Second order inference in NL semantics

Stephen Pulman

Computational Linguistics Group,
Department of Computer Science,
Oxford University.
stephen.pulman@cs.ox.ac.uk

Aug 2012

Aims

- Capture some apparently trivial NL inferences (and lack of inferences).
- Assumptions: syntax-driven composition semantics producing logical form for a (disambiguated) parsed sentence
- Logical forms sent to automated theorem prover (resolution, tableau...).
- Statements added as 'axioms', questions (inferences) treated as 'theorems' to be proved.
- Yes/no questions: yes if there is a proof; Wh-questions: if a proof, return unifying substitutions as wh-value answers.

Simple example:

All bankers are rich:	axiom: $\forall x.banker(x) \rightarrow rich(x)$
Jones is a banker:	axiom: $banker(jones)$
Is Jones rich?	prove: $rich(jones)$
Who is rich?	prove: $\exists x.rich(x)$

Adjective inferences

Jones is a Welsh rugby player

$\models$ Jones is Welsh

$\models$ Jones is a rugby player

All rugby players are beer drinkers

$\models$ Jones is a Welsh beer drinker, etc.

Minnie is a large mouse

$\models$ Minnie is a mouse

All mice are animals

$\models$ Minnie is an animal

$\not\models$ Minnie is a large animal

Tony Blair is a former Prime Minister

$\not\models$ Tony Blair is a Prime Minister

Smith showed an apparent proof of the theorem

$\not\models$ Smith showed a proof of the theorem

Possessive inferences

Smith is Jones's plumber

$\models$ Smith is a plumber

Smith is also a decorator

$\not\models$ Smith is Jones's decorator

John's wooden toy broke

$\models$ John's toy broke

$\models$ a wooden toy broke

$\models$ a toy broke

A student's textbook's cover intrigued Jones

$\models$ A textbook's cover intrigued Jones

$\models$ A cover intrigued Jones

John's mother or father phoned

John's mother or John's father phoned. etc.

Adjective semantics

Customary to distinguish three subclasses:

- Intersective: dead, Welsh, wooden, foreign...
  Adj N implies both Adj and N
- Subsective: tall, old, green, rigid...
  Adj N implies N, but only Adj-for-an-N, not Adj in general
- Privative: apparent, fake, former, alleged...
  Adj N does not imply N (may even imply not-N)

Truth conditions: ($\mathbf{D}(x)$ = 'denotation of x')
Intersective: 'Jones is a Welsh rugby-player' true iff
$\mathbf{D}(\text{jones}) \in \mathbf{D}(\text{Welsh}) \cap \mathbf{D}(\text{rugby-player})$
Subsective: 'Minnie is a large mouse' true iff
$\mathbf{D}(\text{minnie}) \in \{X \mid X \text{ a mouse larger than relevant standard}\}$
Privative: varies - 'X is a former Y' true iff
$\mathbf{D}(X) \in \mathbf{D}(Y \text{ at earlier time})$, etc.

Logical forms

Fine if we are just doing linguistics, but for computational purposes we need a logical form that will support the relevant inferences proof theoretically. No syntactic difference between types of Adj so build semantic differences into their LFs directly:

Assume syntax/semantics rules like:

NP	$\rightarrow$	Det N'	$Det(N')$
N'	$\rightarrow$	Adj N'	$Adj(N')$
N'	$\rightarrow$	N	N
Adj	$\rightarrow$	wooden etc.	$\lambda Px.wooden(x) \wedge P(x)$
Adj	$\rightarrow$	small, etc.	$\lambda Px.small(x,P)$
Adj	$\rightarrow$	apparent, etc.	$\lambda Px.apparent(x,P)$

- intersective: we get the inferences we want immediately
- subsective: $small(x,P)$ = 'small by the standards relevant for P'. To get the inference that $P(x)$ we add an axiom for each adj: $\forall xP.adj(x,P) \rightarrow P(x)$
- privative: we do not add these axioms and so we (correctly) cannot infer from $apparent(x,P)$ that $P(x)$

But the non-intersective Adj have second order arguments

Like this analysis, many natural language constructs are intrinsically higher order:

- Most dogs bark = *most(dog,bark)*
  type(most) =(et)(et)t, type(dog) = type(bark)= et
- John is very tall = *very(tall)(john)*
  type(very) = (et)(et), type(tall) = et, type(john) = e
- But we only have automated theorem provers for first order logic

Reification or 'ontological promiscuity' attempts to avoid the problem: e.g. event analyses of adverbs:

- John runs quickly = *quickly(run)(john)*
  $\Rightarrow$ $\exists e.run(e,john) \wedge quick(e)$
- or the 'standard translation' of modal logic:
  $\Box p \Rightarrow \forall x.R(thisWorld,x) \rightarrow P(x)$

Reification of standards of Adj-ness?

