) \\ agent(e_2, x_2) \\ \hline \end{array} \\ \hline \end{array}\right)$$

Us:

$$\begin{array}{|l|}\hline x_1 \;\; x_2 \;\; e_1 \;\; p_1 \\ \hline named(x_1, abrams) \\ ant(x_2) \\ \partial(dog(x_2)) \\ persuade(e_1) \\ agent(e_1, x_1) \\ theme(e_1, x_2) \\ content(e_1, p_1) \\ p_1 : \begin{array}{|l|}\hline e_2 \\ \hline bark(e_2) \\ agent(e_2, x_2) \\ \hline \end{array} \\ \hline \end{array}$$

Other running examples

(taken from the CCS development suite)

(2) He hemmed and hawed.

x_1 e_1 e_2
$pron.he(x_1)$ $hem(e_1)$ $agent(e_1, x_1)$ $haw(e_2)$ $agent(e_2, x_1)$

(3) The dog they thought we admired barks.

$$
\begin{array}{|l|}
\hline
x_1 \; x_2 \; x_3 \; e_1 \; e_2 \; p_1 \\
\hline
ant(x_1), \partial(dog(x_1)) \\
pron.they(x_2), pron.we(x_3) \\
bark(e_1), agent(e_1, x_1) \\
\partial(think(e_2)), \partial(agent(e_2, x_2)) \\
\partial(content(e_2, p_1)) \\
p_1 : \begin{array}{|l|}
\hline
e_3 \\
\hline
admire(e_3) \\
agent(e_3, x_3) \\
theme(e_3, x_1) \\
\hline
\end{array} \\
\hline
\end{array}
$$

Underlying logic

- The Glue approach relies on meanings being put together by application and abstraction, so we need some form of compositional or λ-DRT for meaning construction.

$$\text{someone} \rightsquigarrow \lambda P.\boxed{\begin{array}{l} x_1 \\ \hline person(x_1) \end{array}} \; ; \; P(x_1)$$

Underlying logic

- The Glue approach relies on meanings being put together by application and abstraction, so we need some form of compositional or λ-DRT for meaning construction.

$$\text{someone} \rightsquigarrow \lambda P.\boxed{\begin{array}{c} x_1 \\ \hline person(x_1) \end{array}} \; ; \; P(x_1)$$

- Conceptually, we are assuming PCDRT (Haug, 2014), which has a definition of the *ant* predicate and (relatedly) a treatment of so-far-unresolved anaphora that doesn't require indexing.
- This specific assumption is not crucial, though.

Plan

'Manning's Law'

(from universaldependencies.org)

'[The UD design is] a very subtle compromise between approximately 6 things:

1. UD needs to be satisfactory on linguistic analysis grounds for individual languages.
2. UD needs to be good for linguistic typology [. . .].
3. UD must be suitable for rapid, consistent annotation by a human annotator.
4. UD must be suitable for computer parsing with high accuracy.
5. UD must be easily comprehended and used by a non-linguist [. . .].
6. UD must support well downstream language understanding tasks [. . .].

It's easy to come up with a proposal that improves UD on one of these dimensions. The interesting and difficult part is to improve UD while remaining sensitive to all these dimensions.'

Syntactic relations

	Nominals	Clauses	Modifier words	Function Words
Core arguments	nsubj obj iobj	csubj ccomp xcomp		
Non-core dependents	obl vocative expl dislocated	advcl	advmod* discourse	aux cop mark
Nominal dependents	nmod appos nummod	acl	amod	det clf case
Coordination	**MWE**	**Loose**	**Special**	**Other**
conj cc	fixed flat compound	list parataxis	orphan goeswith reparandum	punct root dep

Theoretical considerations

- Dependency grammars have severe expressivity constraints
  - Unique head constraint
  - Overt token constraint

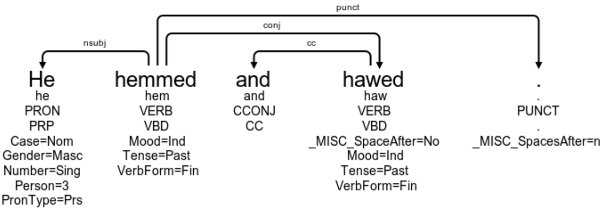
Theoretical considerations

- Dependency grammars have severe expressivity constraints
  - Unique head constraint
  - Overt token constraint
- There are also some UD-specific choices
  - No argument/adjunct distinction

