LING 388: Language and Computers

Sandiway Fong

Lecture 12
Adminstrivia

• Review:
  – Homework 4 on Recursion
  – Continue on with the left to right recursion grammar transformation:
    • Last Time: abstract grammar fragment
    • Today: PP adjunction to NP and VP
Homework 4 Review

• Recursion type 1:
  – we can stack adjectives...
  – the bus
  – the big bus
  – the big red bus
  – the shiny big red bus
  – (cf. the big shiny red bus)

• Recursive rule:
  – right recursive rule
  – nn --> a, nn.
Start with grammar.pl from the course webpage:

- \( \text{nn(nn(A,NN))} \rightarrow \text{a(A)}, \text{nn(NN)}. \)
- \( \text{a(jj(big))} \rightarrow \text{[big]}. \)
- \( \text{a(jj(shiny))} \rightarrow \text{[shiny]}. \)
- \( \text{a(jj(red))} \rightarrow \text{[red]}. \)
- \( \text{nn(nn(bus))} \rightarrow \text{[bus]}. \)

sentence(s(X,Y)) \rightarrow \text{np}(X), \text{vp}(Y).

\( \text{pp(pp(X,Y))} \rightarrow \text{in}(X), \text{np}(Y). \)

\( \text{in(in(with))} \rightarrow \text{[with]}. \)

\( \text{np(np(X))} \rightarrow \text{prp}(X). \)

\( \text{np(np(np(X,Y),Z))} \rightarrow \text{det}(X), \text{nn}(Y), \text{pp}(Z). \)

\( \text{np(np(D,NN))} \rightarrow \text{det}(D), \text{nn}(NN). \)

\( \text{prp(prp(i))} \rightarrow \text{[i]}. \)

\( \text{prp(prp(me))} \rightarrow \text{[me]}. \)

\( \text{nn(nn(boy))} \rightarrow \text{[boy]}. \)

\( \text{nn(nn(telescope))} \rightarrow \text{[telescope]}. \)

\( \text{vp(vp(V,X))} \rightarrow \text{verb}(V), \text{np}(X). \)

\( \text{vp(vp(V,X,Y))} \rightarrow \text{verb}(V), \text{np}(X), \text{pp}(Y). \)

\( \text{verb(vbd(saw))} \rightarrow \text{[saw]}. \)

\( \text{det(dt(the))} \rightarrow \text{[the]}. \)

\( \text{det(dt(a))} \rightarrow \text{[a]}. \)
Homework 4 Review

- Tree representation:

```
?- np(Parse,[the,bus],[[]]).
Parse = np(dt(the), nn(bus));
false.
```
Homework 4 Review

• Tree representation:

```prolog
?- np(Parse, [the, bus], []).
Parse = np(dt(the), nn(bus)) ;
false.

?- np(Parse, [the, big, bus], []).
Parse = np(dt(the), nn(jj(big), nn(bus))) ;
false.
```
Homework 4 Review

- Tree representation:

```prolog
?- np(Parse, [the, bus], []).
Parse = np(dt(the), nn(bus)) ;
false.

?- np(Parse, [the, big, bus], []).
Parse = np(dt(the), nn(jj(big), nn(bus))) ;
false.

?- np(Parse, [the, big, red, bus], []).
Parse = np(dt(the), nn(jj(big), nn(jj(red), nn(bus)))) ;
false.
```
Homework 4 Review

• Tree representation:
Homework 4 review

• Recursion is a property of natural language
  – simple iteration: ... *big shiny red bus*

• Another kind of recursion
  – Some verbs can select for clauses (as well as object NPs), e.g. *notice*
    • I noticed that John noticed that Mary noticed the big red bus
    • [\(S\) I noticed [\(SBAR\) that [\(S\) John noticed [\(SBAR\) that [\(S\) Mary noticed the big red bus]]]]]

**Idea:** some verbs can select for sentences introduced by the complementizer *that*
Homework 4 review

• Example:
  – Mary noticed the big red bus

  1. verb(vbd(noticed)) --> [noticed].
  2. np(np(X)) --> proper_noun(X).
  3. proper_noun(nnp(mary)) --> [mary].

