**General Problem**
- Someone gives you a PCFG $G$
- For any given sentence you might want to:
  - Find the best parse according to $G$
  - Find a bunch of reasonable parses
  - Find the total probability of all parses

**Techniques:**
- Beam search
- Agenda-based search
- The CKY algorithm

**Beam Search**
- State space search
  - States are partial parses
  - Find a way to ensure that all parses of a sentence have the same number $N$ steps
    - Leftmost top-down CFG derivations in CNF
    - Shift-reduce derivations in CNF
    - (Use a binary grammar, or binarize what you've got)

**Kinds of Beam Search**
- Constant beam size $K$
- Constant beam width
  - Additive
  - Multiplicative
- Sometimes do fancier stuff, like try to keep beam diverse
- Beam search can be made very fast
- No measure of how optimal it is
  - Correct hypothesis trick

**Agenda-Based Parsing**
- For general grammars
- Start with a table recording $\delta(X,i,j)$
  - The best score of a parse of $X$ over $[i,j]$
  - All entries start at $\infty$
  - Can be a sparse or dense map
  - Sometimes record backtraces, too
- Step 1: Hit the lexicon
  - For each word $w$, and each tag $t$, set $\delta(t,i,j) = \text{tag-score}(w,t)$
Agenda-Based Parsing

- Keep a list of edges called an agenda
  - Edges are triples \([X,i,j]\)
  - Agenda is a priority queue
- Every time some \(\delta(X,i,j)\) lowers:
  - Stick the edge \([X,i,j]\) into the agenda
  - Update the backtrace for \(\delta(X,i,j)\)

Agenda-Based Parsing

- Step II: While agenda not empty:
  - Get the "next" edge \([X,i,j]\) from the agenda
  - Fetch all compatible neighbors \([Y,j,k]\) or \([Z,k,i]\)
  - Compatible means there are rules \(A \rightarrow XY\) or \(B \rightarrow ZX\)
  - Build parent edges \([A,i,k]\) or \([B,k,j]\)
    - \(\delta(A,i,k) \leq \delta(X,i,j) + \delta(Y,j,k) + P(X|A)\)
    - If we’ve improved \(\delta(A,i,k)\), stick \([A,i,k]\) on the agenda
  - Also project unary rules:
    - When do we know we have a parse for the root?

Open questions:

- Agenda priority: What did “next” mean?
- Efficiency: how do we do as little work as possible?
- Optimality: how do we know when we find the best parse of a sentence?
- If we use \(\delta(X,i,j)\) as the priority:
  - Each edge goes on the agenda at most once
  - When an edge pops off the agenda, its best parse score is known (why?)
  - This is basically uniform cost search

Speeding Up Agenda Parsers

- Two options for doing less work
  - The optimal way: \(A^*\) Parsing
  - The ugly (but possibly faster) way: Best-First Parsing

CKY Parsing

- Assuming:
  - You’ve got a lot of memory
  - You’re willing to do exhaustive parsing
  - Your grammar is in CNF
  - There’s an easy solution: CKY parsing

Next Time

- Grammars beyond PCFGs
- Reading:
  - M+S 11 (over next few classes)
  - J+M 12 (over next few classes)