11-711: Algorithms for NLP

Recitation #7

October 30, 2008

Chart Parsing

Here is a small grammar and lexicon for a fragment of English:

```
Grammar:                          Lexicon:

  (1)  S --> NP VP                  I:         N
  (2) NP --> DET N                  can:       N, AUX, V
  (3) NP --> N                      see:       V
  (4) NP --> NP PP                  the:       DET
  (5) VP --> AUX VP                 man:       N, V
  (6) VP --> V NP                   with:      P
  (7) VP --> V                      telescope: N, V
  (8) PP --> P NP
```

1. Construct the set First(A) for each of the non-terminals in the above grammar.
2. Run the chart parsing algorithm with top-down predictions and ambiguity packing on the first n words of the following sentence, where n is defined as the number of words we get through before recitation ends: `I can see the man with the telescope`. Keep backpointers to all created parse fragments and show the resulting parse tree or trees at the end of running the algorithm.

Solution

1. Construct the set First(A) for all non-terminals A. This set represents all the leftmost non-terminals derivable from A and is constructed recursively. We start with First(A) = $\{A\}$:

$$\begin{aligned}
\text{First(S)} &= \{\ \text{S}\ \} \\
\text{First(NP)} &= \{\ \text{NP}\ \} \\
\text{First(VP)} &= \{\ \text{VP}\ \} \\
\text{First(PP)} &= \{\ \text{PP}\ \} \\
\text{First(N)} &= \{\ \text{N}\ \} \\
\text{First(AUX)} &= \{\ \text{AUX}\ \} \\
\text{First(V)} &= \{\ \text{V}\ \} \\
\text{First(DET)} &= \{\ \text{DET}\ \} \\
\text{First(P)} &= \{\ \text{P}\ \}
\end{aligned}$$

Next, for each A, we look at all the grammar rules $A \rightarrow B$ and add B to First(A):

$$
\begin{aligned}
\text{First(S)} &= \{\ \text{S, NP}\ \} \\
\text{First(NP)} &= \{\ \text{NP, DET, N}\ \} \\
\text{First(VP)} &= \{\ \text{VP, AUX, V}\ \} \\
\text{First(PP)} &= \{\ \text{PP, P}\ \} \\
\text{First(N)} &= \{\ \text{N}\ \} \\
\text{First(AUX)} &= \{\ \text{AUX}\ \} \\
\text{First(V)} &= \{\ \text{V}\ \} \\
\text{First(DET)} &= \{\ \text{DET}\ \} \\
\text{First(P)} &= \{\ \text{P}\ \}
\end{aligned}
$$

Finally, we have the recursive step: For each B in First(A), we add First(B) to First(A). This process stops when no changes are made to First(A) for any A. Let's look at the first iteration as an example. Right now, First(S) includes S and NP. First(NP) contains NP, DET, and N, so we add them to First(S) and move on to First(NP). First(NP) is still {NP, DET, N}, so we have to add First(DET) and First(N) to it. These sets are both just themselves, so there is no change to First(NP). The same is true for the rest of the non-terminals — there are no more changes. Thus, after the first iteration, we have:

$$
\begin{aligned}
\text{First(S)} &= \{\ \text{S, NP, DET, N}\ \} \\
\text{First(NP)} &= \{\ \text{NP, DET, N}\ \} \\
\text{First(VP)} &= \{\ \text{VP, AUX, V}\ \} \\
\text{First(PP)} &= \{\ \text{PP, P}\ \} \\
\text{First(N)} &= \{\ \text{N}\ \} \\
\text{First(AUX)} &= \{\ \text{AUX}\ \} \\
\text{First(V)} &= \{\ \text{V}\ \} \\
\text{First(DET)} &= \{\ \text{DET}\ \} \\
\text{First(P)} &= \{\ \text{P}\ \}
\end{aligned}
$$

There was a change to one of the First(A) sets during the first iteration, so we must have a second. First(S) = {S, NP, DET, N}, but First(S) ∪ First(NP) ∪ First(DET) ∪ First(N) is already equivalent to {S, NP, DET, N}, so there are no changes to First(S). There are also no changes to any of the other sets, so the recursive process terminates.

