LING 388: Language and Computers

Sandiway Fong

Lecture 20
Administrivia

• Next Tuesday is Veterans Day
  – no class
Last Time

• Updated g18.pl to g19.pl

• New:
  1. progressive *be*
      • the man was/*were eating the sandwich
  2. progressive + (passive) + main verb
      • The sandwich was being eaten
      • *The sandwich was been eating* (passive + progressive)
Last Time

• New:
  2. progressive + (passive) + main verb
     • The sandwich was being eaten
     • *The sandwich was been eating  (passive + progressive)

Parse

```plaintext
(S
  (NP (DT The) (NN sandwich))
  (VP (VBD was)
    (VP (VBN been)
      (VP (VBG eating))))))
```

available parse with the Stanford parser
Last Time

?- s(Parse, [the, man, was, being, eating, the, sandwich], []).  
   false.

?- s(Parse, [the, man, was, eating, the, sandwich], []).  
   Parse = s(np(dt(the), nn(man)), vp(vbd(was), vp(vbg(eating), np(dt(the), nn(sandwich))))));  
   false.

?- s(Parse, [the, sandwich, was, being, eaten], []).  
   Parse = s(np(dt(the), nn(sandwich)), vp(vbd(was), vp(vbg(being), vp(vbn(eaten), np(0))))));  
   false.
Last Time

• New contd.
  • iterated progressive *be* blocked

Rule introducing progressive *be*
permitted iteration:

$vp \rightarrow \text{v_aux, } vp.$
Last Time

• New contd.
  • iterated progressive *be* blocked

Parse

```
(ROOT
  (S
    (NP (DT the) (NN man))
    (VP (VBD was)
      (VP (VBG being)
        (S
          (VP (VBG eating)
            (NP (DT the) (NN sandwich)))))
    (VP (VBG being)
      (S
        (NP (DT the) (NN sandwich))))
    (NP (DT the) (NN man))
  (VP (VBD was)
    (VP (VBG being)
      (S
        (VP (VBG eating)
          (NP (DT the) (NN sandwich)))))
```

available parse with the Stanford parser
SBAR and S rules:

\[
\text{\texttt{sbar}(sbar(C,S)) \rightarrow complementizer(C), s(S)}. \\
\text{\texttt{objrel}_s\texttt{bar}(sbar(C,S)) \rightarrow complementizer(C), objrel}_s\texttt{s}(S)}. \\
\text{\texttt{subjrel}}_s\texttt{bar}(sbar(C,S), \text{Number, Person}) \rightarrow \\
\quad \text{complementizer}(C), \text{subjrel}_s(s(S), \text{Number, Person}). \\
\]

\[
\text{\texttt{s}(s(NP, VP)) \rightarrow np(NP, Person, Number), vp(VP, Tag, Number), \\
\quad \{\text{check(Person, Number, Tag)}\}.} \\
\text{\texttt{s}(s(NP, VP)) \rightarrow np(NP, Person, Number), vp\_passive(VP, Tag, Number), \\
\quad \{\text{check(Person, Number, Tag)}\}.} \\
\text{\texttt{objrel}_s}(s(NP, VP)) \rightarrow \\
\quad np(NP, Person, Number), \text{objrel}_v\texttt{p}(VP, Tag, Number), \\
\quad \{\text{check(Person, Number, Tag)}\}. \\
\text{\texttt{subjrel}_s}(s(NP, VP), \text{Number, Person}) \rightarrow \\
\quad \text{empty}_n\texttt{p}(NP), \text{vp}(VP, Tag, Number), \{\text{check(Person, Number, Tag)}\}. \\
\]

need to call \texttt{vp\_main} too!
VP, progressive *be* and passive *be* rules:

\[
\begin{align*}
\text{vp}(\text{vp}(V,\text{VP}),\text{Tag},\text{Number}) & \rightarrow \text{v}_\text{aux}(V,\text{Tag},\text{Number}), \text{vp}_\text{main}(\text{VP},\text{vbg},\_). \\
\text{vp}(\text{vp}(V,\text{VP}),\text{Tag},\text{Number}) & \rightarrow \\
& \quad \text{v}_\text{aux}(V,\text{Tag},\text{Number}), \text{vp}_\text{passive}(\text{VP},\text{vbg},\_). \\
\text{vp}(\text{vp}(V,\text{NP}),\text{Tag},\_)& \rightarrow \text{verb}(V,\text{Tag}), \text{np}(\text{NP},\_,\_). \\
\text{objrel}_\text{vp}(\text{vp}(V,\text{NP}),\text{Tag},\_)& \rightarrow \text{verb}(V,\text{Tag}), \text{empty}_\text{np}(\text{NP}). \\
\text{vp}_\text{main}(\text{vp}(V,\text{NP}),\text{Tag},\_)& \rightarrow \text{verb}(V,\text{Tag}), \text{np}(\text{NP},\_,\_). \\
\text{vp}_\text{passive}(\text{vp}(\text{Aux},\text{vp}(V,\text{NP})),\text{Tag},\text{Number}) & \rightarrow \\
& \quad \text{v}_\text{aux}(\text{Aux},\text{Tag},\text{Number}), \text{verb}(V,\text{vbn}), \text{empty}_\text{np}(\text{NP}).
\end{align*}
\]
lexical entries for auxiliary verb *be*:

```prolog
% v_aux(Parse,Tag,Number)
v_aux(vbd(was),vbd,singular) --> [was].
v_aux(vbd(were),vbd,plural) --> [were].
v_aux(vbz(is),vbz,singular) --> [is].
v_aux(vb(are),vb,_ ) --> [are].
v_aux(vbg(being),vbg,_ ) --> [being].
v_aux(vbn(ben),vbn,_ ) --> [been].
```
NP rules:

