Comp24412 Symbolic AI
Lecture 16: Natural Language Semantics III

Ian Pratt-Hartmann

Room KB2.38
e-mail: ipratt@cs.man.ac.uk

2016–17
In a previous lecture, we saw how to parse CPs.

But what about the semantics?

Consider the following example:

Every boy who kissed Mary kissed Jane

The semantics should be

$$\forall x (\text{boy}(x) \land \text{kissed}(x, \text{mary}) \rightarrow \text{kissed}(x, \text{jane}))$$
• Recall the deep structure:
We have two jobs:

- Explain the contribution made by relative pronouns (e.g. who, which)
- Provide augmented grammar rules for C and its projections C' and CP
• A relative pronoun is extracted from the C’ where it lives in deep structure.
• So the remaining C’ is, as it were, dependent on the extracted relative pronoun
• we will invent an **index** to express this dependency
Pictorially
• Thus, we propose the meaning assignment for who:

\[[\text{NP who}] = \lambda p[p(x)]\]

where \(x\) is the index of the relative pronoun

• Note, \(x\) is an unquantified free variable. It is an item dependent on the context in which it occurs.
• We also propose the augmented grammar rules:
  \[ N'/\varphi(\psi) \rightarrow N'/\psi, \ CP/\varphi \]
  \[ CP/\lambda p[\lambda x[p(x) \land \varphi]] \rightarrow CP-Spec_x, \ C'/\varphi \]
  \[ C'/\varphi \rightarrow C, \ IP/\varphi \]

• Note: the variable \( x \) is the trace (or index) belonging to the antecedent of the relative pronoun.
This enables us to compute the following deep structure:
• All we need to do now is express this in Prolog

• The relevant bits of the NP-rules are:

\[ np(NPSem, MvdNPL, MvdNPL) \rightarrow det(DetSem), nbar(NbarSem), \]
\[ \{ \text{var} \_ \text{replace}(DetSem, DetSem1), \]
\[ \beta(\text{DetSem1} @ NbarSem, NPSem) \}. \]

\[ np(MvdNP, [MvdNP], []) \rightarrow []. \]

\[ nbar(NbarSem) \rightarrow n(\text{NSem}), cp(CPSem), \]
\[ \{ \text{var} \_ \text{replace}(CPSem, CPSem1), \]
\[ \beta(\text{CPSem1} @ \text{NSem}, NbarSem) \}. \]

\[ nbar(\text{NSem}) \rightarrow n(\text{NSem}). \]

• Note how, in the second \textit{np} rule, the movement list gets popped without consuming any of the input’s words
The CP-rules are:

\[
\begin{align*}
\text{cp}(lbd(p, lbd(\text{Ind}, (p@\text{Ind} & \text{CbarSem})))) &\rightarrow c\text{Spec}, \\
&\{\text{gensym}(x, \text{Ind})\}, \\
&\text{cbar}(\text{CbarSem}, [lbd(q, q@\text{Ind})]).
\end{align*}
\]

\[
\begin{align*}
c\text{Spec} &\rightarrow [\text{RelPro}], \\
&\{\text{isRelPro}(\text{RelPro})\}.
\end{align*}
\]

\[
\begin{align*}
c\text{bar}(\text{IPSem}, \text{MvdNPL}) &\rightarrow c, \\
&\text{ip}(\text{IPSem}, \text{MvdNPL}).
\end{align*}
\]

\[
c\rightarrow [\].
\]
• The IP-rules need to be augmented to take a possible MvdNP List for when the IP is a relative clause:

\[
\text{ip(SSem, MvdNPL)} \rightarrow \text{np(NPSem, MvdNPL, MvdNPL1), i(bar(IbarSem, MvdNPL1)}, \\
\{\text{var\_replace(NPSem, NPSem1),}
\beta(NPSem1@IbarSem, SSem)\}\}.
\]

\[
\text{i(bar(VPSem, MvdNPL)} \rightarrow \text{i(MvdVbL), vp(VPSem, MvdVbL, MvdNPL)}.
\]

\[
\text{i([])} \rightarrow \text{[Aux], isAux(Aux).}
\]

\[
\text{i([Verb])} \rightarrow \text{[InflVerb], pastInfl(Verb, InflVerb), isVerb(Verb).}
\]
To deal with object relative pronouns we need to augment the VP-rules to pass the MvdNP list down to the object NP:

\[
vp(V\text{barSem},MvdVbL,MvdNPL) \rightarrow v(V\text{barSem},MvdVbL,MvdNPL).
\]

\[
v(V\text{barSem},MvdVbL,MvdNPL) \rightarrow v(VSem,MvdVbL), np(NPSem,MvdNPL,[]), \\
\{\text{var	extunderscore}\text{replace}(VSem,VSem1), \\
\beta(VSem1@NPSem,V\text{barSem})\}.
\]

\[
v(lbd(s,lbd(x,s@lbd(y,Fla))),[]) \rightarrow [\text{Verb}], \\
\{\text{isVerb}(\text{Verb}),Fla=..[\text{Verb},x,y]\}.
\]

\[
v(lbd(s,lbd(x,s@lbd(y,Fla))),[MvdVb]) \rightarrow [] \\
\{\text{isVerb}(MvdVb),Fla=..[MvdVb,x,y]\}.
\]
• All we need now is a calling predicate
  \[
  \text{ip}(\text{Sentence}) :- \\
  \text{ip}(\text{SSem}, [], \text{Sentence}, []), \text{printf}(\text{SSem}).
  \]
• Note the extra argument!
• Seems to work
  :- ip([every,boy,who,kissed,mary,kissed,jane]).

  (x31)((boy(x31) &
        kiss(x31, mary)) ->
     kiss(x31, jane))

• Changing fonts, this is:
  \[ \forall x_{31}((\text{boy}(x_{31}) \land \text{kissed}(x_{31}, \text{mary})) \rightarrow \text{kissed}(x_{31}, \text{jane})) \]

• Whew!