2. Run the chart parsing algorithm. Now we can move on to the main parsing algorithm. It's important to keep track of the way the word indexes are used in chart entries, agenda entries, and active arcs. The position *before* the first word of the input is marked as 1, and the following numbers mark the positions *between* words:

```
  I   can   see   the   man   with   a   telescope
1   2     3     4     5     6      7   8           9
```

Chart parsing is carried out in a series of iterations, one per input word. For each input word, there are four steps: (1) load the agenda; then, for each agenda item, (2) create and extend active arcs, (3) deal with completed active arcs, and (4) add the agenda item to the chart.

Now we begin processing input words. For $i = 1$, we find all parts of speech for the first input word and add them to the agenda. In this case, the word "`I`" has only one possible part of speech (N). The word spans position 1 to position 2, and we include a unique ID number for backpointers.

Agenda	$N_1(1, 2)$
Active Arcs	
Chart	

Now we go through the agenda items. Since we're including top-down predictions, we only want to create active arcs if they would be consistent with these predictions. For the first iteration, that means that we only want to add an arc if it begins with something in First(S). Our only agenda item is $N_1(1, 2)$, and N is in First(S), so we create new arcs for each grammar rule whose right-hand side begins with an N:

Agenda	$N_1(1, 2)$
Active Arcs	[NP → • N](1, 1)
Chart	

We also use the agenda item to extend existing active arcs. We can extend the arc "looking" for an N from position 1 with the N we found from position 1 to position 2.

Agenda	$N_1(1, 2)$
Active Arcs	[NP → • N](1, 1) [NP → N_1 •](1, 2)
Chart	

Notice that we do *not* delete active arcs that get extended! In general, this is important because a single active arc might be extended in multiple ways, all of which can be grammatical.

Now we need to look for completed active arcs — arcs in which the dot has moved to the end of the right-hand side of the grammar rule. We have one, so we enter it into the agenda, including a unique ID number and the backpointers. (Completed active arcs *can* be removed from the list, since they are no longer useful for future arc extensions.)

Agenda	$N_1(1, 2)$ [NP_2 → N_1](1, 2)
Active Arcs	[NP → • N](1, 1)
Chart	

At this point, we've finished processing the first agenda item, so we enter it as a final item in the chart.

Agenda	[NP_2 → N_1](1, 2)
Active Arcs	[NP → • N](1, 1)
Chart	$N_1(1, 2)$

Now we move to the next agenda item, which is an NP from position 1 to 2. First, we create new active arcs by looking for grammar rules whose right-hand sides start with NP. Again, because of the top-down predictions, we only add arcs whose left-hand sides are elements of First(S).

Agenda	[NP_2 → N_1](1, 2)
Active Arcs	[NP → • N](1, 1) [S → • NP VP](1, 1) [NP → • NP PP](1, 1)
Chart	$N_1(1, 2)$

Then we use the agenda item to extend active arcs.

Agenda	[NP_2 → N_1](1, 2)
Active Arcs	[NP → • N](1, 1) [S → • NP VP](1, 1) [NP → • NP PP](1, 1) [S → NP_2 • VP](1, 2) [NP → NP_2 • PP](1, 2)
Chart	$N_1(1, 2)$

There are no completed active arcs that need to be dealt with, so finally we move the agenda item into the chart.

Agenda	
Active Arcs	[NP → • N](1, 1) [S → • NP VP](1, 1) [NP → • NP PP](1, 1) [S → NP_2 • VP](1, 2) [NP → NP_2 • PP](1, 2)
Chart	N_1(1, 2) [NP_2 → N_1](1, 2)

Since the agenda is now empty, we've completed the processing for the first input word.