\[
\begin{align*}
\text{np}(\text{np}(\text{DT}, \text{NN}), \text{Person}, \text{Number}) & \rightarrow \text{dt}(\text{DT}, \text{Number}), \text{nn}(\text{NN}, \text{Number}, \text{Person}). \\
\text{np}(\text{np}(\text{np}(\text{DT}, \text{NN}), \text{SBAR}), \text{Person}, \text{Number}) & \rightarrow \\
& \text{dt}(\text{DT}, \text{Number}), \text{nn}(\text{NN}, \text{Number}, \text{Person}), \text{objrel_sbar}(\text{SBAR}). \\
\text{np}(\text{np}(\text{np}(\text{DT}, \text{NN}), \text{SBAR}), \text{Person}, \text{Number}) & \rightarrow \\
& \text{dt}(\text{DT}, \text{Number}), \text{nn}(\text{NN}, \text{Number}, \text{Person}), \\
& \text{subjrel_sbar}(\text{SBAR}, \text{Number}, \text{Person}).
\end{align*}
\]

% Person and Number features determined elsewhere
empty_np(np(\emptyset)) \rightarrow [].
g19.pl

Lexicon:

```
complementizer(c(that)) --> [that].
dt(dt(the),_) --> [the].
dt(dt(a),singular) --> [a].
nn(nn(sandwich),singular,3) --> [sandwich].
nn(nn(sandwich+pl),plural,3) --> [sandwiches].
nn(nn(man),singular,3) --> [man].
nn(nn(man+pl),plural,3) --> [men].
nn(nn(rat),singular,3) --> [rat].
nn(nn(cat),singular,3) --> [cat].

nn(nn(Root-Suffix),Number,Person) --> [Word],
{atom_concat(Root,Suffix,Word),
suffix(Number,[Suffix],[[]]),
nn(_,singular,Person,[Root],[[]]),
\+ nn(nn(Root+pl),plural,_,_,_)}.

nn(nn(cheese),mass,3) --> [cheese].
suffix(plural) --> [s].
```
Lexicon:

verb(vb(see),vb) --> [see].
verb(vbz(sees),vbz) --> [sees].
verb(vbg(seeing),vbg) --> [seeing].
verb(vbd(saw),vbd) --> [saw].
verb(vbn(seen),vbn) --> [seen].

verb(vb(eat),vb) --> [eat].
verb(vbz(eats),vbz) --> [eats].
verb(vbg(eating),vbg) --> [eating].
verb(vbd(ate),vbd) --> [ate].
verb(vbn(eaten),vbn) --> [eaten].

verb(vb(chase),vb) --> [chase].
verb(vbz(chases),vbz) --> [chases].
verb(vbg(chasing),vbg) --> [chasing].
verb(vbd(chased),vbd) --> [chased].
verb(vbn(chased),vbn) --> [chased].
g19.pl

Subject-Verb agreement: Person, Number and Verb Inflection

% table of Person Number Tag possible combinations
check(3,plural,vb).
check(3,singular,vbz).
check(3,mass,vbz).
check(_,_,vbd).
Today's Topic

• Adding perfective *have*:

1. the man ate the sandwich (simple past)
2. the man had eaten the sandwich (perfective)
3. the man had been eating the sandwich (perfective + progressive)
4. the sandwich is eaten (passive)
5. the sandwich was eaten (passive)
6. the sandwich had been eaten (perfective + passive)
7. *the sandwich was had eaten (passive + perfective)
8. the sandwich had been being eaten (perfective + progressive + passive)
9. *the man was having eaten the sandwich  (progressive + perfective)
10. the man has been eating the sandwich (perfective + progressive)
11. the man had been eating the sandwich (perfective + progressive)
Today's Topic

• Rule: perfective *have* followed by *V-en* (vbn):
  1. the man had eaten the sandwich (perfective)
  2. the man had been eating the sandwich (perfective + progressive)
  3. the man has been eating the sandwich (perfective + progressive)
  4. the sandwich had been eaten (perfective + passive)
  5. the sandwich had been being eaten (perfective + progressive + passive)

Let's modify g19.pl ...