But it's not obvious how such a strategy could help here:

John is a tall man =?

$\exists s.tall(john,s) \wedge man(john) \wedge tallness\text{-}for\text{-}men(s)$

Not clear how to model interaction with related predicates:

John is a tall man $\models$ John is not a short man

$\forall xyz.tall(x,y) \wedge tallness\text{-}for\text{-}men(y) \rightarrow$
$\neg(short(x,z) \wedge shortness\text{-}for\text{-}men(z))$

Potentially infinite number of such 's' predicates, and therefore such relatedness axioms (ignore conjunctive readings):

this is an old building, an old American building

this is an old English building, old Anglo-Saxon religious site...

Whereas the higher order version generalises cleanly:

...old American building = $old(this,\lambda x.American(x) \wedge building(x))$

and the interaction with related predicates only needs one axiom:

$\forall xP.\ old(x,P) \rightarrow \neg(young(x,P))$

Possessive semantics

a. John's picture/team/sister
b. a picture/team/sister of John's
c. a picture/*team/sister of John
d. That picture/team/sister is John's.

The precise possessive relation is usually contextually inferred:

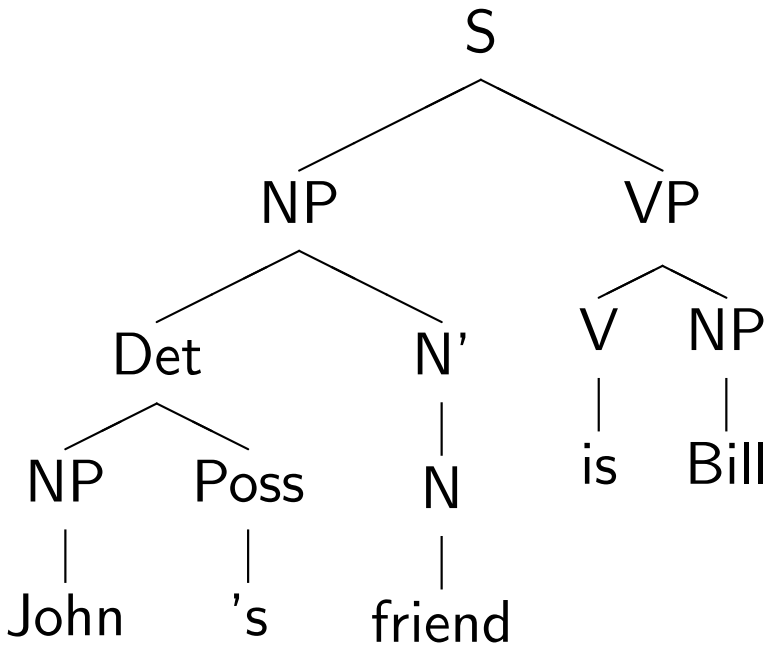
The table's leg...	Monday's lecture...
America's invasion of Iraq...	John's measles...
John's dog...	John's brother...
John's portrait...	etc...

Relational vs. sortal nouns: if there is a relational noun, that usually provides the relation, but not invariably:

> *The history teacher had an argument with one of his parents.*
> *(parents' evening context: parents who came to see him)*
> *This time, Maria's evil daughter will be Goneril*
> *(acting King Lear context: daughter played by Maria)*

An initial simple analysis

A simple analysis (e.g. [Bos et al., 2004, Steedman, 2012]) takes the possessive morpheme 's (or just ' for plurals) to be a function from NP meanings to Det meanings introducing an abstract 'of' or 'poss' relation:



John's friend is Bill = $\exists x.friend(x) \wedge of(x, John) \wedge x{=}Bill$

But this analysis is wrong!

