We’ll start with some puzzles

- The women discussed the dogs on the beach.
We’ll start with some puzzles

- *The women discussed the dogs on the beach.*
- What does *on the beach* modify?
We’ll start with some puzzles

- *The women discussed the dogs on the beach.*
- What does *on the beach* modify?

  *Ford et al., 1982* → *dogs* (90%); *discussed* (10%)*
We’ll start with some puzzles

• The women discussed the dogs on the beach.
  • What does on the beach modify?

  Ford et al., 1982→ dogs (90%); discussed (10%)

• The women kept the dogs on the beach.
We’ll start with some puzzles

• The women discussed the dogs on the beach.
  • What does on the beach modify?

  Ford et al., 1982→ dogs (90%); discussed (10%)

• The women kept the dogs on the beach.
  • What does on the beach modify?
We’ll start with some puzzles

• The women discussed the dogs on the beach.
  • What does on the beach modify?
    
    Ford et al., 1982→ dogs (90%); discussed (10%)

• The women kept the dogs on the beach.
  • What does on the beach modify?
    
    Ford et al., 1982→ kept (95%); dogs (5%)
We’ll start with some puzzles

• **The women discussed the dogs on the beach.**
  • What does *on the beach* modify?

  *Ford et al., 1982*→ *dogs* (90%); *discussed* (10%)

• **The women kept the dogs on the beach.**
  • What does *on the beach* modify?

  *Ford et al., 1982*→ *kept* (95%); *dogs* (5%)

• **The complex houses married children and their families.**
We’ll start with some puzzles

- The women discussed the dogs on the beach.
  - What does on the beach modify?

    *Ford et al., 1982* → *dogs* (90%); *discussed* (10%)

- The women kept the dogs on the beach.
  - What does on the beach modify?

    *Ford et al., 1982* → *kept* (95%); *dogs* (5%)

- The complex houses married children and their families.

- The warehouse fires a dozen employees each year.
Comprehension: Theoretical Desiderata

• Realistic models of human sentence comprehension must account for:
Comprehension: Theoretical Desiderata

- Realistic models of human sentence comprehension must account for:
  - Robustness to arbitrary input
Comprehension: Theoretical Desiderata

• Realistic models of human sentence comprehension must account for:
  • Robustness to arbitrary input
  • Accurate disambiguation
Comprehension: Theoretical Desiderata

- Realistic models of human sentence comprehension must account for:
  - Robustness to arbitrary input
  - Accurate disambiguation
  - Inference on basis of incomplete input

Comprehension: Theoretical Desiderata

- Realistic models of human sentence comprehension must account for:
  - Robustness to arbitrary input
  - Accurate disambiguation
  - Inference on basis of incomplete input
Realistic models of human sentence comprehension must account for:

- Robustness to arbitrary input
- Accurate disambiguation
- Inference on basis of incomplete input


\[ \text{the boy will eat...} \]
Comprehension: Theoretical Desiderata

- Realistic models of human sentence comprehension must account for:
  - Robustness to arbitrary input
  - Accurate disambiguation
  - Inference on basis of incomplete input

*the boy will eat...*
Comprehension: Theoretical Desiderata

- Realistic models of human sentence comprehension must account for:
  - Robustness to arbitrary input
  - Accurate disambiguation
  - Inference on basis of incomplete input
Comprehension: Theoretical Desiderata

- Realistic models of human sentence comprehension must account for:
  - Robustness to arbitrary input
  - Accurate disambiguation

- Processing difficulty is differential and localized
Comprehension: Theoretical Desiderata

- Realistic models of human sentence comprehension must account for:
  - Robustness to arbitrary input
  - Accurate disambiguation
  - Inference on basis of incomplete input
    (Tanenhaus et al 1995, Altmann and Kamide 1999,
    Kaiser and Trueswell 2004)

- Processing difficulty is differential and localized
Comprehension: Theoretical Desiderata

Realistic models of human sentence comprehension must account for:

- Robustness to arbitrary input
- Accurate disambiguation

how to get from here...

- Processing difficulty is differential and localized
Comprehension: Theoretical Desiderata

- Realistic models of human sentence comprehension must account for:
  - Robustness to arbitrary input
  - Accurate disambiguation
  - Inference on basis of incomplete input

how to get from here...
...to here?

- Processing difficulty is differential and localized
Crash course in grammars and parsing
A grammar is a structured set of production rules.
Crash course in grammars and parsing

- A *grammar* is a structured set of production rules
- Most commonly used for syntactic description, but also useful for (semantics, phonology, ...)


Crash course in grammars and parsing

- A grammar is a structured set of production rules
- Most commonly used for syntactic description, but also useful for (semantics, phonology, …)
- E.g., context-free grammars:

  S \rightarrow NP \ VP
  NP \rightarrow Det \ N
  VP \rightarrow V \ NP

  Det \rightarrow the
  N \rightarrow dog
  N \rightarrow cat
  V \rightarrow chased
• A grammar is a structured set of production rules
• Most commonly used for syntactic description, but also useful for (semantics, phonology, …)
• E.g., context-free grammars:

\[
\begin{align*}
S & \rightarrow \text{NP } \text{VP} \\
\text{NP} & \rightarrow \text{Det } \text{N} \\
\text{VP} & \rightarrow \text{V } \text{NP}
\end{align*}
\]

\[
\begin{align*}
\text{Det} & \rightarrow \text{the} \\
\text{N} & \rightarrow \text{dog} \\
\text{N} & \rightarrow \text{cat} \\
\text{V} & \rightarrow \text{chased}
\end{align*}
\]

• A grammar is said to license a derivation if all the derivation’s rules are present in the grammar
A **grammar** is a structured set of production rules

Most commonly used for syntactic description, but also useful for (semantics, phonology, …)

E.g., context-free grammars:

\[
\begin{align*}
S & \rightarrow NP \ VP \\
NP & \rightarrow Det \ N \\
VP & \rightarrow V \ NP \\
Det & \rightarrow \text{the} \\
N & \rightarrow \text{dog} \\
N & \rightarrow \text{cat} \\
V & \rightarrow \text{chased}
\end{align*}
\]

A grammar is said to **license** a derivation if all the derivation’s rules are present in the grammar
Crash course in grammars and parsing

• A grammar is a structured set of production rules
• Most commonly used for syntactic description, but also useful for (semantics, phonology, …)
• E.g., context-free grammars:

```
S → NP  VP
NP → Det  N
VP → V  NP
   Det → the
   N → dog
   N → cat
   V → chased
```

• A grammar is said to license a derivation if all the derivation’s rules are present in the grammar
A grammar is a structured set of production rules. Most commonly used for syntactic description, but also useful for (semantics, phonology, …)

E.g., context-free grammars:

\[
\begin{align*}
S &\rightarrow NP \quad VP \\
NP &\rightarrow Det \quad N \\
VP &\rightarrow V \quad NP \\
Det &\rightarrow the \\
N &\rightarrow dog \\
N &\rightarrow cat \\
V &\rightarrow chased
\end{align*}
\]

A grammar is said to license a derivation if all the derivation’s rules are present in the grammar.
A grammar is a structured set of production rules

Most commonly used for syntactic description, but also useful for (semantics, phonology, …)

E.g., context-free grammars:

\[
\begin{align*}
S & \rightarrow \text{NP} \hspace{5mm} \text{VP} \\
\text{NP} & \rightarrow \text{Det} \hspace{5mm} \text{N} \\
\text{VP} & \rightarrow \text{V} \hspace{5mm} \text{NP} \\
\text{Det} & \rightarrow \text{the} \\
\text{N} & \rightarrow \text{dog} \\
\text{N} & \rightarrow \text{cat} \\
\text{V} & \rightarrow \text{chased}
\end{align*}
\]

A grammar is said to license a derivation if all the derivation’s rules are present in the grammar
Context-free Grammars

A context-free grammar (CFG) consists of a tuple \((N, V, S, R)\) such that:

- \(N\) is a finite set of non-terminal symbols;
- \(V\) is a finite set of terminal symbols;
- \(S\) is the start symbol;
- \(R\) is a finite set of rules of the form \(X \rightarrow \alpha\) where \(X \in N\) and \(\alpha\) is a sequence of symbols drawn from \(N \cup V\).

A CFG \textit{derivation} is the recursive expansion of non-terminal symbols in a string by rules in \(R\), starting with \(S\), and a \textit{derivation tree} \(T\) is the history of those rule applications.
Context-free Grammars: an example

Let our grammar (the rule-set $R$) be

<table>
<thead>
<tr>
<th>Rule</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>$NP , VP$</td>
</tr>
<tr>
<td>$NP$</td>
<td>$Det , N$</td>
</tr>
<tr>
<td>$NP$</td>
<td>$NP , PP$</td>
</tr>
<tr>
<td>$PP$</td>
<td>$P , NP$</td>
</tr>
<tr>
<td>$VP$</td>
<td>$V$</td>
</tr>
<tr>
<td>$Det$</td>
<td>the</td>
</tr>
<tr>
<td>$N$</td>
<td>dog</td>
</tr>
<tr>
<td>$N$</td>
<td>cat</td>
</tr>
<tr>
<td>$P$</td>
<td>near</td>
</tr>
<tr>
<td>$V$</td>
<td>growled</td>
</tr>
</tbody>
</table>

The nonterminal set $N$ is \{ $S, \, NP, \, VP, \, Det, \, N, \, P, \, V$ \}, the terminal set $V$ is \{ $the, \, dog, \, cat, \, near, \, growled$ \}, and our start symbol $S$ is $S$. 
Here is a *derivation* and the resulting *derivation tree*:

```
S → NP VP
NP → Det N
NP → NP PP
PP → P NP
VP → V
```

```
S
```

Det → the
N → dog
N → cat
P → near
V → growled
Context-free Grammars: an example II

S → NP VP
NP → Det N
NP → NP PP
PP → P NP
VP → V

Det → the
N → dog
N → cat
P → near
V → growled

Here is a derivation and the resulting derivation tree:
Context-free Grammars: an example II

\[
\begin{align*}
S & \rightarrow NP \ VP \\
NP & \rightarrow \text{Det} \ N \\
NP & \rightarrow NP \ PP \\
PP & \rightarrow P \ NP \\
VP & \rightarrow V \\
\text{Det} & \rightarrow \text{the} \\
N & \rightarrow \text{dog} \\
N & \rightarrow \text{cat} \\
P & \rightarrow \text{near} \\
V & \rightarrow \text{growled}
\end{align*}
\]

Here is a derivation and the resulting derivation tree:
Context-free Grammars: an example II

\[
S \rightarrow NP \ VP \\
NP \rightarrow \text{Det } N \\
NP \rightarrow NP \ PP \\
PP \rightarrow P \ NP \\
VP \rightarrow V
\]

Det\rightarrow \text{the} \\
N \rightarrow \text{dog} \\
N \rightarrow \text{cat} \\
P \rightarrow \text{near} \\
V \rightarrow \text{growled}

Here is a \textit{derivation} and the resulting \textit{derivation tree}:
Context-free Grammars: an example II

S → NP VP
NP → Det N
NP → NP PP
PP → P NP
VP → V

Det → the
N → dog
N → cat
P → near
V → growled

Here is a *derivation* and the resulting *derivation tree*:

```
S
  /  
 NP   VP
  /    /
 NP   PP
    /   /
 Det N
     /
    the
```
Context-free Grammars: an example II

S → NP VP
NP → Det N
NP → NP PP
PP → P NP
VP → V

Det → the
N → dog
N → cat
P → near
V → growled

Here is a *derivation* and the resulting *derivation tree*:

```
           S
          /|
         / E
        /  |
       NP   VP
      /|
     /  |
    NP PP
   / |
  Det N
 /   |
the dog
```
Context-free Grammars: an example II

\[
\begin{align*}
S & \rightarrow \text{NP VP} \\
\text{NP} & \rightarrow \text{Det N} \\
\text{NP} & \rightarrow \text{NP PP} \\
\text{PP} & \rightarrow \text{P NP} \\
\text{VP} & \rightarrow \text{V} \\
\text{Det} & \rightarrow \text{the} \\
\text{N} & \rightarrow \text{dog} \\
\text{N} & \rightarrow \text{cat} \\
\text{P} & \rightarrow \text{near} \\
\text{V} & \rightarrow \text{growled}
\end{align*}
\]