When $i = 2$, we are now considering the second input word ("`can`"). The word has three possible parts of speech, so we make agenda entries for each of them.

Agenda	N_3(2, 3) AUX_4(2, 3) V_5(2, 3)
Active Arcs	[NP → • N](1, 1) [S → • NP VP](1, 1) [NP → • NP PP](1, 1) [S → NP_2 • VP](1, 2) [NP → NP_2 • PP](1, 2)
Chart	N_1(1, 2) [NP_2 → N_1](1, 2)

We consider the new agenda entries one at a time, starting with N_3. First we create new active arcs from grammar rules whose right-hand sides start with NP, but again we must take into account top-down predictions. For words after the first input word, we only create a new active arc if it's for a constituent that some other active arc might be looking for at the given position. In this case, we are considering an agenda item that starts in position 2, and existing active arcs are looking for VPs and PPs starting in position 2, so we only create new active arcs if their left-hand sides are in First(VP) or First(PP). However, the only grammar rule that starts with an N on the right-hand side is for an NP, which is not in First(VP) or First(PP), so N_3 creates no new active arcs. It also cannot extend any existing active arcs, so it is immediately added to the chart.

Agenda	AUX_4(2, 3) V_5(2, 3)
Active Arcs	[NP → • N](1, 1) [S → • NP VP](1, 1) [NP → • NP PP](1, 1) [S → NP_2 • VP](1, 2) [NP → NP_2 • PP](1, 2)
Chart	N_1(1, 2) [NP_2 → N_1](1, 2) N_3(2, 3)

On the other hand, selecting AUX_4 from the agenda leads to creating an active arc for a VP, which *is* in First(VP), so we do create it. AUX_4 can also extend this new arc, so that happens as well before the item is inserted into the chart.

Agenda	$V_5(2, 3)$
Active Arcs	[NP → • N](1, 1) [S → • NP VP](1, 1) [NP → • NP PP](1, 1) [S → NP_2 • VP](1, 2) [NP → NP_2 • PP](1, 2) [VP → • AUX VP](2, 2) [VP → AUX_4 • VP](2, 3)
Chart	$N_1(1, 2)$ [NP_2 → N_1](1, 2) $N_3(2, 3)$ $AUX_4(2, 3)$

The final agenda item, V_5, creates active arcs for VP → V NP and VP → V. Both of these have left-hand sides in First(VP), so they are consistent with the top-down predictions. Both of the new active arcs will also be immediately extended by V_5 before V_5 is added to the chart.

Agenda	
Active Arcs	[NP → • N](1, 1) [S → • NP VP](1, 1) [NP → • NP PP](1, 1) [S → NP_2 • VP](1, 2) [NP → NP_2 • PP](1, 2) [VP → • AUX VP](2, 2) [VP → • V NP](2, 2) [VP → • V](2, 2) [VP → AUX_4 • VP](2, 3) [VP → V_5 • NP](2, 3) [VP → V_5 •](2, 3)
Chart	$N_1(1, 2)$ [NP_2 → N_1](1, 2) $N_3(2, 3)$ $AUX_4(2, 3)$ $V_5(2, 3)$

V_5 also completed an active arc, so the completed arc becomes a new agenda item itself.

Agenda	[VP_6 → V_5](2, 3)
Active Arcs	[NP → • N](1, 1) [S → • NP VP](1, 1) [NP → • NP PP](1, 1) [S → NP_2 • VP](1, 2) [NP → NP_2 • PP](1, 2) [VP → • AUX VP](2, 2) [VP → • V NP](2, 2) [VP → • V](2, 2) [VP → AUX_4 • VP](2, 3) [VP → V_5 • NP](2, 3)
Chart	$N_1(1, 2)$ [NP_2 → N_1](1, 2) $N_3(2, 3)$ $AUX_4(2, 3)$ $V_5(2, 3)$

The new agenda item VP_6 doesn't create any new active arcs since there are no grammar rules whose right-hand sides begin with VP. It does, however, extend the arc [S → NP_2 • VP](1, 2) and also completes it. VP_6 is then added to the chart, and the completed S arc becomes a new agenda entry. There are no grammar rules whose right-hand side is S, so the S entry creates no new arcs. There are no active arcs looking for an S at position 3, so the entry is immediately added to the chart.