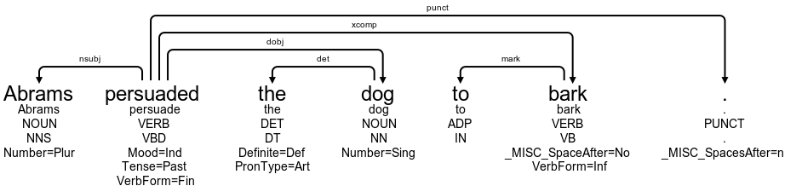
Theoretical considerations

- Dependency grammars have severe expressivity constraints
    - Unique head constraint
    - Overt token constraint
- There are also some UD-specific choices
    - No argument/adjunct distinction
- Some of this will be alleviated through enhanced dependencies but those are not yet widely available

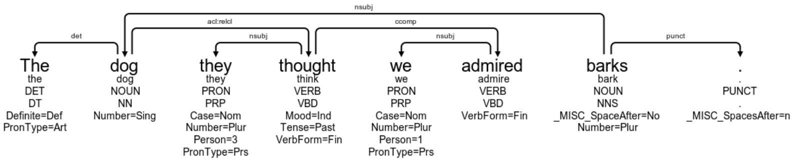
Coordination structure



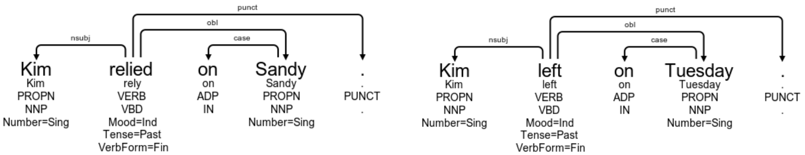
Control structure



Relative clause structure

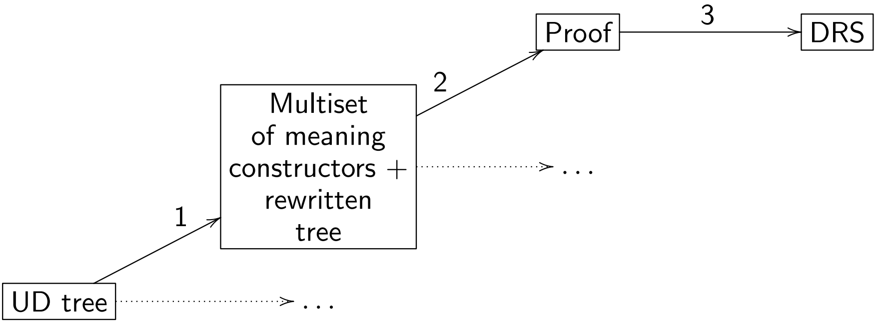


No argument/adjunct distinction



Plan

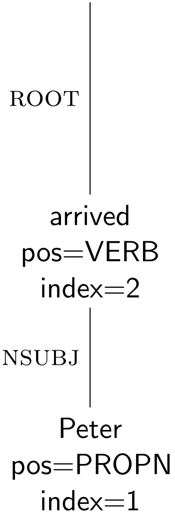
Overview


Proof
3
DRS
2
Multiset of meaning constructors + rewritten tree
. . .
1
UD tree
. . .

Overview

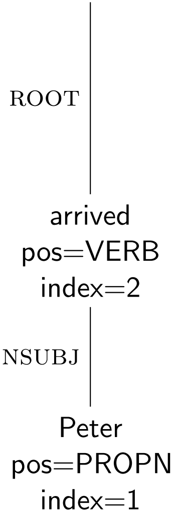
- Traversal of the UD tree, matching each node against a rule file
- For each matched rule, a meaning constructor is produced. . .
- . . . and then instantiated non-deterministically, possibly rewriting the UD tree in the process
- The result is a set of pairs $\langle M, T \rangle$ where M is a multiset of meaning constructors and T is a rewritten UD tree
- Each multiset is fed into a linear logic prover (by Miltiadis Kokkonidis) and beta reduction software (from Johan Bos' Boxer)