Here is a *derivation* and the resulting *derivation tree*:

```
    S
   / \  \\
 NP   VP
 /     |
 NP     PP
|     |
 Det   P   NP
     |   |
   the  dog
```
Context-free Grammars: an example II

\[
S \rightarrow NP \ VP \\
NP \rightarrow \text{Det N} \\
NP \rightarrow \text{NP PP} \\
PP \rightarrow P \ NP \\
VP \rightarrow V \\
\text{Det} \rightarrow \text{the} \\
N \rightarrow \text{dog} \\
N \rightarrow \text{cat} \\
P \rightarrow \text{near} \\
V \rightarrow \text{growled}
\]

Here is a \textit{derivation} and the resulting \textit{derivation tree}:

![Derivation Tree]

- \text{S}
- \text{NP}
- \text{VP}
- \text{NP}
- \text{PP}
- \text{Det} \rightarrow \text{the}
- \text{N} \rightarrow \text{dog}
- \text{N} \rightarrow \text{cat}
- \text{P} \rightarrow \text{near}
- \text{V} \rightarrow \text{growled}
Context-free Grammars: an example II

S → NP VP
NP → Det N
NP → NP PP
PP → P NP
VP → V

Det → the
N → dog
N → cat
P → near
V → growled

Here is a derivation and the resulting derivation tree:

```
S
  /\  
NP | VP
  /\  
NP | PP
  /\  
Det | N | P | NP
     /\  

the | dog | near | Det | N
```
Context-free Grammars: an example II

S → NP VP
NP → Det N
NP → NP PP
PP → P NP
VP → V

Det → the
N → dog
N → cat
P → near
V → growled

Here is a derivation and the resulting derivation tree:

```
S
  /\   \\
 NP /  \ VP
  /\   \\
 NP /  \ PP
    /   /
  Det N P NP
     /  /
   the dog near Det N
      /  /
     the
```
Context-free Grammars: an example II

\[
\begin{align*}
S & \rightarrow NP \ VP \\
NP & \rightarrow Det \ N \\
NP & \rightarrow NP \ PP \\
PP & \rightarrow P \ NP \\
VP & \rightarrow V
\end{align*}
\]

\[
\begin{align*}
Det & \rightarrow the \\
N & \rightarrow dog \\
N & \rightarrow cat \\
P & \rightarrow near \\
V & \rightarrow growled
\end{align*}
\]

Here is a *derivation* and the resulting *derivation tree*:

```
  S
 / \ \\
NP  VP
  /  \\
NP PP
  /   \\
Det N P NP
     /     \\
   the dog near Det N
       /       |
      the       the
```

*the dog near the cat*
Context-free Grammars: an example II

S → NP VP
NP → Det N
NP → NP PP
PP → P NP
VP → V

Det → the
N → dog
N → cat
P → near
V → growled

Here is a *derivation* and the resulting *derivation tree*:
Context-free Grammars: an example II

\[ S \rightarrow NP \ VP \]
\[ NP \rightarrow Det \ N \]
\[ NP \rightarrow NP \ PP \]
\[ PP \rightarrow P \ NP \]
\[ VP \rightarrow V \]

\[ Det \rightarrow \text{the} \]
\[ N \rightarrow \text{dog} \]
\[ N \rightarrow \text{cat} \]
\[ P \rightarrow \text{near} \]
\[ V \rightarrow \text{growled} \]

Here is a *derivation* and the resulting *derivation tree*:

```
S
  /\  \\
 NP /  VP\
  /\   /\  \\
 NP /   PP\\
  /\   /\\
 Det / the\\
   /\
  dog \\
  /\
 the \\
```

```
  /\  \\
 P /  NP\\
  /\   /\\
 Det / the\\
   /\
 cat \\
```

```
  /\  \\
 V /  growled\\
  /\   /\\
 P / the\\
  /\
 near \\
```

Probabilistic Context-Free Grammars

A probabilistic context-free grammar (PCFG) consists of a tuple \((N, V, S, R, P)\) such that:

- \(N\) is a finite set of non-terminal symbols;
- \(V\) is a finite set of terminal symbols;
- \(S\) is the start symbol;
- \(R\) is a finite set of rules of the form \(X \to \alpha\) where \(X \in N\) and \(\alpha\) is a sequence of symbols drawn from \(N \cup V\);
- \(P\) is a mapping from \(R\) into probabilities, such that for each \(X \in N\),

\[
\sum_{[X \to \alpha] \in R} P(X \to \alpha) = 1
\]

PCFG derivations and derivation trees are just like for CFGs. The probability \(P(T)\) of a derivation tree is simply the product of the probabilities of each rule application.
### Example PCFG