A: John's brother is Bill =

$$\exists x.brother(x) \wedge of(x, John) \wedge x{=}Bill$$

$$\Rightarrow_{=} \quad brother(Bill) \wedge of(Bill,John)$$

B: Bill is a doctor =

$$doctor(Bill)$$

C: Is Bill John's doctor? =

$$\exists x.doctor(x) \wedge of(x, John) \wedge x{=}Bill$$

$$\Rightarrow_{=} \quad doctor(Bill) \wedge of(Bill,John)$$

- but now C is provable from A and B, incorrectly.

Contextual interpretation

More sophisticated analyses
([Partee and Borschev, 2003],[Peters and Westerståhl, 2006]) require contextual interpretation of a predicate variable R_{gen} or *Poss*.

If we interpret 'of'/'Poss'/'R_{gen}' in A as the two-place relation 'brother(Bill,John)', and as something else in C, then the incorrect inference will not be made.

A: John's brother is Bill =

$\exists x.brother(x) \wedge Poss(x, John) \wedge x{=}Bill$

$\Rightarrow_{=}$ $brother(Bill) \wedge brother(Bill,John)$

B: Bill is a doctor =

$doctor(Bill)$

C: Is Bill John's doctor? =

$\exists x.doctor(x) \wedge Poss(x, John) \wedge x{=}Bill$

$\Rightarrow_{=}$ $doctor(Bill) \wedge doctor\text{-}employed\text{-}by(Bill,John)$

Contextual interpretation

Although our invalid inference will not go through when relational nouns are involved, we cannot always guarantee this for sortal nouns (or relational nouns when interpreted sortally):

A: Smith is Bill's plumber = (interpret 'Poss' as 'works-for')
plumber(Smith) $\wedge$ *works-for(Smith,Bill)*
B: Smith is also a decorator
decorator(Smith)
C: Is Smith Bill's decorator?
decorator(Smith) $\wedge$ *works-for(Smith, Bill)*

It's surely difficult to argue that 'Poss' should be instantiated differently in A and C.

One solution: an 'of' relation with a second order argument

$$'s = \lambda OPQ.\exists x.P(x) \wedge O(\lambda y.of(x,y,P)) \wedge Q(x)$$

A: Smith is Bill's plumber =

$$plumber(Smith) \wedge of(Smith,Bill,plumber)$$

B: Smith is also a decorator =

$$decorator(Smith)$$

C: Is Smith Bill's decorator? =

$$decorator(Smith) \wedge of(Smith, Bill,decorator)$$

Now the unwanted inference does not go through. We can remove the duplicate 'P' with additional axioms:

$$'s = \lambda OPQ.\exists x.O(\lambda y.of(x,y,P)) \wedge Q(x)$$

Smith is Bill's plumber = $of(Smith,Bill,plumber)$

A: $\forall xyP.of(x,y,P) \rightarrow P(x)$ (for sortal N)

B: $\forall xyP.of(x,y,P) \rightarrow P\text{-}of(x,y)$ (for relational N)

We don't even have to resolve 'of' to avoid bad inferences, and we don't need to distinguish sortal and relational N syntactically.

Generalises to complex N'

John's wooden toy disappeared.

$\exists x.of(x,John,\lambda y.wooden(y) \wedge toy(y)) \wedge disappeared(x)$

Via axiom A we can deduce: $...[\lambda y.wooden(y) \wedge toy(y)](x)...$ and then by β-reduction that:

$\exists x.of(x,John,\lambda y.wooden(y) \wedge toy(y)) \wedge disappeared(x)$
$\wedge wooden(x) \wedge toy(x)$

Now we can capture the inferences:

A toy disappeared: $\exists x.toy(x) \wedge disappeared(x)$

and:

Something wooden disappeared: $\exists x.wooden(x) \wedge disappeared(x)$

Semantically (though not yet proof-theoretically) it also follows that:
John's toy disappeared: $\exists x.of(x,John,toy) \wedge disappeared(x)$

To capture this we need an axiom to the effect that if $of(x,y,P)$ and if property P entails property Q, then $of(x,y,Q)$

Linguistically fine, but computationally OK?

Unhappily not, because our 'of' predicate has a second order argument, like our analysis of adjective modification, so we cannot do any automated inference directly with it.

Could reification help us here?