Example



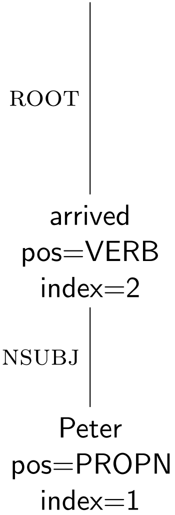
$$\text{pos} = \text{PROPN} \rightarrow$$
$$\lambda P.[x|\mathit{named}(x, \mathit{:lemma:})] \; ; \; P(x) :$$
$$(e_{\downarrow} \multimap t_{\%R}) \multimap t_{\%R}$$

Example



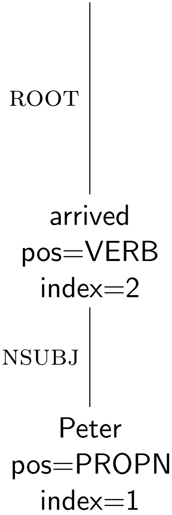
$$\text{pos} = \text{PROPN} \rightarrow$$
$$\lambda P.[x|named(x, Peter)] \; ; \; P(x) :$$
$$(e_1 \multimap t_2) \multimap t_2$$

Example



pos = VERB $\rightarrow$
$\lambda F.[e|\mathit{:lemma:}(e)]; \mathit{:DEP:}(e); F(e)$:
$(v_{\downarrow} \multimap t_{\downarrow}) \multimap t_{\downarrow}$

Example



$$\text{pos} = \text{VERB} \rightarrow$$
$$\lambda x.\lambda F.[e|\mathit{arrive}(e), \mathit{nsubj}(e, x)] \; ; F(e) :$$
$$e_{\downarrow nsubj} \multimap (v_{\downarrow} \multimap t_{\downarrow}) \multimap t_{\downarrow}$$

Example

ROOT

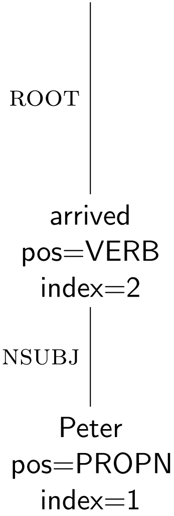
arrived
pos=VERB
index=2

NSUBJ

Peter
pos=PROPN
index=1

$$\text{pos} = \text{VERB} \rightarrow$$
$$\lambda x.\lambda F.[e|arrive(e), nsubj(e,x)] \; ; F(e) :$$
$$e_1 \multimap (v_2 \multimap t_2) \multimap t_2$$

Example



$$\text{relation} = \text{ROOT} \rightarrow$$
$$\lambda\_.[\,|\,] : v(\downarrow) \multimap t(\downarrow)$$

Example

ROOT

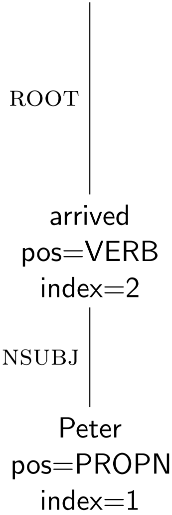
arrived
pos=VERB
index=2

NSUBJ

Peter
pos=PROPN
index=1

relation = ROOT $\rightarrow$
$\lambda\_.[\ |\ ] : v_2 \multimap t_2$

Example



$$\lambda P.[x_1 | \mathit{named}(x_1, \mathit{Peter})] \; ; \; P(x_1) : \\ (e_1 \multimap t_2) \multimap t_2$$

$$\lambda x.\lambda F.[e_1 | \mathit{arrive}(e_1), \mathit{nsubj}(e_1, x)] \; ; F(e_1) : \\ e_1 \multimap (v_2 \multimap t_2) \multimap t_2$$