<table>
<thead>
<tr>
<th></th>
<th>S</th>
<th>NP → VP</th>
<th>Det → the</th>
<th>N → dog</th>
<th>N → cat</th>
<th>P → near</th>
<th>V → growled</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S → NP</td>
<td>VP</td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.8</td>
<td>NP → Det N</td>
<td></td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.2</td>
<td>NP → NP PP</td>
<td></td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>PP → P NP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>VP → V</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
S
  NP
    Det N
      the dog
    PP
      P near
      NP
        Det N
          the cat
  VP
    V growled
```

\[
P(T) = 1 \times 0.2 \times 0.8 \times 1 \times 0.5 \times 0.8 \times 1 \times \times 0.8 \times 1 \times 0.5 \times 1 \times 1 \times 1 \\
= 0.032
\]
Top-down parsing

\[
S \rightarrow \text{NP} \text{ VP} \\
\text{NP} \rightarrow \text{Det} \text{ N} \\
\text{Det} \rightarrow \text{The} \\
\ldots
\]

The coach smiled at the player tossed the frisbee.
Top-down parsing

\[
S \rightarrow NP \ VP \\
NP \rightarrow \ Det \ N \\
Det \rightarrow \ The \\
\ldots
\]

The coach smiled at the player tossed the frisbee.
Top-down parsing

\[
\begin{align*}
S & \rightarrow NP \ VP \\
NP & \rightarrow Det \ N
\end{align*}
\]

Det $\rightarrow$ The

... 

The coach smiled at the player tossed the frisbee.
Top-down parsing

The coach smiled at the player tossed the frisbee.
Top-down parsing

- Fundamental operation:

  \[
  S \rightarrow NP \; VP \\
  NP \rightarrow Det \; N
  \]

  Det \rightarrow The

- Permits structure building inconsistent with perceived input, or corresponding to as-yet-unseen input

The coach smiled at the player tossed the frisbee.
Bottom-up parsing

VP → V NP
PP → P NP
S → NP VP
...
Bottom-up parsing

\[
\begin{align*}
VP & \rightarrow V \ NP \\
PP & \rightarrow P \ NP \\
S & \rightarrow NP \ VP
\end{align*}
\]

...
Bottom-up parsing

\[
\begin{align*}
VP & \rightarrow V \ NP \\
PP & \rightarrow P \ NP \\
S & \rightarrow NP \ VP \\
& \ldots
\end{align*}
\]

The coach smiled at the player tossed the frisbee.
Bottom-up parsing

• Fundamental operation: check whether a sequence of categories matches a rule’s *right-hand* side

  \[
  \begin{align*}
  VP &\rightarrow V \quad NP \\
  PP &\rightarrow P \quad NP \\
  S &\rightarrow NP \quad VP \\
  \quad \vdots
  \end{align*}
  \]

• Permits structure building inconsistent with global context
Bottom-up parsing

- Fundamental operation: check whether a sequence of categories matches a rule's right-hand side

\[
\begin{align*}
    \text{VP} & \rightarrow \text{V} \quad \text{NP} \\
    \text{PP} & \rightarrow \text{P} \quad \text{NP}
\end{align*}
\]

\[
\begin{align*}
    \text{S} & \rightarrow \text{NP} \quad \text{VP} \\
    \text{...}
\end{align*}
\]

- Permits structure building inconsistent with global context

The coach smiled at the player tossed the frisbee.
Ambiguity

- There is usually more than one structural analysis for a (partial) sentence

  *The girl saw the boy with...*

- Corresponds to *choices* (non-determinism) in parsing
Ambiguity

• There is usually more than one structural analysis for a (partial) sentence

  The girl saw the boy with...

• Corresponds to choices (non-determinism) in parsing
• VP can expand to V NP PP…
Ambiguity

- There is usually more than one structural analysis for a (partial) sentence

  *The girl saw the boy with...*

- Corresponds to *choices* (non-determinism) in parsing
- VP can expand to V NP PP...
- ...or VP can expand to V NP and then NP can expand to NP PP
Ambiguity

- There is usually more than one structural analysis for a (partial) sentence

  \textit{The girl saw the boy with...}

- Corresponds to \textit{choices} (non-determinism) in parsing
- VP can expand to \textit{V NP PP}...
- ...or VP can expand to \textit{V NP} and then NP can expand to NP PP
- Ambiguity can be \textit{local} (eventually resolved)...
  - ...\textit{with a puppy on his lap}. 
Ambiguity

- There is usually more than one structural analysis for a (partial) sentence

  *The girl saw the boy with...*

- Corresponds to *choices* (non-determinism) in parsing
- VP can expand to V NP PP...
- ...or VP can expand to V NP and then NP can expand to NP PP
- Ambiguity can be *local* (eventually resolved)...
  - ...*with a puppy on his lap.*
- ...or it can be *global* (unresolved):
  - ...*with binoculars.*
Serial vs. Parallel processing
Serial vs. Parallel processing

A *serial* processing model is one where, when faced with a choice, chooses one alternative and discards the rest.
Serial vs. Parallel processing

- A *serial* processing model is one where, when faced with a choice, chooses one alternative and discards the rest.
- A *parallel* model is one where at least two alternatives are chosen and maintained.
Serial vs. Parallel processing

• A *serial* processing model is one where, when faced with a choice, chooses one alternative and discards the rest

• A *parallel* model is one where at least two alternatives are chosen and maintained
  • A *full parallel* model is one where *all* alternatives are maintained
Serial vs. Parallel processing

- A *serial* processing model is one where, when faced with a choice, chooses one alternative and discards the rest.

- A *parallel* model is one where at least two alternatives are chosen and maintained.
  - A *full parallel* model is one where *all* alternatives are maintained.
  - A *limited parallel* model is one where *some but not necessarily all* alternatives are maintained.
A *serial* processing model is one where, when faced with a choice, chooses one alternative and discards the rest.

A *parallel* model is one where at least two alternatives are chosen and maintained.

- A *full parallel* model is one where *all* alternatives are maintained.
- A *limited parallel* model is one where *some but not necessarily all* alternatives are maintained.

A joke about the man with an umbrella that I heard…
Serial vs. Parallel processing

• A *serial* processing model is one where, when faced with a choice, chooses one alternative and discards the rest

• A *parallel* model is one where at least two alternatives are chosen and maintained
  • A *full parallel* model is one where *all* alternatives are maintained
  • A *limited parallel* model is one where *some but not necessarily all* alternatives are maintained

A joke about the man with an umbrella that I heard…

*ambiguity goes as the Catalan numbers (Church and Patel 1982):* 1 2 5 14 42 132…
Dynamic programming

- There is an exponential number of parse trees for a given sentence (Church & Patil 1982)
- So sentence comprehension can’t entail an exhaustive enumeration of possible structural representations
- But parsing can be made tractable by *dynamic programming*
Dynamic programming

- There is an exponential number of parse trees for a given sentence (Church & Patil 1982)
- So sentence comprehension can’t entail an exhaustive enumeration of possible structural representations
- But parsing can be made tractable by *dynamic programming*
Dynamic programming (2)

- Dynamic programming = storage of partial results
- There are two ways to make an NP out of...

...but the resulting NP can be stored just once in the parsing process
- Result: parsing time polynomial (cubic for CFGs) in sentence length
- Still problematic for modeling human sentence processing
The workhorses of psycholinguistic experimentation involve *behavioral* measures:

- What choices do people make in various types of language-producing and language-comprehending situations?
- and how long do they take to make these choices?

**Offline** measures:
- rating sentences, completing sentences, …

**Online** measures:
- tracking people’s eye movements, having people read words aloud, reading under (implicit) time pressure…
Psycholinguistic methodology (2)
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press

while -----------------------------------------------
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press

----- the -----
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press
Psycholinguistic methodology (2)

• Example online-processing methodology: word-by-word self-paced reading
• Reveal each consecutive word with a button press

---------------- crackled, ---------------------------------------------
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press

-------------------------------- the ----------------------------------
• Example online-processing methodology: word-by-word self-paced reading
• Reveal each consecutive word with a button press
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press

------------------------------------------- soared --------------------------

soared --------------------------
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press

--- soared ----------------------

- Readers aren’t allowed to backtrack
Psycholinguistic methodology (2)

- Example online-processing methodology: word-by-word self-paced reading
- Reveal each consecutive word with a button press
- Readers aren’t allowed to backtrack
- We measure time between button presses and use it as a proxy for incremental processing difficulty
Psycholinguistic methodology (3)

- Caveat: *neurolinguistic* experimentation more and more widely used to study language comprehension
  - methods vary in temporal and spatial resolution
  - people are more passive in these experiments: sit back and listen to/read a sentence, word by word
  - strictly speaking *not* behavioral measures
  - the question of “what is difficult” becomes a little less straightforward
Estimating grammar probabilities

• To test our models, we need to approximate the probabilistic linguistic knowledge of the native speaker

• *By principle of rational analysis*: native speaker’s probabilities should match those of the environment

• We can use *syntactically annotated datasets* (Treebanks) to estimate frequencies in the environment

• e.g., via *relative frequency estimation*

\[
P(\text{NP} \rightarrow \text{Det N}) = \frac{2}{3} \\
P(\text{NP} \rightarrow \text{N}) = \frac{1}{3} \\
P(\text{Det} \rightarrow \text{the}) = 1 \\
P(\text{N} \rightarrow \text{dog}) = \frac{1}{3} \\
P(\text{N} \rightarrow \text{cats}) = \frac{2}{3}
\]
Pruning approaches

- Jurafsky 1996: a probabilistic approach to lexical access and syntactic disambiguation
- Main argument: sentence comprehension is probabilistic, construction-based, and parallel
- Probabilistic parsing model explains
  - human disambiguation preferences
  - garden-path sentences
- The probabilistic parsing model has two components:
  - *constituent* probabilities – a probabilistic CFG model
  - *valence* probabilities
• Every word is immediately completely integrated into the parse of the sentence (i.e., *full incrementality*)
• Alternative parses are ranked in a probabilistic model
• Parsing is *limited-parallel*: when an alternative parse has unacceptably low probability, it is *pruned*
• “Un acceptably low” is determined by *beam search* (described a few slides later)
Jurafsky 1996: valency model
Whereas the constituency model makes use of only phrasal, not lexical information, the valency model tracks lexical subcategorization, e.g.:
Jurafsky 1996: valency model

- Whereas the constituency model makes use of only phrasal, not lexical information, the valency model tracks lexical subcategorization, e.g.:
  \[ P( <NP \text{ PP}> \mid discuss ) = 0.24 \]
Jurafsky 1996: valency model

Whereas the constituency model makes use of only phrasal, not lexical information, the valency model tracks lexical subcategorization, e.g.:

\[
P( <NP \text{ PP}> \mid \text{discuss} ) = 0.24
\]

\[
P( <NP> \mid \text{discuss} ) = 0.76
\]
Whereas the constituency model makes use of only phrasal, not lexical information, the valency model tracks lexical subcategorization, e.g.:

\[ P( \text{NP PP} | \text{discuss} ) = 0.24 \]
\[ P( \text{NP} | \text{discuss} ) = 0.76 \]

(in today’s NLP, these are called monolexical probabilities)
Jurafsky 1996: valency model

- Whereas the constituency model makes use of only phrasal, not lexical information, the valency model tracks lexical subcategorization, e.g.:
  \[ P( <NP \text{ PP}> | \text{ discuss} ) = 0.24 \]
  \[ P( <NP> | \text{ discuss} ) = 0.76 \]
  (in today’s NLP, these are called monolexical probabilities)
- In some cases, Jurafsky bins across categories:*

*valence probs are RFES from Connine et al. (1984) and Penn Treebank
Jurafsky 1996: valency model

• Whereas the constituency model makes use of only phrasal, not lexical information, the valency model tracks lexical subcategorization, e.g.:
  \[ P(\text{<NP PP}> | \text{discuss}) = 0.24 \]
  \[ P(\text{<NP}> | \text{discuss}) = 0.76 \]
  (in today’s NLP, these are called *monolexical* probabilities)

• In some cases, Jurafsky bins across categories:*
  \[ P(\text{<NP XP[+pred]> | keep}) = 0.81 \]

*valence probs are RFEs from Connine et al. (1984) and Penn Treebank
Jurafsky 1996: valency model

- Whereas the constituency model makes use of only phrasal, not lexical information, the valency model tracks lexical subcategorization, e.g.:
  \[
  P( <NP \text{ PP} > | \text{discuss} ) = 0.24 \\
  P( <NP > | \text{discuss} ) = 0.76
  \]
  (in today’s NLP, these are called monolexical probabilities)

- In some cases, Jurafsky bins across categories:*
  \[
  P( <NP \text{ XP} [+\text{pred}] > | \text{keep} ) = 0.81 \\
  P( <NP > | \text{keep} ) = 0.19
  \]

*valence probs are RFEs from Connine et al. (1984) and Penn Treebank
Jurafsky 1996: valency model

- Whereas the constituency model makes use of only phrasal, not lexical information, the valency model tracks lexical subcategorization, e.g.:
  \[ P( <NP \text{ PP}> \mid \text{discuss} ) = 0.24 \]
  \[ P( <NP> \mid \text{discuss} ) = 0.76 \]
  (in today’s NLP, these are called monolexical probabilities)

- In some cases, Jurafsky bins across categories:
  \[ P( <NP \text{ XP}[+\text{pred}]> \mid \text{keep}) = 0.81 \]
  \[ P( <NP> \mid \text{keep} ) = 0.19 \]
  where \( \text{XP}[+\text{pred}] \) can vary across \( \text{AdjP, VP, PP, Particle} \)…

*valence probs are RFEs from Connine et al. (1984) and Penn Treebank
The syntactic component of Jurafsky’s model is just probabilistic context-free grammars (PCFGs).

Total probability: \(0.7 \times 0.35 \times 0.15 \times 0.3 \times 0.03 \times 0.02 \times 0.4 \times 0.07 = 1.85 \times 10^{-7}\)
Modeling offline preferences

- Ford et al. 1982 found effect of lexical selection in PP attachment preferences (offline, forced-choice):
  - The women *discussed* the dogs on the beach
Modeling offline preferences

- Ford et al. 1982 found effect of lexical selection in PP attachment preferences (offline, forced-choice):
  - The women *discussed* the dogs on the beach
    - NP-attachment (the dogs that were on the beach) -- 90%
    - VP-attachment (discussed while on the beach) – 10%
Modeling offline preferences

- Ford et al. 1982 found effect of lexical selection in PP attachment preferences (offline, forced-choice):
  - The women *discussed* the dogs on the beach
    - NP-attachment (the dogs that were on the beach) -- 90%
    - VP-attachment (discussed while on the beach) – 10%
  - The women *kept* the dogs on the beach
Modeling offline preferences

- Ford et al. 1982 found effect of lexical selection in PP attachment preferences (offline, forced-choice):
  - The women *discussed* the dogs on the beach
    - NP-attachment (the dogs that were on the beach) -- 90%
    - VP-attachment (discussed while on the beach) – 10%
  - The women *kept* the dogs on the beach
    - NP-attachment – 5%
    - VP-attachment – 95%
Modeling offline preferences

• Ford et al. 1982 found effect of lexical selection in PP attachment preferences (offline, forced-choice):
  • The women *discussed* the dogs on the beach
    • NP-attachment (the dogs that were on the beach) -- 90%
    • VP-attachment (discussed while on the beach) – 10%
  • The women *kept* the dogs on the beach
    • NP-attachment – 5%
    • VP-attachment – 95%
• These are *uncertainties about what has already been said*
Modeling offline preferences

• Ford et al. 1982 found effect of lexical selection in PP attachment preferences (offline, forced-choice):
  • The women *discussed* the dogs on the beach
    • NP-attachment (the dogs that were on the beach) -- 90%
    • VP-attachment (discussed while on the beach) – 10%
  • The women *kept* the dogs on the beach
    • NP-attachment – 5%
    • VP-attachment – 95%

• These are *uncertainties about what has already been said*

• Overall results broadly confirmed in online attachment study by Taraban and McClelland 1988
Modeling offline preferences (2)

- Jurafsky ranks parses as the *product* of constituent and valence probabilities:

  \[
  \begin{array}{c}
  \text{root} \\
  \text{keep}
  \end{array}
  \begin{array}{c}
  \text{valence} \\
  <\text{NP}, \text{XP [pred +]}>
  \end{array}
  \begin{array}{c}
  \text{.81}
  \end{array}
  \]

  \[
  \begin{array}{c}
  \text{VP} \\
  \text{V}
  \end{array}
  \begin{array}{c}
  \text{NP} \\
  \text{the dogs}
  \end{array}
  \begin{array}{c}
  \text{PP} \\
  \text{on the beach}
  \end{array}
  \begin{array}{c}
  [\text{.15}] \text{VP} \rightarrow \text{V NP XP}
  \end{array}
  \]

  

  (a) \[.15 \times .81 = .12 \text{ (preferred)}\]

  \[
  \begin{array}{c}
  \text{root} \\
  \text{keep}
  \end{array}
  \begin{array}{c}
  \text{valence} \\
  <\text{NP}> \text{.19}
  \end{array}
  \]

  \[
  \begin{array}{c}
  \text{VP} \\
  \text{V}
  \end{array}
  \begin{array}{c}
  \text{NP} \\
  \text{the dogs}
  \end{array}
  \begin{array}{c}
  \text{PP} \\
  \text{on the beach}
  \end{array}
  \begin{array}{c}
  [\text{.39}] \text{VP} \rightarrow \text{V NP}
  \end{array}
  \begin{array}{c}
  [\text{.14}] \text{NP} \rightarrow \text{NP Postmodifier}
  \end{array}
  \]

  (b) \[.19 \times .39 \times .14 = .01 \text{ (dispreferred)}\]
Modeling offline preferences (3)

(a) \(0.15 \times 0.24 = 0.036\) (disfavored)

(b) \(0.76 \times 0.39 \times 0.14 = 0.041\) (preferred)
Result

• Ranking with respect to parse probability matches offline preferences
• Note that only monotonicity, not degree of preference is matched
Modeling online parsing

Frazier and Rayner 1987
Modeling online parsing

- Does this sentence make sense?

*Frazier and Rayner 1987*
Modeling online parsing

- Does this sentence make sense?
  *The complex houses married and single students and their families.*

*Frazier and Rayner 1987*
Modeling online parsing

• Does this sentence make sense?
  The complex houses married and single students and their families.

• How about this one?

Frazier and Rayner 1987
Modeling online parsing

• Does this sentence make sense?
  The complex houses married and single students and their families.

• How about this one?
  The warehouse fires a dozen employees each year.

Frazier and Rayner 1987
Modeling online parsing

• Does this sentence make sense?
  The complex houses married and single students and their families.

• How about this one?
  The warehouse fires a dozen employees each year.

• And this one?

Frazier and Rayner 1987
Modeling online parsing

• Does this sentence make sense?
  *The complex houses married and single students and their families.*

• How about this one?
  *The warehouse fires a dozen employees each year.*

• And this one?
  *The warehouse fires destroyed all the buildings.*
Modeling online parsing

• Does this sentence make sense?
  The complex houses married and single students and their families.

• How about this one?
  The warehouse fires a dozen employees each year.

• And this one?
  The warehouse fires destroyed all the buildings.

• *fires* can be either a noun or a verb. So can *houses*:

*Frazier and Rayner 1987*
Modeling online parsing

• Does this sentence make sense?
  The complex houses married and single students and their families.

• How about this one?
  The warehouse fires a dozen employees each year.

• And this one?
  The warehouse fires destroyed all the buildings.

• fires can be either a noun or a verb. So can houses:
  \([np \text{ The complex}] [vp \text{ houses married and single students…}]\).

Frazier and Rayner 1987
Does this sentence make sense?
*The complex houses married and single students and their families.*

How about this one?
*The warehouse fires a dozen employees each year.*

And this one?
*The warehouse fires destroyed all the buildings.*

*fires* can be either a noun or a verb. So can *houses*:  

\[ [NP \text{The complex}] [VP \text{houses married and single students...}]. \]

These are *garden path* sentences

*Frazier and Rayner 1987*
Modeling online parsing

• Does this sentence make sense?
  *The complex houses married and single students and their families.*

• How about this one?
  *The warehouse fires a dozen employees each year.*

• And this one?
  *The warehouse fires destroyed all the buildings.*

• *fires* can be either a noun or a verb. So can *houses*:
  \[\text{[NP The complex]} \text{[VP houses married and single students…]}\].

• These are *garden path* sentences

• Originally taken as some of the strongest evidence for *serial* processing by the human parser

  *Frazier and Rayner 1987*
Limited parallel parsing

- Full-serial: keep only one incremental interpretation
- Full-parallel: keep all incremental interpretations
- Limited parallel: keep some but not all interpretations
- In a limited parallel model, garden-path effects can arise from the discarding of a needed interpretation

\[ S \left[ NP \text{ The complex} \right] \left[ VP \text{ houses} \ldots \right] \ldots \] \hspace{1cm} \text{discarded}

\[ S \left[ NP \text{ The complex houses} \ldots \right] \ldots \] \hspace{1cm} \text{kept}
Modeling online parsing: garden paths

- **Pruning** strategy for limited ranked-parallel processing
  - Each incremental analysis is ranked
  - Analyses falling below a threshold are discarded
  - In this framework, a model must characterize
    - The incremental analyses
    - The threshold for pruning
- Jurafsky 1996: partial context-free parses as analyses
- *Probability ratio* as pruning threshold
  - Ratio defined as $P(I) : P(I_{best})$
- (Gibson 1991: *complexity ratio* for pruning threshold)
Garden path models 1: N/V ambiguity
Garden path models 1: N/V ambiguity

- Each analysis is a partial PCFG tree
Garden path models 1: N/V ambiguity

- Each analysis is a partial PCFG tree
- *Tree prefix probability* used for ranking of analysis
Garden path models 1: N/V ambiguity

- Each analysis is a partial PCFG tree
- *Tree prefix probability* used for ranking of analysis

