Practice Exam Problems: Syntax

Natural Language Processing (JHU 601.465/665)

Prof. Jason Eisner

1. [8 points] Draw a parse tree for the sentence below. Think carefully about what **nonterminals** to use. Include **tense** and **number** attributes on the nonterminals that should have such attributes.

`Every child has sometimes felt alone in the world`

2. Last night at a pumpkin-carving party at Charles Commons, my 3-year-old son asked about the recipe for the yummy roasted pumpkin seeds: "What did you mix them with?" The answer he was looking for was "Salt, paprika, garlic powder, and oil." But the resident faculty member instead answered: "This little spoon."

   (a) [5 points] The misunderstanding arose because my son's question was ambiguous. What is the reason for this ambiguity? (*Warning:* Think carefully! Give a sufficiently detailed answer that you can be sure it's right.)

   (b) [15 points] Draw a detailed parse tree that is appropriate for at least one meaning of "What did you mix them with?"

   You will be graded on the (1) structure of your tree; (2) appropriate choices of nonterminals; (3) your use of empty terminal categories and slashed nonterminal categories; (4) your use of appropriate *tense* and *wh* features.

   Note: Parses for specific sentences arise from grammars for the whole language. In designing your parse tree, you must have in mind a grammar that will work for other questions and statements as well! For example, the point of (3) is to allow "What did you mix them with?" without allowing any ungrammatical sentences like "You mixed them with." or "What did you mix them with paprika?" Similarly, the point of (4) is to allow "What did you mix them with?" without allowing any ungrammatical sentences like "Did you mix them with?" or "What you mixed them with?" or "What do you mixed them with?"

`What` `did` `you` `mix` `them` `with`

3. Suppose you are given an English grammar that includes some *unary* nonterminal rules such as these:

   - NP → ProperNoun
   - AdjP → Adj *(where AdjP = adjective phrase)*
   - NP → Adj
   - Adj → Noun
   - CP → S *(where CP = complementizer phrase, e.g.,* `that you love pretzels`*)*

   Let's consider the motivation for these rules, and whether they are wise.

   (a) [2 points] In general, unary rules allow phrases of one type to be "coerced" into use as another type.

   For example, most programming languages allow integers to be coerced to reals, allowing you to write `sqrt(3)` rather than `sqrt(3.0)`. Which rule appears in the grammar of such programming languages? (*circle <u>one</u>*)

   Int → Real Real → Int

   (b) [2 points] Consider this grammatical English sentence:

   - `The boy is proud of his height, but his mother is taller than him.`

   This includes some NP and AdjP constituents. Replace those constituents in a way that is allowed by the rules NP → ProperNoun and AdjP → Adj. That is, write a slightly modified sentence that illustrates the use of those 2 unary rules.

   (c) [3 points] How about the rule NP → Adj? Give a grammatical English sentence that this rule would help to explain.

   (d) [3 points] But perhaps the grammar you got isn't so good. Give an objection to the rule NP → Adj.

(e) [3 points] The rule Adj → Noun allows the noun `flower` to be used as an adjective in

- `flower garden`

This rule appears to be helpful, but is probably a bad idea! Give an argument against it.

(f) [2 points] What is an argument *for* the rule CP → S? Give a grammatical sentence that illustrates its use.

(g) [2 points] Give an argument *against* the rule CP → S.

4. Abbreviated writing is common on Twitter:

> "Took midterm today. Fun problems, but almost fell asleep. Too exhausted from studying!"

The standard-English version would be something like:

> "**I** took **the** midterm today. **There were** fun problems, but **I** almost fell asleep. **I was** too exhausted from studying!"

or maybe

> "**I** took **a** midterm today. **It had** fun problems, but **I** almost fell asleep. **I am** too exhausted from studying!"

(a) [4 points] What is going on *linguistically*? You might think at first that this is a matter of post-processing on the word string. That is, perhaps Twitter users first generate a sentence from a standard English grammar, and then optionally delete some short common words.

But in fact, Twitter users do not just freely delete the words that are boldfaced above, such as **it** and **I**. It would sound strange if they wrote

> "*Was what. *Think need more sleep next time. *Sleep is good for."

instead of the standard-English version

> "**It** was what **it was**. **I** think **I** need more sleep next time. Sleep is good for **me**."

So let's modify the grammar directly instead of post-processing. What rules would you add to a standard English grammar in order to permit "good" abbreviated sentences (like the original examples, but not like the starred examples)? You may want to think about other examples besides the ones listed here.

(b) [6 points] Now, what can you try as an *engineer*? You would like to restore deleted words to Twitter posts (i.e., tweets). You have a probabilistic context-free grammar of English, G, which was estimated from the Penn Treebank (10^6 words of newspaper text). You also have a 5-gram language model of English, L, which was estimated from 10^{12} words of webpage text.

You are choosing whether to use G or L to help you with your task. Discuss how the following concerns might affect your choice:

- **bias**:

- **variance**:

- **runtime**:

(c) [3 points] You decide to use L for your task, as part of a noisy-channel approach. Your noisy channel corresponds to the naive post-processing approach suggested in part (a). For example, you expect that "It" will delete about 15% of the time, independent of syntax. This is not as smart as the syntactic model in (a), but you hope that L will be good enough to compensate for that.

Let w be the observed sentence, e.g., "Fun problems, but almost fell asleep." What maximization problem should you solve to find the most likely standard English sentence?

(Just write down what you need to maximize. Don't worry about efficient algorithms for maximizing it; we'll get to that later in the course.)