At this point, the agenda is empty, so we've completed processing the second input word.

Agenda	
Active Arcs	[NP → • N](1, 1)
	[S → • NP VP](1, 1)
	[NP → • NP PP](1, 1)
	[S → NP_2 • VP](1, 2)
	[NP → NP_2 • PP](1, 2)
	[VP → • AUX VP](2, 2)
	[VP → • V NP](2, 2)
	[VP → • V](2, 2)
	[VP → AUX_4 • VP](2, 3)
	[VP → V_5 • NP](2, 3)
Chart	N_1(1, 2)
	[NP_2 → N_1](1, 2)
	N_3(2, 3)
	AUX_4(2, 3)
	V_5(2, 3)
	[VP_6 → V_5](2, 3)
	[S_7 → NP_2 VP_6](1, 3)

When $i = 3$, we move to processing "`see`", the third word of the input. The word can only be a verb, so we start with only one agenda entry, V_8(3, 4). Active arcs ending at position 3 are looking for VPs and NPs, and V is in First(VP), so we create new active arcs for grammar rules whose right-hand sides start with V. There are two — the rule VP → V NP and the rule VP → V — so we add them to the list of active arcs. The agenda entry V_8 extends them both. V_8 doesn't extend the other active arcs where the dot is immediately before a V because those arcs are looking for a V at position 2; the agenda item is a V at position 3. V_8 is then inserted into the chart.

Extending the active arc [VP → •V](3, 3) to [VP → V_8 •](3, 4) completes the arc, so it becomes a new agenda entry VP_9. There are no grammar rules with VP starting the right-hand side, so we create no new active arcs. We do, however, extend the arc [VP → AUX_4 •VP](2, 3) looking for a VP at position 3. Extending the arc completes it, so it becomes a new agenda entry VP_{10}. VP_9 goes into the chart.

VP_{10} as an agenda entry starts at position 2. Again, there are no grammar rules that start with VPs on the right-hand side, so no new active arcs are created. VP_{10} extends the arc [S → NP_2 •VP](1, 2) and is then added to the chart. The extension completes the S arc, so it becomes a new agenda entry S_{11}.

There are no grammar rules whose right-hand sides start with S, nor are there any active arcs looking for an S. S_{11} thus goes directly into the chart. The agenda is again empty, so we have completed processing the third word.

The chart at this point is shown below. Notice how partial parses (including some sentence-level parses for portions of the input) and partial parse trees can be extracted from the chart by following the backpointers contained in the entries.

Agenda	
Active Arcs	[NP → • N](1, 1)
	[S → • NP VP](1, 1)
	[NP → • NP PP](1, 1)
	[S → NP_2 • VP](1, 2)
	[NP → NP_2 • PP](1, 2)
	[VP → • AUX VP](2, 2)
	[VP → • V NP](2, 2)
	[VP → • V](2, 2)
	[VP → AUX_4 • VP](2, 3)
	[VP → V_5 • NP](2, 3)
	[VP → V_8 • NP](3, 4)
Chart	N_1(1, 2)
	[NP_2 → N_1](1, 2)
	N_3(2, 3)
	AUX_4(2, 3)
	V_5(2, 3)
	[VP_6 → V_5](2, 3)
	[S_7 → NP_2 VP_6](1, 3)
	V_8(3, 4)
	[VP_9 → V_8](3, 4)
	[VP_{10} → AUX_4 VP_9](2, 4)
	[S_{11} → NP_2 VP_{10}](1, 4)