```
S
   /   
  NP   VP
     / |
   DT NN V
   the complex houses
```
Garden path models 1: N/V ambiguity

- Each analysis is a partial PCFG tree
- *Tree prefix probability* used for ranking of analysis

```
S
   NP          VP
  /      \    /     \
DT   NN      V
```

*the complex houses*

*these nodes are actually still undergoing expansion*
Garden path models 1: N/V ambiguity

- Each analysis is a partial PCFG tree
- *Tree prefix probability* used for ranking of analysis
- Partial rule probs *marginalize* over rule completions

```
S
|NP   | VP   |
|DT   | NN   | V   |
|the complex houses|
```

these nodes are actually still undergoing expansion
Each analysis is a partial PCFG tree

Tree prefix probability used for ranking of analysis

Partial rule probs marginalize over rule completions

\[ P(\text{VP} \rightarrow V \ldots) = \sum_{\alpha} P(\text{VP} \rightarrow V \; \alpha) \]
N/V ambiguity (2)

- Partial CF tree analysis of *the complex houses*...

- Analysis of *houses* as noun has much lower probability than analysis as verb (> 250:1)

- Hypothesis: the low-ranking alternative is discarded
N/V ambiguity (3)

• Note that top-down vs. bottom-up questions are immediately implicated, in theory

• Jurafsky includes the cost of generating the initial NP under the S
  • of course, it’s a small cost as $P(S \rightarrow NP \ldots) = 0.92$

• If parsing were bottom-up, that cost would not have been explicitly calculated yet
Garden path models 2
Garden path models 2

- The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)
Garden path models 2

- The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)

  The horse raced past the barn fell.
The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)

*The horse raced past the barn fell.*

(\textit{that was})
Garden path models 2

• The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)
  
  The horse raced past the barn fell.

  *(that was)*

• From the valence + simple-constituency perspective, MC and RRC analyses differ in two places:
The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)

*The horse raced past the barn fell.*

From the valence + simple-constituency perspective, MC and RRC analyses differ in two places:

```
S
  | NP
  |   NP  VP
  |     |
  DT  NN  V
the horse raced
```
Garden path models 2

- The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)

  The horse raced past the barn fell.

  (that was)

- From the valence + simple-constituency perspective, MC and RRC analyses differ in two places:

  ![Diagram of sentence structure]

  \( p = 0.14 \)
Garden path models 2

- The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)
  
  The horse raced past the barn fell. (that was)

- From the valence + simple-constituency perspective, MC and RRC analyses differ in two places:

  transitive valence: $p=0.08$
Garden path models 2

- The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)
  
  \[ The \text{ horse raced past the barn fell. } \]

  \[(that \text{ was})\]

- From the valence + simple-constituency perspective, MC and RRC analyses differ in two places:

  \[\text{transitive valence: } p=0.08\]
Garden path models 2

- The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)
  
  *The horse raced past the barn fell.*
  
  *(that was)*

- From the valence + simple-constituency perspective, MC and RRC analyses differ in two places:

  ![Diagram of the sentence structure](image)

  *transitive valence: $p=0.08$*
Garden path models 2

• The most famous garden-paths: reduced relative clauses (RRCs) versus main clauses (MCs)

  *The horse raced past the barn fell.*

  *(that was)*

• From the valence + simple-constituency perspective, MC and RRC analyses differ in two places:

  - Transitive valence: $p = 0.08$
  - Best intransitive: $p = 0.92$
Garden path models 2, cont.
Garden path models 2, cont.

- 82 : 1 probability ratio means that lower-probability analysis is discarded
Garden path models 2, cont.

- 82 : 1 probability ratio means that lower-probability analysis is discarded
- In contrast, some RRCs do not induce garden paths:
Garden path models 2, cont.

- 82 : 1 probability ratio means that lower-probability analysis is discarded
- In contrast, some RRCs do not induce garden paths: 
  
  *The bird found in the room died.*
82 : 1 probability ratio means that lower-probability analysis is discarded.

In contrast, some RRCs do not induce garden paths:

The bird found in the room died.

Here, the probability ratio turns out to be much closer (≈4 : 1) because found is preferentially transitive.
Garden path models 2, cont.

- 82 : 1 probability ratio means that lower-probability analysis is discarded
- In contrast, some RRCs do not induce garden paths: *The bird found in the room died.*
- Here, the probability ratio turns out to be much closer (≈4 : 1) because *found* is preferentially transitive
- Conclusion within pruning theory: *beam threshold is between 4 : 1 and 82 : 1*
• 82 : 1 probability ratio means that lower-probability analysis is discarded
• In contrast, some RRCs do not induce garden paths: *The bird found in the room died.*
• Here, the probability ratio turns out to be much closer (∼4 : 1) because *found* is preferentially transitive
• Conclusion within pruning theory: *beam threshold is between 4 : 1 and 82 : 1*
• (granularity issue: when exactly does probability cost of valence get paid???)
• Jurafsky 1996 is a product-of-experts (PoE) model
  \[ P(X) = \frac{1}{Z} \prod_i P_i(X) \]
  • Expert 1: the constituency model
  • Expert 2: the valence model

• PoEs are flexible and easy to define, but hard to learn
  • The Jurafsky 1996 model is actually deficient (loses probability mass), due to relative frequency estimation

\[ \sum_i P(\text{valence}_i | \text{discuss}) = P(\text{NP PP | discuss})P(\text{VP} \rightarrow \text{V NP XP}) + P(\text{NP | discuss})P(\text{VP} \rightarrow \text{V NP}) \]
\[ = 0.15 \times 0.24 + 0.76 \times 0.39 \]
\[ = 0.036 + 0.2964 \leq 1 \]
Notes on the probabilistic model (2)

• Jurafsky 1996 predated most work on lexicalized parsers (Collins 1999, Charniak 1997)

• In a generative lexicalized parser, valence and constituency are often combined through decomposition & Markov assumptions, e.g.,

\[
P(\text{valence, verb}|\text{VP}) = P(\text{head} = \text{verb}|\text{VP})P(\text{valence}|\text{VP, verb})
\]

sometimes approximated as

\[
P(\text{valence}|\text{VP})
\]

• The use of decomposition makes it easy to learn non-deficient models
Jurafsky 1996 & pruning: main points

- Syntactic comprehension is probabilistic
- Offline preferences explained by syntactic + valence probabilities
- Online garden-path results explained by same model, when beam search/pruning is assumed
General issues

- What is the granularity of incremental analysis?
  - In \( [\text{NP } \text{the complex houses}] \), \textit{complex} could be an adjective (=\textit{the houses are complex})
  - \textit{complex} could also be a noun (=\textit{the houses of the complex})
  - Should these be distinguished, or combined?
  - When does valence probability cost get paid?
- What is the criterion for abandoning an analysis?
- Should the \textit{number} of maintained analyses affect processing difficulty as well?
Generalizing incremental disambiguation
Generalizing incremental disambiguation

- Another type of uncertainty:
Generalizing incremental disambiguation

- Another type of uncertainty:

*The old man stopped and stared at the*
Generalizing incremental disambiguation

• Another type of uncertainty:

    The old man stopped and stared at the statue?
Generalizing incremental disambiguation

- Another type of uncertainty:

  The old man stopped and stared at the statue? dog?
Another type of uncertainty:

The old man stopped and stared at the statue? dog? view?
Generalizing incremental disambiguation

• Another type of uncertainty:

_The old man stopped and stared at the statue? dog? view? woman?_
Generalizing incremental disambiguation

• Another type of uncertainty:

*The old man stopped and stared at the statue? dog? view? woman?*

*The squirrel stored some nuts in the*
Generalizing incremental disambiguation

- Another type of uncertainty:

  *The old man stopped and stared at the **statue? dog? view? woman?***

  *The squirrel stored some nuts in the **tree***
Generalizing incremental disambiguation

• Another type of uncertainty:

The old man stopped and stared at the statue? dog? view? woman?

The squirrel stored some nuts in the tree

• This is uncertainty about what has not yet been said
Generalizing incremental disambiguation

- Another type of uncertainty:

The old man stopped and stared at the statue? dog? view? woman?

The squirrel stored some nuts in the tree

- This is uncertainty about what has not yet been said
- Reading-time (Ehrlich & Rayner, 1981) and EEG (Kutas & Hillyard, 1980, 1984) evidence shows this affects processing rapidly
Generalizing incremental disambiguation

• Another type of uncertainty:

The old man stopped and stared at the statue? dog? view? woman?

The squirrel stored some nuts in the tree

• This is uncertainty about what has not yet been said
• Reading-time (Ehrlich & Rayner, 1981) and EEG (Kutas & Hillyard, 1980, 1984) evidence shows this affects processing rapidly
• A good model should account for expectations about how this uncertainty will be resolved
Generalizing incremental disambiguation

- Another type of uncertainty:

  The old man stopped and stared at the **statue? dog? view? woman?**

  The squirrel stored some nuts in the **tree**

- This is uncertainty about *what has not yet been said*

- Reading-time (Ehrlich & Rayner, 1981) and EEG (Kutas & Hillyard, 1980, 1984) evidence shows this affects processing rapidly

- A good model should account for expectations about how this uncertainty will be resolved
Quantifying probabilistic online processing difficulty
Quantifying probabilistic online processing difficulty

- Let a word’s difficulty be its *surprisal* given its context:

\[
\text{Surprisal}(w_i) \equiv \log \frac{1}{P(w_i|\text{CONTEXT})} \\
\approx \log \frac{1}{P(w_i|w_1\ldots i-1)}
\]

(Hale, 2001, NAACL; Levy, 2008, Cognition)
Quantifying probabilistic online processing difficulty

- Let a word’s difficulty be its surprisal given its context:

$$\text{Surprisal}(w_i) \equiv \log \frac{1}{P(w_i | \text{CONTEXT})}$$

$$\approx \log \frac{1}{P(w_i | w_1 \ldots w_{i-1})}$$

- Captures the expectation intuition: the more we expect an event, the easier it is to process

(Hale, 2001, NAACL; Levy, 2008, Cognition)
Quantifying probabilistic online processing difficulty

• Let a word’s difficulty be its surprisal given its context:

\[
\text{Surprisal}(w_i) \equiv \log \frac{1}{P(w_i|\text{CONTEXT})}
\approx \log \frac{1}{P(w_i|w_1\ldots i-1)}
\]

• Captures the expectation intuition: the more we expect an event, the easier it is to process
  • Brains are prediction engines!

(Hale, 2001, NAACL; Levy, 2008, Cognition)
Quantifying probabilistic online processing difficulty

- Let a word’s difficulty be its *surprisal* given its context:

\[
\text{Surprisal}(w_i) \equiv \log \frac{1}{P(w_i|\text{CONTEXT})} \\
\approx \log \frac{1}{P(w_i|w_1\ldots i-1)}
\]

- Captures the *expectation* intuition: the more we expect an event, the easier it is to process
  - Brains are prediction engines!
- Predictable words are read faster (Ehrlich & Rayner, 1981) and have distinctive EEG responses (Kutas & Hillyard 1980)

(Hale, 2001, NAACL; Levy, 2008, Cognition)
Quantifying probabilistic online processing difficulty

• Let a word’s difficulty be its *surprisal* given its context:

$$\text{Surprisal}(w_i) \equiv \log \frac{1}{P(w_i|\text{CONTEXT})} \approx \log \frac{1}{P(w_i|w_{1\ldots i-1})}$$

• Captures the *expectation* intuition: the more we expect an event, the easier it is to process
  • Brains are prediction engines!
• Predictable words are read faster (Ehrlich & Rayner, 1981) and have distinctive EEG responses (Kutas & Hillyard 1980)
• Combine with probabilistic grammars to give *grammatical expectations*

(Hale, 2001, NAACL; Levy, 2008, Cognition)
The surprisal graph

Surprisal (-log P)

Probability
Syntactic complexity--non-probabilistic

- On the *resource limitation* view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process
Syntactic complexity--non-probabilistic

- On the *resource limitation* view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process

*the reporter who attacked the senator*
Syntactic complexity--non-probabilistic

- On the *resource limitation* view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process

*the reporter who* **attacked** *the senator*
Syntactic complexity--non-probabilistic

- On the *resource limitation* view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process

*the reporter who attacked the senator*
Syntactic complexity--non-probabilistic

- On the resource limitation view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process

Processing

the reporter who attacked the senator
Syntactic complexity--non-probabilistic

- On the *resource limitation* view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process

Processing

the reporter who **attacked** the senator

Easy
Syntactic complexity--non-probabilistic

- On the *resource limitation* view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process

\[
\text{the reporter who attacked the senator}
\]

\[
\text{the reporter who the senator attacked}
\]
Syntactic complexity--non-probabilistic

- On the *resource limitation* view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process
Syntactic complexity--non-probabilistic

- On the resource limitation view, memory demands are a “processing bottleneck”
- Gibson 1998, 2000 (DLT): multiple and/or more distant dependencies are harder to process

Processing

the reporter who **attacked** the senator

the reporter who the senator **attacked**

Easy

Hard
Expectations versus memory

• Suppose you know that some event class X has to happen in the future, but you don’t know:
  1. When X is going to occur
  2. Which member of X it’s going to be
• The things W you see before X can give you hints about (1) and (2)
  • If expectations facilitate processing, then seeing W should generally speed processing of X
• But you also have to keep W in memory and retrieve it at X
  • This could slow processing at X

NP-head RelPro Verb (a) NP-head RelPro Dep2 Verb (b)
What happens in German final-verb processing?

- Variation in pre-verbal dependency structure also found in verb-final clauses such as in German

Die Einsicht, dass der Freund dem Kunden das Auto aus Plastik verkaufte, erheiterte die Anderen.

The insight, that the friend the client the car of plastic sold, amused the others.
What happens in German final-verb processing?

- Variation in pre-verbal dependency structure also found in verb-final clauses such as in German

Die Einsicht, dass der Freund dem Kunden das Auto aus Plastik verkaufte, erheiterte die Anderen.
The insight, that the friend sold, amused the others.
What happens in German final-verb processing?

...daß der Freund DEM Kunden das Auto verkaufte

‘...that the friend sold the client a car...’

(Konieczny & Döring 2003)