?- sentence(Parse, [mary, noticed, the, big, red, bus], []).
Parse = s(np(nnp(mary)), vp(vbd(noticed), np(dt(the), nn(jj(big), nn(jj(red), nn(bus))))));
false.
• Example:
  – John noticed that Mary noticed the big red bus

1. vp(vp(V,SBAR)) --> verb(V), sbar(SBAR).
2. sbar(sbar(C,S)) --> c(C), sentence(S).
3. c(c(that)) --> [that].
4. proper_noun(nnp(john)) --> [john].
Homework 4 review

• Example:
  – I noticed that John
    noticed that Mary
    noticed the big red bus

?- sentence(Parse, [i, noticed, that, john, noticed, that, mary, noticed, the, big, red, bus], []).
Parse = s(np(prp(i)), vp(vbd(noticed), sbar(c(that), s(np(nnp(john)), vp(vbd(noticed)), sbar(c(that), s(np(nnp(mary)), vp(vbd(noticed)), np(dt(...), nn(..., ...)))))))) w
Parse = s(np(prp(i)), vp(vbd(noticed), sbar(c(that), s(np(nnp(john)), vp(vbd(noticed), sbar(c(that), s(np(nnp(mary)), vp(vbd(noticed)), np(dt(the), nn(jj(big), nn(jj(red), nn(bus)))))))))))
false.
Homework 4 review

• Example:
  – *I noticed that John noticed that Mary noticed the big red bus*
Today’s Topic

• Exercise:
  – Convert left recursive natural language grammar rules into right recursive grammar rules
Step 1

• Recipe:
  1. Let $x$ be the nonterminal with the left recursion.
  2. Let $z$ be the terminal (or nonterminal) sequence that $x$ also expands to.
  3. Let $y$ be the terminal (and/or nonterminal) sequence after the left recursive call.

• Abstract example from last time:
  - $x \rightarrow x, [y]$.
  - $x \rightarrow [z]$. 
Step 1

• Part 1:
  – identify x, y, and z in the following rules:
  1. np → dt, nn.
  2. np → np, pp.
  
  – identify x, y, and z in the following rules:
  3. vp → vbd, np.
  4. vp → vp, pp.

  \[ x \rightarrow x, \ [y]. \]
  \[ x \rightarrow [z]. \]
Step 2

• Transformation:
  - \( x \rightarrow x, [y] \).
  - \( x \rightarrow [z] \).
  
  into
  - \( x \rightarrow [z], v \).
  - \( v \rightarrow [y], v \).
  - \( v \rightarrow [y] \).
  - \( x \rightarrow [z] \).

• Step 2:
  - apply transformation to:
  - \( np \rightarrow dt, nn \).
  - \( np \rightarrow np, pp \).
  
  also to:
  - \( vp \rightarrow vbd, np \).
  - \( vp \rightarrow vp, pp \).
Step 3

- Step 3:
  - modify the grammar fragments from step 2 into grammars that compute parse trees
  - test your grammar fragments on VPs like *saw a boy with a telescope*
  - $\text{vp}(\text{Parse}, \text{[saw, a, boy, with, a, telescope]}, [])$.

- Abstract example:
  - $x \rightarrow [z], v$.
  - $v \rightarrow [y], v$.
  - $v \rightarrow [y]$.
  - $x \rightarrow [z]$.
  - modified version:
    - $x(x(z, V)) \rightarrow [z], v(V)$.
    - $v(v(y, V)) \rightarrow [y], v(V)$.
    - $v(v(y)) \rightarrow [y]$.
    - $x(x(z)) \rightarrow [z]$. 
Comparison

• “Quick Fix” grammar from last time:

\[
\begin{align*}
\text{pp(pp(IN,NP))} & \rightarrow \text{in(IN), np(NP)}. \\
\text{np(np(DT,NN))} & \rightarrow \text{dt(DT), nn(NN)}. \\
\text{np(np(np(DT,NN),PP))} & \rightarrow \text{dt(DT), nn(NN), pp(PP)}. \\
\text{in(in(with))} & \rightarrow \text{[with]}. \\
\text{dt(dt(a))} & \rightarrow \text{[a]}. \\
\text{nn(nn(telescope))} & \rightarrow \text{[telescope]}. \\
\text{nn(nn(limp))} & \rightarrow \text{[limp]}. \\
\text{nn(nn(boy))} & \rightarrow \text{[boy]}. 
\end{align*}
\]

• Result:

?- np(Parse,[a,boy,with,a,telescope],[[]]).
Parse = np(np(dt(a), nn(boy)), pp(in(with), np(dt(a), nn(telescope))))

?- np(Parse,[a,boy,with,a,telescope,with,a,limp],[[]]).
Parse = np(np(dt(a), nn(boy)), pp(in(with), np(np(dt(a), nn(telescope)), pp(in(with), np(dt(a), nn(limp))))));
false.
Step 4

• Step 4:
  – Compare the transformed grammar parses with those obtained using the “Quick Fix” grammar on the NP
    • a boy with a telescope with a limp
    • number of parses?
    • attachment of the PPs?