$$\lambda\_.[\;|\;] : v_2 \multimap t_2$$

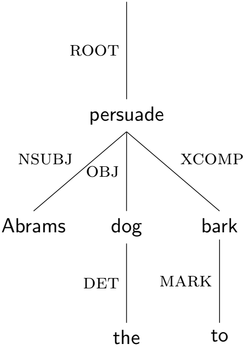
Interpretation in Glue

$$\dfrac{\dfrac{\llbracket Peter \rrbracket : \\ (e_1 \multimap t_2) \multimap t_2}{} \quad \dfrac{\dfrac{\dfrac{\dfrac{\llbracket arrived \rrbracket : \\ e_1 \multimap (v_2 \multimap t_2) \multimap t_2 \quad [y : e_1]^1}{\llbracket arrived \rrbracket(y) : (v_2 \multimap t_2) \multimap t_2} \multimap_E \quad \llbracket \mathtt{root} \rrbracket : v_2 \multimap t_2}{\llbracket arrived \rrbracket(y)(\llbracket \mathtt{root} \rrbracket) : t_2} \multimap_E}{\lambda y.\llbracket arrived \rrbracket(y)(\llbracket \mathtt{root} \rrbracket) : e_1 \multimap t_2} \multimap_{I,1}}{}}{\llbracket Peter \rrbracket(\lambda y.\llbracket arrived \rrbracket(y)(\llbracket \mathtt{root} \rrbracket)) : t_2} \multimap_E$$

$$\left(\lambda P.\boxed{\begin{array}{l} x_1 \\ \hline named(x_1, Peter) \end{array}}; P(x_1)\right) \left(\lambda y. \left(\lambda x.\lambda F.\boxed{\begin{array}{l} e_1 \\ \hline arrive(e_1) \\ nsubj(e_1, x) \end{array}}; F(e_1)\right)(y) \left(\lambda V_{-}.\boxed{\begin{array}{l} \\ \hline \\ \end{array}}\right)\right)$$

$$\rightsquigarrow_\beta \boxed{\begin{array}{l} x_1 \;\; e_1 \\ \hline named(x_1, Peter) \\ arrive(e_1) \\ nsubj(e_1, x_1) \end{array}}$$

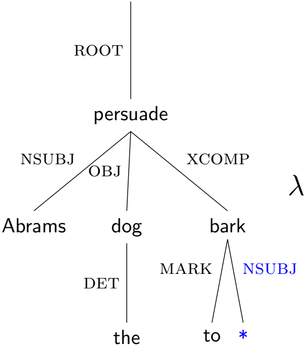
Control



$$\lambda P.\lambda y.\lambda x.\lambda F. \left[\begin{array}{l} e_1\ x_1\ x_2 \\ \hline persuade(e_1) \\ controldep(e_1, x_2) \\ xcomp(e_1, x_1) \\ obj(e_1, y) \\ nsubj(e_1, x) \\ q : P(x_2)(\lambda\_.[\ |\ ]) \end{array}\right] ; F(e_1)$$

$$(e_{\downarrow\text{XCOMP NSUBJ}} \multimap (v_{\downarrow\text{XCOMP}} \multimap t_{\downarrow\text{XCOMP}}) \multimap t_{\downarrow\text{XCOMP}}) \multimap (e_{\downarrow\text{NSUBJ}}) \multimap (e_{\downarrow\text{OBJ}}) \multimap (v_{\downarrow} \multimap t_{\downarrow}) \multimap t_{\downarrow}$$

Control



$\lambda P.\lambda y.\lambda x.\lambda F.$

$e_1 \;\; x_1 \;\; x_2$
$persuade(e_1)$ $controldep(e_1, x_2)$ $xcomp(e_1, x_1)$ $obj(e_1, y)$ $nsubj(e_1, x)$ $q : P(x_2)(\lambda\_.[\;|\;])$

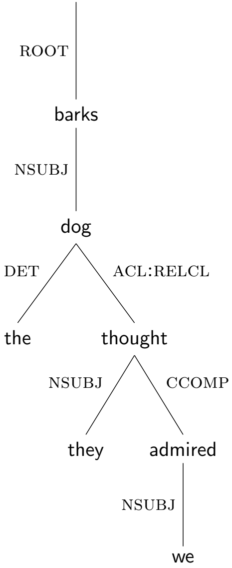
$; F(e_1)$

$$(e_8 \multimap (v_6 \multimap t_6) \multimap t_6) \multimap e_4 \multimap e_1 \multimap (v_2 \multimap t_2) \multimap t_2$$