What happens in German final-verb processing?

...daß der Freund DEM Kunden das Auto verkaufte

‘...that the friend sold the client a car...’

(Konieczny & Döring 2003)
What happens in German final-verb processing?

...daß der Freund DEM Kunden das Auto verkauft

‘...that the friend sold the client a car...’

(Konieczny & Döring 2003)
What happens in German final-verb processing?

...daß der Freund DEM Kunden das Auto verkaufte

‘...that the friend sold the client a car...’

(Konieczny & Döring 2003)
What happens in German final-verb processing?

...daß der Freund DEM Kunden das Auto verkaufte

‘...that the friend sold the client a car...’

(Konieczny & Döring 2003)
What happens in German final-verb processing?

...dass der Freund DEM Kunden das Auto verkauft...

'...that the friend sold the client a car...'

...dass der Freund DES Kunden das Auto verkauft...

'...that the friend of the client sold a car...'

(Konieczny & Döring 2003)
What happens in German final-verb processing?

... daß der Freund DEM Kunden das Auto verkauft

‘...that the friend sold the client a car...’

(Konieczny & Döring 2003)
What happens in German final-verb processing?

... daß der Freund DEM Kunden das Auto verkauft.

‘...that the friend sold the client a car...’

Konieczny & Döring 2003
What happens in German final-verb processing?

 Locality: final verb read faster in *DES* condition
 Observed: final verb read faster in *DEM* condition

(Konieczny & Döring 2003)
daß
daß

SBAR

COMP

daß

SBAR

COMP

daß
daß der Freund

Next:
NP\text{nom}
NP\text{acc}
NP\text{dat}
PP
ADVP
Verb

Next:
NP\text{nom}
NP\text{acc}
NP\text{dat}
PP
ADVP
Verb
daß der Freund

Next:
NP
NP acc
NP dat
PP
ADVP
Verb

Next:
NP
NP acc
NP dat
PP
ADVP
Verb
daß der Freund DEM Kunden

daß der Freund DES Kunden
der Freund

Kunden
daß der Freund DEM Kunden das Auto

daß der Freund DES Kunden das Auto
verbuchte

daß der Freund DEM Kunden das Auto verkaufte

Next:

NP nom
NP acc
NP dat
PP
ADVP
Verb
Verkauft der Freund das Auto?

Verkauft der Freund des Kunden das Auto?
Der Freund verkaufte das Auto dem Kunden.
### Model results

<table>
<thead>
<tr>
<th></th>
<th>Reading time (ms)</th>
<th>P(wᵢ): word probability</th>
<th>Locality-based predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>dem Kunden</em></td>
<td>555</td>
<td>8.38×10⁻⁸</td>
<td>slower</td>
</tr>
<tr>
<td><em>des Kunden</em></td>
<td>793</td>
<td>6.35×10⁻⁸</td>
<td>faster</td>
</tr>
</tbody>
</table>

~30% greater expectation in dative condition

once again, wrong monotonicity
Garden-pathing and surprisal

When the dog scratched the vet and his new assistant removed the muzzle.
Garden-pathing and surprisal

• Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

• Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

• Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

- Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

• Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

difficulty here (68ms/char)

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

Compare with:

When the dog scratched, the vet and his new assistant removed the muzzle.

When the dog scratched its owner the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

• Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

• Compare with:

When the dog scratched, the vet and his new assistant removed the muzzle.

When the dog scratched its owner the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

• Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

- difficulty here (68ms/char)

• Compare with:

When the dog scratched, the vet and his new assistant removed the muzzle.

When the dog scratched its owner the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

• Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

• Compare with:

When the dog scratched, the vet and his new assistant removed the muzzle.

When the dog scratched its owner the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
Garden-pathing and surprisal

- Here’s another type of local syntactic ambiguity

When the dog scratched the vet and his new assistant removed the muzzle.

- Compare with:

When the dog scratched, the vet and his new assistant removed the muzzle.

When the dog scratched its owner the vet and his new assistant removed the muzzle.

(Frazier & Rayner, 1982)
How is surprisal efficiently computed?

• To understand this, we’ll look at a different set of slides…
We just learned how to calculate the *probability of a tree*

The *probability of a string* $w_1...n$ is the sum of the probabilities of all trees whose yield is $w_1...n$

The *probability of a string prefix* $w_1...i$ is the sum of the probabilities of all trees whose yield begins with $w_1...i$

If we had the probabilities of two string prefixes $w_1...i−1$ and $w_1...i$, we could calculate the conditional probability $P(w_i|w_1...i−1)$ as their ratio:

$$P(w_i|w_1...i−1) = \frac{P(w_1...i)}{P(w_1...i−1)}$$
Consider the following noun-phrase grammar:

\[
\begin{align*}
2 & \quad \text{NP} \rightarrow \text{Det} \text{ N} \\
3 & \quad \text{NP} \rightarrow \text{NP} \ \text{PP} \\
1 & \quad \text{PP} \rightarrow \text{P} \ \text{NP} \\
1 & \quad \text{Det} \rightarrow \text{the} \\
2 & \quad \text{N} \rightarrow \text{dog} \\
1 & \quad \text{P} \rightarrow \text{near} \\
3 & \quad \text{N} \rightarrow \text{cat}
\end{align*}
\]
Consider the following noun-phrase grammar:

\[
\begin{align*}
\frac{2}{3} & \quad \text{NP} \rightarrow \text{Det N} \\
\frac{2}{3} & \quad \text{NP} \rightarrow \text{NP PP} \\
\frac{1}{3} & \quad \text{PP} \rightarrow \text{P NP} \\
\text{Det} & \rightarrow \text{the} \\
\text{N} & \rightarrow \text{dog} \\
\text{N} & \rightarrow \text{cat} \\
\text{P} & \rightarrow \text{near}
\end{align*}
\]

Question: given a sentence starting with \textit{the}... what is the probability that the next word is \textit{dog}?
Consider the following noun-phrase grammar:

\[
\begin{align*}
2 & \quad \text{NP} \rightarrow \text{Det N} \\
\frac{2}{3} & \quad \text{NP} \rightarrow \text{NP PP} \\
1 & \quad \text{PP} \rightarrow \text{P NP} \\
\end{align*}
\]

1 \quad \text{Det} \rightarrow \text{the} \\
\frac{2}{3} \quad \text{N} \rightarrow \text{dog} \\
\frac{1}{3} \quad \text{N} \rightarrow \text{cat} \\
1 \quad \text{P} \rightarrow \text{near}

Question: given a sentence starting with \textit{the...}

what is the probability that the next word is \textit{dog}?

Intuitively, the answers to this question should be

\[
P(\text{dog}|\text{the}) = \frac{2}{3}
\]
Consider the following noun-phrase grammar:

\[
\begin{align*}
\frac{2}{3} & : \text{NP} \rightarrow \text{Det N} \\
\frac{1}{3} & : \text{NP} \rightarrow \text{NP PP} \\
1 & : \text{PP} \rightarrow \text{P NP}
\end{align*}
\]

1 \quad \text{Det} \rightarrow \text{the}

\[
\begin{align*}
\frac{2}{3} & : \text{N} \rightarrow \text{dog} \\
\frac{1}{3} & : \text{N} \rightarrow \text{cat} \\
1 & : \text{P} \rightarrow \text{near}
\end{align*}
\]

Question: given a sentence starting with \textit{the}…

what is the probability that the next word is \textit{dog}?

Intuitively, the answers to this question should be

\[
P(\text{dog} | \text{the}) = \frac{2}{3}
\]

because the second word HAS to be either \textit{dog} or \textit{cat}.
Inference over infinite tree sets (2)

We “should” just enumerate the trees that cover the dog . . . ,
Inference over infinite tree sets (2)

<table>
<thead>
<tr>
<th>Rule</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP → Det N</td>
<td>Det → the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NP → NP PP</td>
<td>N → dog</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PP → P NP</td>
<td>N → cat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P → near</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We “should” just enumerate the trees that cover the dog . . . , and divide their total probability by that of the . . .
Inference over infinite tree sets (2)

We “should” just enumerate the trees that cover *the dog* . . . , and divide their total probability by that of *the* . . .

. . . but there are infinitely many trees.
Inference over infinite tree sets (2)

- We “should” just enumerate the trees that cover *the dog* . . . , and divide their total probability by that of *the* . . .
- . . . but there are infinitely many trees.
Shortcut 1: you can think of a *partial* tree as marginalizing over all completions of the partial tree. It has a corresponding marginal probability in the PCFG.
Shortcut 1: you can think of a *partial* tree as marginalizing over all completions of the partial tree. It has a corresponding marginal probability in the PCFG.
Shortcut 1: you can think of a *partial* tree as marginalizing over all completions of the partial tree. It has a corresponding marginal probability in the PCFG.
Problem 2: there are still an infinite number of incomplete trees covering a partial input.
Problem 2: there are still an infinite number of incomplete trees covering a partial input.

BUT! These tree probabilities form a geometric series:

\[
P(\text{the dog . . .}) = \frac{4}{9} + \frac{4}{27} + \frac{4}{81} + \frac{4}{243} + \cdots
\]
Problem 2: there are still an infinite number of incomplete trees covering a partial input.

\[ P(\text{the dog} \ldots) = \frac{4}{9} + \frac{4}{27} + \frac{4}{81} + \frac{4}{243} + \cdots \]

\[ = \frac{4}{9} \sum_{i=0}^{\infty} \left(\frac{1}{3}\right)^i \]
Problem 2: there are still an infinite number of incomplete trees covering a partial input.

BUT! These tree probabilities form a geometric series:

\[ P(\text{the dog} \ldots) = \frac{4}{9} + \frac{4}{27} + \frac{4}{81} + \frac{4}{243} + \cdots \]

\[ = \frac{4}{9} \sum_{i=0}^{\infty} \left( \frac{1}{3} \right)^i \]

\[ = \frac{2}{3} \]
Problem 2: there are still an infinite number of incomplete trees covering a partial input.

\[
P(\text{the dog . . .}) = \frac{4}{9} + \frac{4}{27} + \frac{4}{81} + \frac{4}{243} + \cdots
\]

\[
= \frac{4}{9} \sum_{i=0}^{\infty} \left(\frac{1}{3}\right)^i
\]

\[
= \frac{2}{3}
\]

\[\frac{2}{3} N \rightarrow \text{dog}\]

BUT! These tree probabilities form a geometric series:
Generalizing the geometric series induced by rule recursion

In general, these infinite tree sets arise due to *left recursion* in a probabilistic grammar

\[ A \rightarrow B \alpha \]
\[ B \rightarrow A \beta \]

*(Stolcke, 1995)*
Generalizing the geometric series induced by rule recursion

In general, these infinite tree sets arise due to *left recursion* in a probabilistic grammar

\[ A \rightarrow B \alpha \]

\[ B \rightarrow A \beta \]

We can formulate a stochastic *left-corner matrix* of transitions between categories:

\[
P_L = \begin{array}{ccccc}
A & B & \ldots & K \\
A & 0.3 & 0.7 & \ldots & 0 \\
B & 0.1 & 0.1 & \ldots & 0.2 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
K & 0.2 & 0.1 & \ldots & 0.2 \\
\end{array}
\]

(Stolcke, 1995)
Generalizing the geometric series induced by rule recursion

In general, these infinite tree sets arise due to *left recursion* in a probabilistic grammar

\[ A \rightarrow B \alpha \]
\[ B \rightarrow A \beta \]

We can formulate a stochastic *left-corner matrix* of transitions between categories:

\[
P_L = \begin{pmatrix}
A & B & \ldots & K \\
A & 0.3 & 0.7 & \ldots & 0 \\
B & 0.1 & 0.1 & \ldots & 0.2 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
K & 0.2 & 0.1 & \ldots & 0.2
\end{pmatrix}
\]

and solve for its closure \( R_L = (I - P_L)^{-1} \).

*(Stolcke, 1995)*
Generalizing the geometric series

1  ROOT → NP  1  Det → the
2/3  NP → Det N  2/3  N → dog
1/3  NP → NP PP  1/3  N → cat
1  PP → P NP  1  P → near

The closure of our left-corner matrix is

\[
R_L = \begin{bmatrix}
1 & \frac{3}{2} & 0 & 1 & 0 & 0 & 0 \\
0 & \frac{3}{2} & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]
Generalizing the geometric series

1. \( \text{ROOT} \rightarrow \text{NP} \)  
2. \( \frac{2}{3} \) \( \text{NP} \rightarrow \text{Det N} \)  
3. \( \frac{1}{3} \) \( \text{NP} \rightarrow \text{NP PP} \)  
1. \( \text{PP} \rightarrow \text{P NP} \)

1. \( \text{Det} \rightarrow \text{the} \)  
2. \( \frac{2}{3} \) \( \text{N} \rightarrow \text{dog} \)  
3. \( \frac{1}{3} \) \( \text{N} \rightarrow \text{cat} \)  
1. \( \text{P} \rightarrow \text{near} \)

- The closure of our left-corner matrix is

\[
R_L = \begin{pmatrix}
1 & \frac{3}{2} & 0 & 1 & 0 & 0 & 0 \\
0 & \frac{3}{2} & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]

- Refer to an entry \((X, Y)\) in this matrix as \(R(X \Rightarrow_L Y)\)
Generalizing the geometric series

\[ 1 \text{ ROOT } \rightarrow \text{ NP} \]
\[ 2^{\frac{1}{3}} \text{ NP } \rightarrow \text{ Det N} \]
\[ 3^{\frac{1}{3}} \text{ NP } \rightarrow \text{ NP PP} \]
\[ 1 \text{ PP } \rightarrow \text{ P NP} \]
\[ 1 \text{ Det } \rightarrow \text{ the} \]
\[ 2^{\frac{1}{3}} \text{ N } \rightarrow \text{ dog} \]
\[ 3^{\frac{1}{3}} \text{ N } \rightarrow \text{ cat} \]
\[ 1 \text{ P } \rightarrow \text{ near} \]

The closure of our left-corner matrix is