Johan Bos ([Bos, 2009]) has suggested a reification approach to this problem.

We translate sentences like 'Vincent is Mia's husband' as:

$$person(Vincent) \wedge \exists y.role(Vincent,y) \wedge husband(y) \wedge of(y,Mia)$$

paraphrased as something like 'Vincent is a person who is playing the role of Mia's husband'.

(NB Bos attributes the idea to Yuliya Lierler and Vladimir Lifschitz)

I don't find this satisfactory

- Compositionality: where do 'role' and 'person' come from?
- Blocks unwanted inference, but does not support a valid inference: If Vincent is Mia's husband, then Vincent is a husband: $husband(Vincent)$.
- Only obvious way to restore the missing inference would be to add a (second order) axiom:
  $\forall xyP.role(x,y) \wedge P(y) \rightarrow P(x)$
- And not clear how to extend to complex N': John's wooden toy disappeared
  $???\exists x.thing(x) \wedge \exists y.role(x,y) \wedge wooden(y) \wedge toy(y)...$
- We want to infer 'Something wooden disappeared', but can roles be wooden?

Suggestion 1: encode (some) HOL as FOL

Two examples we want to capture:

Bill is John's dentist $\models$ Bill is a dentist

$of(Bill,John,dentist) \models dentist(Bill)$

John's wooden toy disappeared $\models$ A toy/ John's toy disappeared

$\exists x.of(x,John,\lambda y.(wooden(y) \wedge toy(y))) \wedge disappeared(x) \models$

$\exists x.toy(x) \wedge disappeared(x)$

$\exists x.toy(x) \wedge of(x,John,toy) \wedge disappeared(x)$

Axioms + beta-reduction:

A: $\forall xyP.of(x,y,P) \rightarrow P(x)$

B: $\forall xyPQ.of(x,y,\lambda z.P(z) \wedge Q(z)) \rightarrow of(x,y,P) \wedge of(x,y,Q)$ etc.

(It would be nice if someone added these (not very) higher order features to a FOL theorem prover. But in the meantime...)

Encoding: the gory details[1]

Represent literals in applicative form:

$sleep(john) = p(a(sleep,john))$

$like(john,jane) = p(a(a(like,john),jane))$

Now we can represent predicate variables: $P(j) = p(a(P,j))$

What's the 'p' doing? Well, 'a' is actually a function symbol, so to respect FOL syntax we wrap a dummy predicate 'p' around the translation.

$$of(x,y,\lambda z.P(z) \wedge Q(z)) = p(a(a(a(of,x),y),\lambda z.a(a(and,a(P,z)),a(Q,z)))))$$

Now eliminate lambda expressions by using combinators:

Ix	=	x
Kxy	=	x
Sxyz	=	xz(yz)
Cfxy	=	fyx
Bfgx	=	f(gx)

[1] The basic idea is taken from [Hurd, 2002]

FOL Encoding continued...

Define function T(ranslate), lambda-term to combinators:

$T[x] \Rightarrow x$
$T[(E1\ E2)] \Rightarrow (T[E1]\ T[E2])$
$T[\lambda x.E] \Rightarrow (K\ T[E])$ *(if x is not free in E)*
$T[\lambda x.x] \Rightarrow I$
$T[\lambda x.\lambda y.E] \Rightarrow T[\lambda x.T[\lambda y.E]]$ *(if x is free in E)*
$T[\lambda x.(E1\ E2)] \Rightarrow (S\ T[\lambda x.E1]\ T[\lambda x.E2])$ *(if x is free in both E1 and E2)*
$T[\lambda x.(E1\ E2)] \Rightarrow (C\ T[\lambda x.E1]\ T[E2])$ *(if x is free in E1 but not E2)*
$T[\lambda x.(E1\ E2)] \Rightarrow (B\ T[E1]\ T[\lambda x.E2])$ *(if x is free in E2 but not E1)*
$T[\lambda x.(E\ x)] \Rightarrow T[E]$ *(if x is not free in E: this is eta reduction)*

Now our axioms look like this:

A: $p(a(a(a(of,X),Y),Q)) \rightarrow p(a(Q,X))$
B: $p(a(a(a(of,X),Y),a(a(S,a(a(B,and),Q)),R))) \rightarrow$
$p(a(a(a(of,X),Y),Q)) \wedge p(a(a(a(of,X),Y),R))$

