Ambiguity
Programs must be written for people to read

Preface of *Structure and Interpretation of Computer Programs* by Harold Abelson and Gerald Sussman with Julie Sussman
Syntactic Ambiguity in English

Programs must be written for people to read

Preface of *Structure and Interpretation of Computer Programs* by Harold Abelson and Gerald Sussman with Julie Sussman
Syntactic Ambiguity in English

Programs must be written for people to read

Preface of *Structure and Interpretation of Computer Programs* by Harold Abelson and Gerald Sussman with Julie Sussman
Syntactic Ambiguity in English

program (noun)
a series of coded software instructions

program (verb)
provide a computer with coded instructions

Programs must be written for people to read

must (verb)
be obliged to

must (noun)
dampness or mold

Definitions from the New Oxford American Dictionary
Syntax Trees
Representing Syntactic Structure

A **Tree** represents a phrase:
- **tag** — What kind of phrase (e.g., S, NP, VP)
- **branches** — Sequence of Tree or Leaf components

A **Leaf** represents a single word:
- **tag** — What kind of word (e.g., N, V)
- **word** — The word

```
cows = Leaf('N', 'cows')
intimidate = Leaf('V', 'intimidate')
S, NP, VP = 'S', 'NP', 'VP'
Tree(S, [Tree(NP, [cows]),
    Tree(VP, [intimidate,
        Tree(NP, [cows])])])
```
Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words.

A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

<table>
<thead>
<tr>
<th>Rule</th>
<th>Expansions</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → NP VP</td>
<td>cows intimidate</td>
</tr>
<tr>
<td>NP → N</td>
<td></td>
</tr>
<tr>
<td>N → cows</td>
<td></td>
</tr>
<tr>
<td>VP → V NP</td>
<td></td>
</tr>
<tr>
<td>V → intimidate</td>
<td></td>
</tr>
</tbody>
</table>

(Demo)
Parsing
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input

Constraint: A Leaf must match the input word
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input.

(Demo)
Learning

(Demo)
Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

```
S  →  NP  VP
NP →  NN  NNS
VP →  VB  NP
NP →  NNS
```

```
NN  →  teacher
NNS →  strikes
VB  →  idle
NNS →  kids
```

Rule frequency per 100,000 tags

```
S  →  NP  VP  25372
NP →  NN  NNS  1335
VP →  VB  NP    6679
NP →  NNS      4282
```
Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common.

Rule frequency per 100,000 tags

- $S \rightarrow NP \ VP$ 25372  
  $NN \rightarrow$ teacher 5
- $NP \rightarrow NN$ 1335 4358  
  $VBZ \rightarrow$ strikes 25 19
- $VP \rightarrow VBZ NP$ 6679 3160  
  $JJ \rightarrow$ idle 26 18
- $NP \rightarrow JJ NNS$ 4282 2526  
  $NNS \rightarrow$ kids 32

(Demo)
Syntactic Reordering

**English** → **Yoda-English**

- **S**
  - **VP**
    - **VB** help
    - **PRP** you
  - **NP**
    - **PRP** I
  - **VP**
    - **MD** can

**Help you, I can!**

**Yes! Mm!**

**When 900 years old you reach,**

**look as good, you will not. Hm.**

(Demo)