\[ R_L = \begin{pmatrix}
1 & 3^{\frac{1}{2}} & 0 & 1 & 0 & 0 & 0 \\
0 & 3^{\frac{1}{2}} & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix} \]

Refer to an entry \((X, Y)\) in this matrix as \(R(X \Rightarrow L Y)\)

Note that the \(3^{\frac{1}{2}}\) “bonus” accrued for left-recursion of NPs appears in the \((\text{ROOT},\text{NP})\) and \((\text{NP},\text{NP})\) cells of the matrix
Generalizing the geometric series

| 1  | ROOT → NP       | 1  | Det → the |
| 2  | NP → Det N      | 2  | N → dog   |
| 3  | NP → NP PP      | 3  | N → cat   |
| 1  | PP → P NP       | 1  | P → near  |

▶ The closure of our left-corner matrix is

\[
R_L = \begin{pmatrix}
1 & 3^2 & 0 & 1 & 0 & 0 & 0 \\
0 & 3^2 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]

▶ Refer to an entry \((X, Y)\) in this matrix as \(R(X \Rightarrow_L Y)\)

▶ Note that the \(3^2\) “bonus” accrued for left-recursion of NPs appears in the (ROOT,NP) and (NP,NP) cells of the matrix

▶ We need to do the same with unary chains, constructing a unary-closure matrix \(R_U\).
Efficient incremental parsing: the probabilistic Earley algorithm

We can use the Earley algorithm (Earley, 1970) in a probabilistic incarnation (Stolcke, 1995) to deal with these infinite tree sets.

The (slightly oversimplified) probabilistic Earley algorithm has two fundamental types of operations:

- **Prediction**: if $Y$ is a possible goal, and $Y$ can lead to $Z$ through a left corner, choose a rule $Z \rightarrow \alpha$ and set up $\alpha$ as a new sequence of possible goals.

- **Completion**: if $Y$ is a possible goal, $Y$ can lead to $Z$ through unary rewrites, and we encounter a completed $Z$, absorb it and move on to the next sub-goal in the sequence.
Efficient incremental parsing: the probabilistic Earley algorithm

- Parsing consists of constructing a chart of states (items)
- A state has the following structure:

\[ X \rightarrow \alpha \circ \beta \]

- The forward probability is the total probability of getting from the root at the start of the sentence through to this state
- The inside probability is the “bottom-up” probability of the state
Efficient incremental parsing: the probabilistic Earley algorithm

Inference rules for probabilistic Earley:

- **Prediction:**

  \[
  X \rightarrow \beta \circ Y \gamma \\
  p \quad q
  \]

  \[
  a : R(Y \Rightarrow^*_L Z) \\
  b : Z \rightarrow \alpha
  \]

  \[
  Z \rightarrow \circ \alpha
  \]

  \[
  abp \quad b
  \]
Efficient incremental parsing: the probabilistic Earley algorithm

Inference rules for probabilistic Earley:

- **Prediction:**
  \[
  X \rightarrow \beta \circ Y \gamma \\
  p \quad q \\
  a : R(Y \Rightarrow_{L} Z) \quad b : Z \rightarrow \alpha \\
  Z \rightarrow \circ \alpha \\
  abp \quad b
  \]

- **Completion:**
  \[
  X \rightarrow \beta \circ Y \gamma \\
  p \quad q \\
  a : R(Y \Rightarrow_{U} Z) \quad b \quad c \\
  Z \rightarrow \alpha \circ \\
  X \rightarrow \beta \gamma \\
  acp \quad acq
  \]
Efficient incremental parsing: probabilistic Earley
Efficient incremental parsing: probabilistic Earley
Efficient incremental parsing: probabilistic Earley

Det \rightarrow \text{the} \\
1 \quad 1 \\

NP \rightarrow \text{Det N} \\
\frac{2}{3} \times \frac{3}{2} = \frac{2}{3} \\

NP \rightarrow \text{NP PP} \\
\frac{1}{3} \times \frac{3}{2} = \frac{1}{3} \\

ROOT \rightarrow \text{NP} \\
1 \quad 1 \\

the \quad dog \quad near \quad the
Efficient incremental parsing: probabilistic Earley

Det $\rightarrow$ the
1 1

NP $\rightarrow$ Det N
$\frac{2}{3} \times \frac{3}{2} = \frac{2}{3}$

NP $\rightarrow$ NP PP
$\frac{1}{3} \times \frac{3}{2} = \frac{1}{3}$

ROOT $\rightarrow$ NP
1 1

the dog near the
Efficient incremental parsing: probabilistic Earley

Det \rightarrow \text{the}
1 \quad 1

NP \rightarrow \text{Det N}
\frac{2}{3} \times \frac{3}{2} = \frac{2}{3}

NP \rightarrow \text{NP PP}
\frac{1}{3} \times \frac{3}{2} = \frac{1}{3}

ROOT \rightarrow \text{NP}
1 \quad 1

NP \rightarrow \text{Det N}
1 \quad \frac{2}{3}

Det \rightarrow \text{the}
1 \quad 1

the \quad \text{dog} \quad \text{near} \quad \text{the}
Efficient incremental parsing: probabilistic Earley

\[
\begin{align*}
\text{Det} & \rightarrow \text{o the} \\
1 & \quad 1 \\
\text{NP} & \rightarrow \text{o Det N} \\
\frac{2}{3} & \times \frac{3}{2} \\
\text{NP} & \rightarrow \text{o NP PP} \\
\frac{1}{3} & \times \frac{3}{2} \\
\text{ROOT} & \rightarrow \text{o NP} \\
1 & \quad 1 \\
\end{align*}
\]
Efficient incremental parsing: probabilistic Earley

Det → o the
1   1

NP → o Det N
2/3 × 3/2  2/3

NP → o NP PP
1/3 × 3 2/3  1/3

ROOT → o NP
1   1

N → o cat
1/3   1/3

N → o dog
2/3   2/3

NP → Det o N
1   2/3

Det → the o
1   1

the      dog      near      the
Efficient incremental parsing: probabilistic Earley

\[
\begin{align*}
\text{Det} & \rightarrow \text{the} \\
1 & \quad 1 \\
\text{NP} & \rightarrow \text{Det} \; \text{N} \\
\frac{2}{3} \times \frac{3}{2} & \quad \frac{2}{3} \\
\text{NP} & \rightarrow \text{NP} \; \text{PP} \\
\frac{1}{3} \times \frac{3}{2} & \quad \frac{1}{3} \\
\text{ROOT} & \rightarrow \text{NP} \\
1 & \quad 1 \\
\text{N} & \rightarrow \text{cat} \\
\frac{1}{3} & \quad \frac{1}{3} \\
\text{N} & \rightarrow \text{dog} \\
\frac{2}{3} & \quad \frac{2}{3} \\
\text{NP} & \rightarrow \text{Det} \; \text{N} \\
1 & \quad \frac{2}{3} \\
\text{Det} & \rightarrow \text{the} \\
1 & \quad 1
\end{align*}
\]
Efficient incremental parsing: probabilistic Earley

Det → °the
1 1

NP → °Det N
\( \frac{2}{3} \times \frac{3}{2} \times \frac{2}{3} \)

NP → °NP PP
\( \frac{1}{3} \times \frac{3}{2} \times \frac{1}{3} \)

ROOT → °NP
\( \frac{1}{3} \times \frac{1}{3} \)

N → °cat
\( \frac{1}{3} \)

N → °dog
\( \frac{2}{3} \)

NP → °Det °N
\( \frac{2}{3} \)

Det → °the
1 1

NP → °Det °N
\( \frac{2}{3} \)

N → °dog
\( \frac{2}{3} \)

the dog near the
Efficient incremental parsing: probabilistic Earley

Det $\rightarrow$ the
1 1

NP $\rightarrow$ Det N
$\frac{2}{3} \times \frac{3}{2} = \frac{2}{3}$

NP $\rightarrow$ NP PP
$\frac{1}{3} \times \frac{3}{2} = \frac{1}{3}$

ROOT $\rightarrow$ NP
1 1

NP $\rightarrow$ Det N
1 $\frac{2}{3}$

Det $\rightarrow$ the
1 1

N $\rightarrow$ cat
$\frac{1}{3} \times \frac{1}{3}$

N $\rightarrow$ dog
$\frac{2}{3} \times \frac{2}{3}$

the
dog
near
do the
Efficient incremental parsing: probabilistic Earley

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Det → the</td>
<td>(1 \times 1)</td>
</tr>
<tr>
<td>NP → Det (\circ) N</td>
<td>(\frac{2}{9} \times \frac{4}{9})</td>
</tr>
<tr>
<td>NP → NP (\circ) PP</td>
<td>(\frac{2}{3} \times \frac{4}{27})</td>
</tr>
<tr>
<td>NP → NP (\circ) PP</td>
<td>(\frac{1}{3} \times \frac{1}{3})</td>
</tr>
<tr>
<td>ROOT → NP</td>
<td>(1 \times 1)</td>
</tr>
<tr>
<td>NP → Det (\circ) N</td>
<td>(\frac{2}{3})</td>
</tr>
<tr>
<td>Det → the</td>
<td>(1 \times 1)</td>
</tr>
<tr>
<td>N → cat</td>
<td>(\frac{1}{3} \times \frac{1}{3})</td>
</tr>
<tr>
<td>N → dog</td>
<td>(\frac{2}{3} \times \frac{2}{3})</td>
</tr>
<tr>
<td>N → dog</td>
<td>(\frac{2}{3} \times \frac{2}{3})</td>
</tr>
</tbody>
</table>

The diagram shows the probabilistic Earley parsing process for the sentence "the dog near the the."
Efficient incremental parsing: probabilistic Earley

\[
\begin{align*}
\text{Det} & \rightarrow \text{the} & \text{NP} & \rightarrow \text{NP} \circ \text{PP} \\
1 & 1 & \frac{2}{9} & \frac{4}{27} \\
\frac{2}{3} \times \frac{3}{2} \times \frac{2}{3} & \frac{2}{3} & \text{NP} & \rightarrow \text{Det} \circ \text{N} \\
\frac{1}{3} \times \frac{3}{2} \times \frac{1}{3} & \text{NP} & \rightarrow \text{NP} \circ \text{PP} \\
1 & 1 & \frac{1}{3} & \frac{1}{3} \\
\text{ROOT} & \rightarrow \text{ NP} \\
\frac{4}{9} & \frac{4}{9} \\
\text{ROOT} & \rightarrow \text{ NP} \\
\frac{1}{3} & \frac{2}{3} \\
\text{Det} & \rightarrow \text{the} \\
1 & 1 & \frac{2}{3} & \frac{2}{3} \\
\text{NP} & \rightarrow \text{Det} \circ \text{N} \\
\frac{1}{3} & \frac{1}{3} \\
\text{Det} & \rightarrow \text{the} \\
1 & 1 & \frac{2}{3} & \frac{2}{3} \\
\text{N} & \rightarrow \text{dog} \\
\frac{2}{3} & \frac{2}{3} \\
\text{N} & \rightarrow \text{dog} \\
\frac{2}{3} & \frac{2}{3} \\
\end{align*}
\]
Efficient incremental parsing: probabilistic Earley
Efficient incremental parsing: probabilistic Earley

\[
\begin{align*}
\text{ROOT} & \rightarrow \text{NP} \circ \\
& \quad \frac{4}{9} \quad \frac{4}{9} \\
\text{Det} & \rightarrow \text{the} \\
& \quad 1 \quad 1 \\
\text{NP} & \rightarrow \text{NP} \circ \text{PP} \\
& \quad \frac{2}{9} \quad \frac{4}{27} \\
\text{NP} & \rightarrow \text{Det} \circ \text{N} \\
& \quad \frac{2}{3} \quad \frac{4}{9} \\
\text{NP} & \rightarrow \text{NP} \circ \text{NP} \circ \text{PP} \\
& \quad \frac{1}{3} \times \frac{3}{2} \quad \frac{1}{3} \\
\text{ROOT} & \rightarrow \text{NP} \circ \\
& \quad 1 \quad 1 \\
\text{N} & \rightarrow \text{cat} \\
& \quad \frac{1}{3} \quad \frac{1}{3} \\
\text{N} & \rightarrow \text{dog} \\
& \quad \frac{2}{3} \quad \frac{2}{3} \\
\text{NP} & \rightarrow \text{Det} \circ \text{N} \\
& \quad 1 \quad \frac{2}{3} \\
\text{Det} & \rightarrow \text{the} \circ \\
& \quad 1 \quad 1 \\
\text{N} & \rightarrow \text{dog} \circ \\
& \quad \frac{2}{3} \quad \frac{2}{3} \\
\text{P} & \rightarrow \text{near} \\
& \quad \frac{2}{9} \quad 1 \\
\text{PP} & \rightarrow \text{P} \circ \text{NP} \\
& \quad \frac{2}{9} \quad 1 \\
\end{align*}
\]
Efficient incremental parsing: probabilistic Earley
Efficient incremental parsing: probabilistic Earley

\[
\begin{align*}
\text{Det} & \rightarrow °\text{the} & 1 & 1 \\
\text{NP} & \rightarrow °\text{Det} \text{ N} & \frac{2}{3} \times \frac{3}{2} & \frac{2}{3} \\
\text{NP} & \rightarrow °\text{NP PP} & \frac{1}{3} \times \frac{3}{2} & \frac{1}{3} \\
\text{ROOT} & \rightarrow °\text{NP} & 1 & 1 \\
\text{N} & \rightarrow °\text{cat} & \frac{1}{3} & \frac{1}{3} \\
\text{N} & \rightarrow °\text{dog} & \frac{2}{3} & \frac{2}{3} \\
\text{NP} & \rightarrow °\text{Det} \text{ N} & \frac{2}{9} & \frac{1}{3} \\
\text{NP} & \rightarrow °\text{NP PP} & \frac{2}{9} & \frac{4}{27} \\
\text{ROOT} & \rightarrow °\text{NP} & \frac{4}{9} & \frac{4}{9} \\
\text{P} & \rightarrow °\text{near} & \frac{2}{9} & 1 \\
\text{PP} & \rightarrow °\text{P NP} & \frac{2}{9} & 1 \\
\end{align*}
\]
Efficient incremental parsing: probabilistic Earley

\[
\begin{align*}
\text{det} & \rightarrow \text{the} & \frac{2}{9} \quad \frac{4}{27} \\
\text{np} & \rightarrow \text{np} \circ \text{pp} & \frac{2}{9} \quad \frac{4}{27} \\
\text{np} & \rightarrow \text{np} \circ \text{det} \circ \text{n} & \frac{2}{3} \quad \frac{4}{9} \\
\text{np} & \rightarrow \text{np} \circ \text{n} & \frac{2}{3} \quad \frac{2}{3} \quad \frac{1}{3} \\
\text{root} & \rightarrow \text{np} & \frac{4}{9} \quad \frac{4}{9} \\
\text{np} & \rightarrow \text{det} \circ \text{n} & \frac{2}{9} \quad \frac{4}{27} \\
\text{np} & \rightarrow \text{det} \circ \text{n} & \frac{2}{9} \quad \frac{4}{27} \\
\text{pp} & \rightarrow \text{p} \circ \text{np} & \frac{2}{9} \quad \frac{4}{27} \\
\text{p} & \rightarrow \text{near} & \frac{2}{3} \quad \frac{3}{2} \quad \frac{1}{3} \\
\text{near} & \rightarrow \text{near} & \frac{2}{3} \quad \frac{3}{2} \quad \frac{1}{3} \\
\end{align*}
\]
Efficient incremental parsing: probabilistic Earley

The diagram shows the probabilistic Earley parsing process for the sentence "the dog near the dog." The tree structure is represented with rules for parsing each part of speech, along with their associated probabilities. For example, the root node is marked as "ROOT → NP." Each non-terminal node (NP, Det, N, PP, P) is connected to the root with a probability, and the terminals (the, dog, near, the) are linked to the non-terminals with their respective probabilities.
Efficient incremental parsing: probabilistic Earley
Prefix probabilities from probabilistic Earley

- If you have just processed word $w_i$, then the prefix probability of $w_1...i$ can be obtained by summing all forward probabilities of items that have the form $X \rightarrow \alpha w_i \circ \beta$
Prefix probabilities from probabilistic Earley

If you have just processed word $w_i$, then the prefix probability of $w_1...i$ can be obtained by summing all forward probabilities of items that have the form $X \rightarrow \alpha w_i \circ \beta$

In our example, we see:

$$P(\text{the}) = 1$$
$$P(\text{the dog}) = \frac{2}{3}$$
$$P(\text{the dog near}) = \frac{2}{9}$$
$$P(\text{the dog near the}) = \frac{2}{9}$$
Prefix probabilities from probabilistic Earley

- If you have just processed word $w_i$, then the prefix probability of $w_1...i$ can be obtained by summing all forward probabilities of items that have the form $X \rightarrow \alpha w_i \circ \beta$

- In our example, we see:

  $P(\text{the}) = 1$
  $P(\text{the dog}) = \frac{2}{3}$
  $P(\text{the dog near}) = \frac{2}{9}$
  $P(\text{the dog near the}) = \frac{2}{9}$

- Taking the ratios of these prefix probabilities can give us conditional word probabilities
Probabilistic Earley as an “eager” algorithm