Capturing our inferences

A: $p(a(a(a(of,X),Y),Q)) \rightarrow p(a(Q,X))$
B: $p(a(a(a(of,X),Y),a(a(S,a(a(B,and),Q)),R))) \rightarrow$
$p(a(a(a(of,X),Y),Q)) \wedge p(a(a(a(of,X),Y),R))$

$of(Bill,John,dentist) = p(a(a(a(of,John),Bill),dentist))$
Using axiom A we deduce $p(a(dentist,Bill)) = dentist(Bill)$

John's wooden toy disappeared =
$\exists x.of(x,John,\lambda y.(wooden(y) \wedge toy(y))) \wedge disappeared(x)$
Using axiom B we can deduce $...of(x,John,toy)...$ and from A $...toy(x)...$
enabling us to prove the queries:
$\exists x.toy(x) \wedge disappeared(x)$
$\exists x.toy(x) \wedge of(x,John,toy) \wedge disappeared(x)$

Adjective inferences

We can encode our adjective inferences in the same way:

$$\forall xP.\ small(x,P) \rightarrow P(x) \Rightarrow$$
$$p(a(a(small,X),P)) \rightarrow p(a(P,X))$$
$$\forall xPQ.\ small(x,\lambda y.P(y) \wedge Q(y)) \rightarrow P(x) \wedge Q(x) \Rightarrow$$
$$p(a(a(small,X),a(a(S,a(a(B,and),a(a(B,P),I))),a(a(B,Q),I)))) \rightarrow$$
$$p(a(a(and,a(P,X)),a(Q,X)))$$

These axioms and others will enable us to capture inferences like:
Jones is a short Welsh rugby-player

- $\models$ Jones is Welsh
- $\models$ Jones is a rugby-player
- $\models$ Jones is not a tall Welsh rugby-player

(does it follow that Jones is not a tall rugby player? Strictly speaking, no!)

Works, partly, but clumsy...

All these examples work (NB translations and proofs tested with Prover9). But I find this method is clumsy:

- Logical form $\Rightarrow$ Applicative form $\Rightarrow$ Combinator form $\Rightarrow$ Clausal form
- To interpret the results we often need to partly reverse the process (e.g. for wh-answers that may be complex functions)
- Probably inefficient on a large scale (predicates all the same, deeply nested functions)
- And not complete: note that in the final form of the literals we still have logical connectives.
- We have to axiomatise the inferences associated with connectives inside lambda terms...

Suggestion 2: second order axioms as schemata

Partly higher order logical form $\Rightarrow$ Forward chaining schemata $\Rightarrow$
Expanded set of first order LFs $\Rightarrow$ Theorem prover

Reinterpret axioms as rewriting schemata: match to input LF using higher order matching, then beta-reduce results:

$\forall xP.\ small(x,P) \rightarrow P(x) \Rightarrow$

A: $small(x,P) \Rightarrow small(x,hash(P)) \wedge P(x)$

'hash' is something which produces a unique symbol of type e for its argument (same arg, same symbol).

Harvard is an old American university

$old(harvard,\ \lambda y.\ american(y) \wedge university(y)) \Rightarrow$ (via A)

$old(harvard,AU) \wedge [\lambda y.\ american(y) \wedge university(y)](harvard)$

$old(harvard,AU) \wedge american(harvard) \wedge university(harvard)$

And via our earlier axiom, which can now be interpreted as a first order one:

$old(X,P) \rightarrow \neg young(X,P) \models \quad \ldots \neg young(harvard,AU)$

Combining possessive and adjective inferences

Our main axiom for possessives is now:

B: $of(x,y,P) \Rightarrow of(x,y,hash(P)) \wedge P(x)$

This interacts with axiom A for adjectives, and we have to recursively apply these rewritings:

John's old wooden toy broke =

$\exists x.of(x,john, \lambda y.old(y, \lambda z.wooden(z) \wedge toy(z))) \wedge broke(x)$

via B:

$\exists x.of(x,john,OWT) \wedge [\lambda y.old(x,\lambda z.wooden(z) \wedge toy(z))](x) \wedge broke(x)$

