1. Ambiguity is still the problem

Features help express the grammar neatly, but they don't change the ambiguity problem

Chart parsing gives us flexibility, but that doesn't solve the ambiguity problem

Examples of ambiguity, a reminder:

**global PP attachment**

“One morning I shot an elephant in my pyjamas ...” (Groucho Marx)

**gerundive VP attachment**

We saw penguins flying over Antarctica

**coordination**

young boys and girls

vs

empty bottles and fag-ends

2. An aside about compositional semantics

We've never discussed the significance of attachment ambiguity

- An appeal to a **compositional** semantics has been implicit

Consider a simple grammar for Python arithmetic expressions:
Assuming some more work to **tokenise** the input, this will give us two analyses for the string "x/y+1":

If we associate the appropriate computation with each production
(skipping a lot of details here, some of which will be covered in a few weeks)
And assuming that x is 4 and y is 1

We see that the difference in attachment makes a very *real* difference

We call that kind of semantic computation a **compositional** one

"The meaning of the whole [constituent] is a function of the meaning of the parts [children]"

### 3. Compositional semantics, cont'd

How does this relate to natural language?

- Many approaches to language *understanding* proceed in a similar way
  - In what’s called a *rule-to-rule* way
  - Associating a (compositional) *semantic* rule with each *syntactic* rule

So, for example, for the kind of PP attachment ambiguity we keep encountering

- Where the PP is attached determines where its semantic contribution is made

In other words, with respect to "saw the child with the telescope"

**VP → V NP, NP → NP PP**

The semantics of the PP contributes to the semantics of the higher NP
- The child has the telescope

**VP → V NP PP**

The semantics of the PP contributes to the semantics of the VP
- The seeing was done with the telescope
4. Back to ambiguity

We need a way to choose the best analysis from among many

- Very many
- The average sentence in the Wall Street Journal dataset we used in lab on Tuesday is just under 26 words
- The exponential consequence of multiple local ambiguities given a broad-coverage CF-PSG
- Will be 1000s, if not 100s of 1000s, of parses

And we need a sound basis for ranking these

A gold standard provides at least partial solutions. . .

5. Treebanks: big investment, big reward

Mitch Marcus at the Univ. of Pennsylvania took this seriously

- As did the US DARPA (Defense Advanced Research Projects Agency)
  - In the context of their evaluation-led research funding approach

The Penn Treebank project was launched in 1989

- Over a number of phases has annotated many datasets, including
  - the Brown Corpus, WSJ data, air-travel request data, phone conversation data
- and has spawned many follow-ons for other languages

6. Treebank examples and evolution

The first release contained 'skeletal' parses, that is, simple syntactic trees

- With a few quirks based on a choice of underlying grammatical theory
- Now somewhat dated, this mostly involves the use of what are called traces to indicate 'missing' material
- For example in relative clauses and some complement clauses
  - "I liked the show that we watched * last night"
  - "Robin found it difficult to * lift the boxes"

Here's an example:
This annotation was produced by manually correcting the output of a simple chunking parser, then removing the POS-tags and some NP-internal structure.

7. Penn Treebank, mark two

The second release added much more information, including some predicate-argument indications:

8. Deriving a grammar from a treebank

Trivial, really

For every node in every tree with non-lexical children

- Add a rule to the grammar (or increment the count of an existing rule)
So, for the tree in the previous slide, we'd get e.g.

- NP-SBJ → NP ADVP-LOC
- NP → NP PP
- NP → NP PP SBAR

The result is guaranteed to give at least one parse to every sentence in the Treebank

### 9. Simple Probabilistic CF-PSGs

So given a treebank, we can easily compute a simple kind of probabilistic grammar

- Either directly with respect to treebank-derived rules
- Or by mapping from treebank statistics to some other ruleset

For a treebank-derived ruleset, the maximum likelihood probability estimate is simple:

\[
P(NT \rightarrow C_1 C_2 \ldots C_n | NT) = \frac{\text{count}(NT \rightarrow C_1 C_2 \ldots C_n)}{\text{count}(NT)}
\]

So, for example, given that in the NLTK treebank subset, there are 52041 subtrees labelled **S**

And in 29201 of these there are exactly two children of **S**, labelled **NP-SBJ** followed by **VP**, we get

\[
P(S \rightarrow \text{NP-SBJ VP} | S) = \frac{29201}{52041} = 0.56
\]

The probability of a whole parse tree is then just the product of the probabilities of every nonterminal node in that tree

- Or we can use the sum of the costs for simpler calculations
  - Recall that costs are negative log probabilities
  - We sum them, instead of multiplying
  - And prefer lower costs (== higher probabilities)

It's not hard to modify a chart parser to use probabilities to actually guide parsing

- So that you get the most likely parse first
- By maintaining the agenda in sorted order
- And always taking the most probable agenda entry to put in the chart next

### 10. Evaluation against a treebank

No-one's grammatical theory/detailed grammar is going to give exactly the same results as a treebank

- Unless it's derived automatically from the treebank

So how do you evaluate a parser against a treebank?

It turns out just looking at major constituent boundaries is surprisingly good
• Basically, because most parsers are pretty bad

The same (D)ARPA push towards evaluation-driven funding drove the development of the PARSEVAL metric

PARSEVAL just compares bracketings, without regard to labels

• In its simplest form, it just counts parenthesis-crossing
  ◦ (A (B C)) vs. ((A B) C)
  ◦ Fewer is better
• And constituent recall
  ◦ (A B C) vs. (A (B C))
  ◦ More is better

Each of these is effectively penalising an attachment mistake

11. PARSEVAL examples

```
(saw (a child with a telescope))
```

will be penalised wrt

```
(saw (a child) (with a telescope))
```

• by the second rule above, because of the mismatch in number of sub-constituents

```
((young boys) and girls)
```

will be penalised wrt

```
(young (boys and girls))
```

• because of the crossing parentheses

12. Treebank grammar problems

The bad news: just using the treebank subset that ships with NLTK, there are 1798 different expansions for S

• Some are very common, such as S → NP-SBJ VP
• Most occur only once (Zipf again)

As always, there's a tradeoff between specificity and statistical significance

• More detailed symbols means more symbols means fewer examples of each

For getting good PARSEVAL scores

• learned-from-treebank grammars tended to collapse many of the release-2 categories back to something more like their release-1 counterparts