$$
\dfrac{
\dfrac{[\![Abrams]\!] : (e_1 \multimap t_2) \multimap t_2}{}
\quad
\dfrac{
\dfrac{
\begin{array}{c}\vdots \\ [\![the]\!]([\![dog]\!]) : (e_4 \multimap t_2) \multimap t_2\end{array}
\quad
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\begin{array}{c}[\![persuade]\!] : \\ ((v_6 \multimap t_6) \multimap t_6) \multimap \\ e_4 \multimap e_1 \multimap (v_2 \multimap t_2) \multimap t_2\end{array}
\quad
\begin{array}{c}[\![bark]\!] : \\ (v_6 \multimap t_6) \multimap t_6\end{array}
}{[\![persuade]\!]([\![bark]\!]) : e_4 \multimap e_1 \multimap (v_2 \multimap t_2) \multimap t_2}
\quad [u : e_4]^1
}{[\![persuade]\!]([\![bark]\!])(u) : e_1 \multimap (v_2 \multimap t_2) \multimap t_2}
\quad [v : e_1]^2
}{[\![persuade]\!]([\![bark]\!])(u)(v) : (v_2 \multimap t_2) \multimap t_2}
\quad
\begin{array}{c}[\![\mathtt{root}]\!] : \\ v_2 \multimap t_2\end{array}
}{[\![persuade]\!]([\![bark]\!])(u)(v)([\![\mathtt{root}]\!]) : t_2}
}{\lambda u.[\![persuade]\!]([\![bark]\!])(u)(v)([\![\mathtt{root}]\!]) : e_4 \multimap t_2}\ 1
}{[\![the]\!]([\![dog]\!])(\lambda u.[\![persuade]\!]([\![bark]\!])(u)(v)([\![\mathtt{root}]\!])) : t_2}
}{\lambda v.[\![the]\!]([\![dog]\!])(\lambda u.[\![persuade]\!]([\![bark]\!])(u)(v)([\![\mathtt{root}]\!])) : e_1 \multimap t_2}\ 2
}{[\![Abrams]\!](\lambda v.[\![the]\!]([\![dog]\!])(\lambda u.[\![persuade]\!]([\![bark]\!])(u)(v)([\![\mathtt{root}]\!]))) : t_2}
$$

$$
\rightsquigarrow_\beta
\begin{array}{|l|}
\hline
x_1 \;\; x_2 \;\; x_3 \;\; e_1 \;\; p_1 \\
\hline
named(x_1, abrams), ant(x_2) \\
\partial(dog(x_2)), persuade(e_1) \\
nsubj(e_1, x_1), obj(e_1, x_2) \\
controldep(e_1, x_3), xcomp(e_1, p_1) \\
p_1 : \begin{array}{|l|}
\hline
e_2 \\
\hline
bark(e_2) \\
nsubj(e_2, x_3) \\
\hline
\end{array} \\
\hline
\end{array}
$$

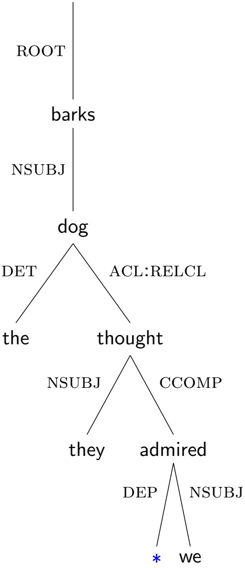
Relative clauses



$$\lambda P.\lambda V.\lambda x.P(x); V(x)(\lambda\_.[\ |\ ])$$

$$(e_{\uparrow} \multimap t_{\uparrow}) \multimap$$
$$(e_{\downarrow dep^* \, dep\{PType=Rel\}} \multimap (v_{\downarrow} \multimap t_{\downarrow}) \multimap t_{\downarrow}) \multimap$$
$$e_{\uparrow} \multimap t_{\uparrow}$$

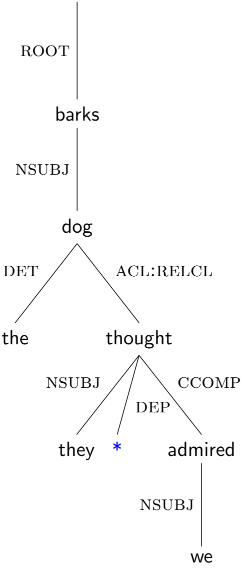
Relative clauses



$$\lambda P.\lambda V.\lambda x.P(x); V(x)(\lambda\_.[\ |\ ])$$

$$(e_2 \multimap t_2) \multimap$$
$$(e_9 \multimap (v_4 \multimap t_4) \multimap t_4) \multimap$$
$$e_2 \multimap t_2$$