$\exists x.of(x,john,OWT) \wedge old(x,\lambda z.wooden(z) \wedge toy(z)) \wedge broke(x)$

via A:

$\exists x.of(x,john,OWT) \wedge old(x,WT) \wedge [\lambda z.wooden(z) \wedge toy(z)](x) \wedge broke(x)$

$\exists x.of(x,john,OWT) \wedge old(x,WT) \wedge wooden(x) \wedge toy(x) \wedge broke(x)$

Connectives again

These schemata will still not allow us to capture inferences like

John's wooden toy broke $\models$ John's toy broke

We need an axiom to the effect that if $of(x,y,P)$ and if property P entails property Q, then $of(x,y,Q)$. Recall that in our combinator treatment we had a special case of such an axiom:

$\forall xyPQ.of(x,y,\lambda z.P(z) \wedge Q(z)) \rightarrow of(x,y,P) \wedge of(x,y,Q)$ etc.

While this will generalise to multiple conjunctions, it will not handle analogous cases with disjuncts:

John's mother or father phoned

$\models$ John's mother or John's father phoned

What we really need is to frame our axioms as schemata that are able to quantify over more than second order.

Go even higher order?

Intuitively, what we need to do is to 'raise' the internal logical structure of a second-order lambda-term argument over the propositional level.
Schematically:
Cn: $of(x,y,\lambda z.P(z)\ C\ Q(z)) \Rightarrow of(x,y,P)\ C\ of(x,y\ Q)$
('C' can be $\wedge$ or $\vee$ - but linguistically (I think!) we will never get $\rightarrow$).
Smith's mother or father phoned

$\exists x.of(x,smith,\lambda y.mother(y) \vee\ father(y)) \wedge\ phoned(x)$

via Cn: $\exists x.(of(x,smith,mother) \vee\ of(x,smith,father)) \wedge\ phoned(x)$
Now we can handle our earlier example:

$\exists x.of(x,john,\ \lambda z.wooden(z) \wedge\ toy(z))) \wedge\ broke(x)$

by B: $\exists x.of(x,john,\ WT) \wedge\ wooden(x) \wedge\ toy(x) \wedge\ broke(x)$
by Cn also: $\exists x.of(x,john,\ wooden) \wedge\ of(x,john,toy) \wedge\ broke(x)$
by B: $\exists x.of(x,john,W) \wedge\ of(x,john,T) \wedge\ wooden(x) \wedge\ toy(x) \wedge\ broke(x)$

which $\models$ John's toy broke $= \exists x.of(x,john,T) \wedge\ toy(x) \wedge\ broke(x)$

Conclusions

- There are some simple but central NL constructs that seem to need second order inference
- It may be possible to reduce to first order via reification, but I haven't been able to
- Translation to FOL via combinators may work, but is a little clumsy and may not be possible in full generality
- A better approach seems to be to pre-process the higher order LFs using second (or higher) order matching, rewriting in a forward-chaining manner to add extra first-order LFs
- Health warning: I haven't implemented the second (rewriting) approach yet, so there is sure to be some problem I haven't foreseen...

References

Bos, J. (2009).
Computing genitive superlatives.
In *Proceedings of the Eighth International Conference on Computational Semantics*, IWCS-8 '09, pages 18–32, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bos, J., Clark, S., Steedman, M., Curran, J. R., and Hockenmaier, J. (2004).
Wide-coverage semantic representations from a CCG parser.
In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*, pages 1240–1246, Geneva, Switzerland.

Hurd, J. (2002).
An LCF-style interface between HOL and first-order logic.
In Voronkov, A., editor, *Automated Deduction - CADE-18, 18th International Conference on Automated Deduction, Copenhagen, Denmark, July 27-30, 2002, Proceedings*, volume 2392 of *Lecture Notes in Computer Science*, pages 134–138. Springer.

Partee, B. H. and Borschev, V. (2003).
Genitives, relational nouns, and argument-modifier ambiguity.
In E. Lang C. Maienborn, C. F.-H., editor, *Modifying Adjuncts*, Interface Explorations, pages 67–112. Mouton de Gruyter, Berlin.

Peters, S. and Westerståhl, D. (2006).
Quantifiers in Language and Logic.
Clarendon Press, Oxford.

Steedman, M. (2012).
Taking Scope.
MIT Press.