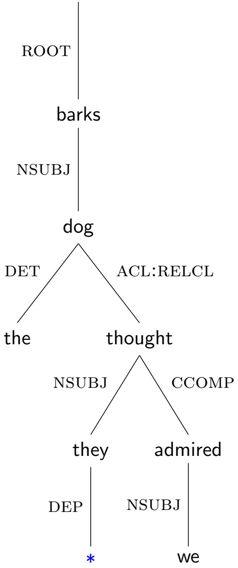
Relative clauses



$$\lambda P.\lambda V.\lambda x.P(x); V(x)(\lambda\_.[\ |\ ])$$

$(e_2 \multimap t_2) \multimap$
$(e_9 \multimap (v_4 \multimap t_4) \multimap t_4) \multimap$
$\quad e_2 \multimap t_2$

Relative clauses



$$\lambda P.\lambda V.\lambda x.P(x); V(x)(\lambda\_.[\ |\ ])$$

$$(e_2 \multimap t_2) \multimap$$
$$(e_9 \multimap (v_4 \multimap t_4) \multimap t_4) \multimap$$
$$e_2 \multimap t_2$$

Other rules

relation = case; ↑↑ {coarsePos = VERB} →
    lam(Y,(lam(X,drs([ ],[rel(:LEMMA:,Y,X)])))) : e(↑)⊸v(↑↑)⊸t(↓)
relation = case; ↑↑ {coarsePos = VERB} →
relation = case →
    lam(Y,(lam(X,drs([ ],[rel(:LEMMA:,Y,X)])))) : e(↑)⊸e(↑↑)⊸t(↓)

coarsePos = DET, lemma = a; ↑ cop { } →

relation = conj; det { } →
lam(X,lam(Q,lam(C,lam(Y,app(app(C,drs([],[leq(X,Y)])),app(app(Q,C),Y)))))
    e(↓)⊸((t(↑)⊸t(↑)⊸t(↑))⊸n(↑))⊸(t(↑)⊸t(↑)⊸t(↑))⊸n(↑)

Plan

Discussion of output

$x_1 \;\; e_1$
$named(x_1, Peter)$ $arrive(e_1)$ $nsubj(e_1, x_1)$

- What kind of θ-role is 'nsubj'?
  - A syntactic name, lifted from the arc label.
  - In and of itself, uninformative.

Discussion of output

x_1 e_1
$named(x_1, Peter)$ $arrive(e_1)$ $nsubj(e_1, x_1)$

- What kind of θ-role is 'nsubj'?
  - A syntactic name, lifted from the arc label.
  - In and of itself, uninformative.
- What we have in the DRS above is as much information as can be extracted from the UD tree alone, without lexical knowledge.
- Lexical knowledge in the form of meaning postulates such as (4) can be harnessed to further specify the meaning representation.

(4) $\forall e \forall x((arrive(e) \wedge nsubj(e, x)) \rightarrow theme(e, x))$

Discussion of output

x_1 e_1
$named(x_1, Peter)$
$arrive(e_1)$
$theme(e_1, x_1)$

- What kind of θ-role is ‘nsubj’?
  - A syntactic name, lifted from the arc label.
  - In and of itself, uninformative.
- What we have in the DRS above is as much information as can be extracted from the UD tree alone, without lexical knowledge.
- Lexical knowledge in the form of meaning postulates such as (4) can be harnessed to further specify the meaning representation.

(4) $$\forall e \forall x((arrive(e) \wedge nsubj(e, x)) \rightarrow theme(e, x))$$

$x_1\ x_2\ x_3\ e_1\ p_1$
$\ldots$
$persuade(e_1), obj(e_1, x_2), controldep(e_1, x_3), xcomp(e_1, p_1)$
p_1 : [e_2 | $\ldots, nsubj(e_2, x_3)$]

- The *persuade* + `xcomp` meaning constructor has
  - introduced an *xcomp* relation between the persuading event e_1 and the proposition p_1 that there is a barking event e_2, and
  - introduced an individual x_3 as the *nsubj* of e_2 and the *controldep* of e_1.

<table>
<tr><td colspan="2">$x_1 \ x_2 \ x_3 \ e_1 \ p_1$</td></tr>
<tr><td colspan="2">$\ldots$
$persuade(e_1), obj(e_1, x_2), controldep(e_1, x_3), xcomp(e_1, p_1)$
p_1 : <table><tr><td>e_2</td></tr><tr><td>$\ldots, nsubj(e_2, x_3)$</td></tr></table></td></tr>
</table>

- The *persuade* + `xcomp` meaning constructor has
  - introduced an *xcomp* relation between the persuading event e_1 and the proposition p_1 that there is a barking event e_2, and
  - introduced an individual x_3 as the *nsubj* of e_2 and the *controldep* of e_1.
- But the information that *persuade* is an object control verb can again be encoded in a meaning postulate:

  $\forall e \forall x((persuade(e) \wedge controldep(e, x)) \rightarrow obj(e, x))$

$$\begin{array}{|l|}\hline x_1\ x_2\ x_3\ e_1\ p_1 \\ \hline \ldots \\ \mathit{persuade}(e_1), \mathit{obj}(e_1, x_2), \mathit{obj}(e_1, x_3), \mathit{xcomp}(e_1, p_1) \\ p_1 : \begin{array}{|l|}\hline e_2 \\ \hline \ldots, \mathit{nsubj}(e_2, x_3) \\ \hline \end{array} \\ \hline \end{array}$$

- The *persuade* + `xcomp` meaning constructor has
  - introduced an *xcomp* relation between the persuading event e_1 and the proposition p_1 that there is a barking event e_2, and
  - introduced an individual x_3 as the *nsubj* of e_2 and the *controldep* of e_1.
- But the information that *persuade* is an object control verb can again be encoded in a meaning postulate:

$$\forall e \forall x((\mathit{persuade}(e) \land \mathit{controldep}(e, x)) \rightarrow \mathit{obj}(e, x))$$

x_1 x_2 x_3 e_1 p_1
$\ldots$ $persuade(e_1), obj(e_1, x_2), obj(e_1, x_3), xcomp(e_1, p_1)$ p_1 : [e_2 | $\ldots, nsubj(e_2, x_3)$]

- The *persuade* + `xcomp` meaning constructor has
  - introduced an *xcomp* relation between the persuading event e_1 and the proposition p_1 that there is a barking event e_2, and
  - introduced an individual x_3 as the *nsubj* of e_2 and the *controldep* of e_1.
- But the information that *persuade* is an object control verb can again be encoded in a meaning postulate:

  $\forall e \forall x((persuade(e) \wedge controldep(e, x)) \rightarrow obj(e, x))$
- With thematic uniqueness, we get $x_2 = x_3$ in this case.
- Blurs the distinction between lexical syntax and semantics.

VP/Sentence coordination: He hemmed and hawed

x_1 e_2 e_3
$pron.he(x_1)$ $hem(e_2)$ $nsubj(e_2, x_1)$ $haw(e_3)$

- No way to distinguish V/VP/S coordination in DG because of the overt token constraint
- No argument sharing because of the unique head constraint

NP Coordination: Abrams and/or Browne danced

$$\begin{array}{|l|}\hline e_1 \;\; x_2 \;\; x_3 \;\; x_4 \\ \hline \\ \textit{dance}(e_1) \\ \textit{nsubj}(e, x_2) \\ \textit{named}(x_3, \textit{browne}) \\ \textit{named}(x_4, \textit{abrams}) \\ x_3 \sqsubseteq x_2 \\ x_4 \sqsubseteq x_2 \\ \\ \hline \end{array}$$

$$\begin{array}{|l|}\hline e_1 \;\; x_2 \;\; x_3 \;\; x_4 \\ \hline \\ \textit{dance}(e_1) \\ \textit{nsubj}(e, x_2) \\ \textit{named}(x_3, \textit{browne}) \\ \textit{named}(x_4, \textit{abrams}) \\ \begin{array}{|c|}\hline \\ \hline x_3 \sqsubseteq x_2 \\ \hline \end{array} \vee \begin{array}{|c|}\hline \\ \hline x_4 \sqsubseteq x_2 \\ \hline \end{array} \\ \\ \hline \end{array}$$

Argument/adjunct distinction

$$\begin{array}{|l|}\hline e_1 \ x_2 \ x_3 \\ \hline \\ \mathit{rely}(e_1) \\ \mathit{named}(x_2, \mathit{kim}) \\ \mathit{named}(x_3, \mathit{sandy}) \\ \mathit{on}(x_3, e_1) \\ \\ \hline \end{array} \qquad \begin{array}{|l|}\hline e_1 \ x_2 \ x_3 \\ \hline \\ \mathit{leave}(e_1) \\ \mathit{named}(x_2, \mathit{kim}) \\ \mathit{named}(x_3, \mathit{tuesday}) \\ \mathit{on}(x_3, e_1) \\ \\ \hline \end{array}$$

- Again, we will have to rely on meaning postulates to resolve the *on* relation to a thematic role in one case and a temporal relation in the other

Evaluation

- What we have so far is a proof of concept tested on carefully crafted examples
  - application of LFG techniques (functional uncertainties) to enrich underspecified UD syntax
  - application of glue semantics to dependency structures

Evaluation

- What we have so far is a proof of concept tested on carefully crafted examples
    - application of LFG techniques (functional uncertainties) to enrich underspecified UD syntax
    - application of glue semantics to dependency structures
- Very far from something practically useful
    - Basic coverage of UD relations except `vocative`, `dislocated`, `clf`, `list`, `parataxis`, `orphan`
    - Little or no work on interactions, special constructions, real data noise

Pros and cons of glue semantics

- No need for binarization
- Flexible approach to scoping yield different readings
- Hard to restrict unwanted/non-existing scopings
- Computing lots of uninteresting scope differences

Unwanted scopings

$$\lambda F. \begin{array}{|l|}\hline e \\ \hline arrive(e) \\ \hline\end{array} ; F(e) : (v_1 \multimap t_1) \multimap t_1$$

$$\lambda \_. \begin{array}{|l|}\hline \quad \\ \hline \quad \\ \hline\end{array} : v_1 \multimap t_1$$

It is clear which DRS sentence-level operators (negation, auxiliaries etc.) should target!

- Modalities in the linear logic
- Different types for the two DRSs

Efficient scoping

- Two parameters:
  - level of scope
  - order of combination of quantifiers at each level
- We currently naively compute everything with a light-weight prover
  $\rightarrow$ obvious performance problems
- Disallow intermediate scopings?
- Structure sharing across derivations (building on work in an LFG context)

Conclusions

- Theoretical achievement: application of glue to dependency grammar also exploiting other LFG techniques such as functional uncertainty

Conclusions

- Theoretical achievement: application of glue to dependency grammar also exploiting other LFG techniques such as functional uncertainty
- Practical achievement: an interesting proof of concept implementation

Conclusions

- Theoretical achievement: application of glue to dependency grammar also exploiting other LFG techniques such as functional uncertainty
- Practical achievement: an interesting proof of concept implementation
- Potentially useful for low-resource languages because of postponement of lexical knowledge

Conclusions

- Theoretical achievement: application of glue to dependency grammar also exploiting other LFG techniques such as functional uncertainty
- Practical achievement: an interesting proof of concept implementation
- Potentially useful for low-resource languages because of postponement of lexical knowledge
- Allows combining a data-driven approach to syntactic parsing with a rule-driven interface to logic-based semantics
- But lots of work remains
  - Support for partial proofs
  - Axiomatization of lexical knowledge
  - Ambiguity management

References I

Bos, Johan. 2008. Wide-coverage semantic analysis with Boxer. In *Proceedings of the 2008 conference on semantics in text processing* STEP '08, 277–286. Stroudsburg, PA, USA: Association for Computational Linguistics. `http://dl.acm.org/citation.cfm?id=1626481.1626503`.

Haug, Dag Trygve Truslew. 2014. Partial dynamic semantics for anaphora. *Journal of Semantics* 31. 457–511.