- From the *inside probabilities* of the states on the chart, the posterior distribution on (incremental) trees can be directly calculated.
- This posterior distribution is *precisely* the correct result of the application of Bayes’ rule:

\[
P(T_{\text{incremental}}\mid w_{1...i}) = \frac{P(w_{1...i}, T_{\text{incremental}})}{P(w_{1...i})}
\]

- Hence, probabilistic Earley is also performing rational disambiguation.
- Hale (2001) called this the “eager” property of an incremental parsing algorithm.
Probabilistic Earley algorithm: key ideas

- We want to use probabilistic grammars for both disambiguation and calculating probability distributions over upcoming events.
- Infinitely many trees can be constructed in polynomial time and space.
- The prefix probability of the string is calculated in the process.
- By taking the log-ratio of two prefix probabilities, the surprisal of a word in its context can be calculated.
We want to use probabilistic grammars for both disambiguation and calculating probability distributions over upcoming events.

Infinitely many trees can be constructed in polynomial time ($O(n^3)$) and space ($\Theta$).

The prefix probability of the string is calculated in the process.

By taking the log-ratio of two prefix probabilities, the surprisal of a word in its context can be calculated.
Probabilistic Earley algorithm: key ideas

- We want to use probabilistic grammars for both disambiguation and calculating probability distributions over upcoming events.
- Infinitely many trees can be constructed in polynomial time ($O(n^3)$) and space ($O(n^2)$).
- The *prefix probability* of the string is calculated in the process.
- By taking the log-ratio of two prefix probabilities, the surprisal of a word in its context can be calculated.
Other introductions

- You can read about the (non-probabilistic) Earley algorithm in (Jurafsky and Martin, 2000, Chapter 13)
- Prefix probabilities can also be calculated with an extension of the CKY algorithm due to Jelinek and Lafferty (1991)
A small PCFG for this sentence type

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
<th>Non-terminal</th>
<th>Terminals</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → SBAR S</td>
<td>0.3</td>
<td>Conj → and</td>
<td>1</td>
</tr>
<tr>
<td>S → NP VP</td>
<td>0.7</td>
<td>Det → the</td>
<td>0.8</td>
</tr>
<tr>
<td>SBAR → COMPL S</td>
<td>0.3</td>
<td>Det → its</td>
<td>0.1</td>
</tr>
<tr>
<td>SBAR → COMPL S COMMA</td>
<td>0.7</td>
<td>Det → his</td>
<td>0.1</td>
</tr>
<tr>
<td>COMPL → When</td>
<td>1</td>
<td>N → dog</td>
<td>0.2</td>
</tr>
<tr>
<td>NP → Det N</td>
<td>0.6</td>
<td>N → vet</td>
<td>0.2</td>
</tr>
<tr>
<td>NP → Det Adj N</td>
<td>0.2</td>
<td>N → assistant</td>
<td>0.2</td>
</tr>
<tr>
<td>NP → NP Conj NP</td>
<td>0.2</td>
<td>N → muzzle</td>
<td>0.2</td>
</tr>
</tbody>
</table>

(analysis in Levy, 2011)
A small PCFG for this sentence type

<table>
<thead>
<tr>
<th>Production</th>
<th>Probability</th>
<th>Non-terminal</th>
<th>Terminals</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → SBAR S</td>
<td>0.3</td>
<td>Conj</td>
<td>and, new</td>
</tr>
<tr>
<td>S → NP VP</td>
<td>0.7</td>
<td>Det</td>
<td>the, V, NP</td>
</tr>
<tr>
<td>SBAR → COMPL S</td>
<td>0.3</td>
<td>Det</td>
<td>its, V</td>
</tr>
<tr>
<td>SBAR → COMPL S COMMA</td>
<td>0.7</td>
<td>Det</td>
<td>his, V</td>
</tr>
<tr>
<td>COMPL → When</td>
<td>1</td>
<td>N</td>
<td>dog, removed</td>
</tr>
<tr>
<td>NP → Det N</td>
<td>0.6</td>
<td>N</td>
<td>vet, assistant, muzzle, owner</td>
</tr>
<tr>
<td>NP → Det Adj N</td>
<td>0.2</td>
<td>N</td>
<td>assistant, muzzle, owner</td>
</tr>
<tr>
<td>NP → NP Conj NP</td>
<td>0.2</td>
<td>N</td>
<td>muzzle, owner</td>
</tr>
</tbody>
</table>

(analysis in Levy, 2011)
Two incremental trees
Two incremental trees

• “Garden-path” analysis:
Two incremental trees

• “Garden-path” analysis:

\[
\text{S} \rightarrow \text{SBAR}\rightarrow \text{COMPL}\rightarrow \text{S} \rightarrow \text{NP VP}
\]

\[
\text{When} \rightarrow \text{NP} \rightarrow \text{V} \rightarrow \text{NP} \rightarrow \text{NP Conjunction NP}
\]

\[
\text{the dog scratched} \rightarrow \text{NP} \rightarrow \text{and} \rightarrow \text{Det Adj N}
\]

\[
\text{the vet} \rightarrow \text{Det} \rightarrow \text{his} \rightarrow \text{new assistant}
\]
Two incremental trees

• “Garden-path” analysis:

\[
P(T|w_{1...10}) = 0.826
\]
Two incremental trees

• “Garden-path” analysis:

\[ P(T|w_{1...10}) = 0.826 \]

• Ultimately-correct analysis
Two incremental trees

• “Garden-path” analysis:

$$P(T|w_{1...10}) = 0.826$$

• Ultimately-correct analysis

$$P(T|w_{1...10}) = 0.174$$
Two incremental trees

- “Garden-path” analysis:

\[ P(T|w_{1\ldots10}) = 0.826 \]

- Ultimately-correct analysis

\[ P(T|w_{1\ldots10}) = 0.174 \]
Two incremental trees

- “Garden-path” analysis:

\[ P(T|w_{1...10}) = 0.826 \]

- Ultimately-correct analysis:

\[ P(T|w_{1...10}) = 0.174 \]
Two incremental trees

• “Garden-path” analysis:

Disambiguating word probability marginalizes over incremental trees:

\[ P(T|w_{1...10}) = 0.826 \]

• Ultimately-correct analysis

\[ P(T|w_{1...10}) = 0.174 \]
Two incremental trees

- “Garden-path” analysis:

Disambiguating word probability marginalizes over incremental trees:

$$P(T|w_{1...10}) = 0.826$$

- Ultimately-correct analysis

$$P(T|w_{1...10}) = 0.174$$
Preceding context can disambiguate

• “its owner” takes up the object slot of scratched

<table>
<thead>
<tr>
<th>Condition</th>
<th>Surprisal at Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP absent</td>
<td>4.2</td>
</tr>
<tr>
<td>NP present</td>
<td>2</td>
</tr>
</tbody>
</table>
Sensitivity to verb argument structure

• A superficially similar example:

When the dog arrived the vet and his new assistant removed the muzzle.

(Staub, 2007)
Sensitivity to verb argument structure

- A superficially similar example:

When the dog arrived the vet and his new assistant removed the muzzle.

(Easier here)

(Staub, 2007)
Sensitivity to verb argument structure

• A superficially similar example:

When the dog arrived the vet and his new assistant removed the muzzle.

But harder here!  Easier here

(Staub, 2007)
Sensitivity to verb argument structure

- A superficially similar example:

When the dog arrived the vet and his new assistant removed the muzzle.

But harder here!

Easier here

(c.f. When the dog scratched the vet and his new assistant removed the muzzle.)

(Staub, 2007)
### Modeling argument-structure sensitivity

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
<th>Argument 1</th>
<th>Argument 2</th>
<th>Argument 3</th>
<th>Argument 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → SBAR S</td>
<td>0.3</td>
<td>Conj</td>
<td>and</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>S → NP VP</td>
<td>0.7</td>
<td>Det</td>
<td>the</td>
<td>0.8</td>
<td>VP</td>
</tr>
<tr>
<td>SBAR → COMPL S</td>
<td>0.3</td>
<td>Det</td>
<td>its</td>
<td>0.1</td>
<td>VP</td>
</tr>
<tr>
<td>SBAR → COMPL S COMMA</td>
<td>0.7</td>
<td>Det</td>
<td>his</td>
<td>0.1</td>
<td>V</td>
</tr>
<tr>
<td>COMPL → When</td>
<td>1</td>
<td>N</td>
<td>dog</td>
<td>0.2</td>
<td>V</td>
</tr>
<tr>
<td>NP → Det N</td>
<td>0.6</td>
<td>N</td>
<td>vet</td>
<td>0.2</td>
<td>V</td>
</tr>
<tr>
<td>NP → Det Adj N</td>
<td>0.2</td>
<td>N</td>
<td>assistant</td>
<td>0.2</td>
<td>COMMA</td>
</tr>
<tr>
<td>NP → NP Conj NP</td>
<td>0.2</td>
<td>N</td>
<td>muzzle</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>N</td>
<td>owner</td>
<td>0.2</td>
<td></td>
</tr>
</tbody>
</table>
Modeling argument-structure sensitivity

- The “context-free” assumption doesn’t preclude relaxing probabilistic locality:

(Johnson, 1999; Klein & Manning, 2003)
The “context-free” assumption doesn’t preclude relaxing probabilistic locality:

Modeling argument-structure sensitivity

- The “context-free” assumption doesn’t preclude relaxing probabilistic locality:
# Modeling argument-structure sensitivity

The “context-free” assumption doesn’t preclude relaxing probabilistic locality:

| VP → V NP | 0.5 | VP → Vtrans NP | 0.45 |
| VP → V | 0.5 | VP → Vtrans | 0.05 |
| V → scratched | 0.25 | VP → Vintrans | 0.45 |
| V → removed | 0.25 | VP → Vintrans NP | 0.05 |
| V → arrived | 0.5 | Vtrans → scratched | 0.5 |
| | | Vtrans → removed | 0.5 |
| | | Vintrans → arrived | 1 |

(Johnson, 1999; Klein & Manning, 2003)
Result

When the dog arrived the vet and his new assistant removed the muzzle.

transitivity-distinguishing PCFG

<table>
<thead>
<tr>
<th>Condition</th>
<th>Ambiguity onset</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intransitive (arrived)</td>
<td>2.11</td>
<td>3.20</td>
</tr>
<tr>
<td>Transitive (scratched)</td>
<td>0.44</td>
<td>8.04</td>
</tr>
</tbody>
</table>

When the dog scratched the vet and his new assistant removed the muzzle.
Move to broad coverage

- Instead of the pedagogical grammar, a “broad-coverage” grammar from the parsed Brown corpus (11,984 rules)
- Relative-frequency estimation of rule probabilities ("vanilla" PCFG)
But why would surprisal be the specific function relating probability to processing difficulty?

And is there empirical evidence that surprisal is the right function?

(Smith & Levy, 2013)
Surprisal vs. predictability in general

Surprisal($w_i$) $\equiv \log \frac{1}{P(w_i|\text{CONTEXT})}$

$\approx \log \frac{1}{P(w_i|w_1...i-1)}$

- But why would surprisal be the specific function relating probability to processing difficulty?
- And is there empirical evidence that surprisal is the right function?

(Smith & Levy, 2013)
Theoretical interpretations of surprisal
Theoretical interpretations of surprisal

- Three proposals:
Theoretical interpretations of surprisal

• Three proposals:
  • Relative entropy from prior to posterior distribution in the probabilistic grammar (Levy, 2008)

\[ D(P_i \| P_{i-1}) = \sum_I P_i(I) \log \frac{P_{i-1}(I)}{P_i(I)} \]
Theoretical interpretations of surprisal

- Three proposals:
  - Relative entropy from prior to posterior distribution in the probabilistic grammar (Levy, 2008)
  - Optimal sensory discrimination: Bayesian sequential probability ratio test on “what is this word?” (Stone, 1960; Laming, 1968; Norris, 2006)

\[
D(P_i || P_{i-1}) = \sum_{I} P_i(I) \log \frac{P_{i-1}(I)}{P_i(I)}
\]
Theoretical interpretations of surprisal

• Three proposals:
  • Relative entropy from prior to posterior distribution in the probabilistic grammar (Levy, 2008)
  • Optimal sensory discrimination: Bayesian sequential probability ratio test on “what is this word?” (Stone, 1960; Laming, 1968; Norris, 2006)
  • Any kind of general probability sensitivity plus *highly incremental processing* (Smith & Levy, 2013)

\[
D(P_i \| P_{i-1}) = \sum_I P_i(I) \log \frac{P_{i-1}(I)}{P_i(I)}
\]

\[
\text{Time}(\text{together}|\text{come}) = \text{Time}(\text{to-}|\text{come}) + \text{Time}(\text{-ge-}|\text{come, to-}) + \text{Time}(\text{-ther}|\text{come, toge-})
\]
Theoretical interpretations of surprisal

• Three proposals:
  • Relative entropy from prior to posterior distribution in the probabilistic grammar (Levy, 2008)
  • Optimal sensory discrimination: Bayesian sequential probability ratio test on “what is this word?” (Stone, 1960; Laming, 1968; Norris, 2006)
  • Any kind of general probability sensitivity plus highly incremental processing (Smith & Levy, 2013)

\[
D(P_i || P_{i-1}) = \sum_{I} P_i(I) \log \frac{P_{i-1}(I)}{P_i(I)}
\]
Estimating probability/time curve shape
Estimating probability/time curve shape

- As a proxy for “processing difficulty,” reading time in two different methods: self-paced reading & eye-tracking
Estimating probability/time curve shape

- As a proxy for “processing difficulty,” reading time in two different methods: self-paced reading & eye-tracking
- Challenge: we need big data to estimate curve shape, but probability correlated with confounding variables
Estimating probability/time curve shape

- As a proxy for “processing difficulty,” reading time in two different methods: self-paced reading & eye-tracking
- Challenge: we need big data to estimate curve shape, but probability correlated with confounding variables
Estimating probability/time curve shape

- GAM regression: total contribution of word (trigram) probability to RT near-linear over 6 orders of magnitude!

(Smith & Levy, 2013)
Surprisal theory: summary

• Surprisal: a simple theory of how linguistic knowledge guides expectation formation in online processing
  • Unifies ambiguity resolution and syntactic complexity
  • Covers a range of findings in both domains
  • Accounts for anti-locality effects problematic for memory-based theories of syntactic complexity
Levels of analysis

- Most of what we’ve covered today is at Marr’s *computational* level of analysis
  - Focus on the goals of computation and solutions based on all available sources of knowledge
- Jurafsky’s beam search introduced *algorithmic* level considerations
  - Use *beam search* to keep # candidate parses manageable
- We’ll now see an appealing alternative to beam search
Memory constraints: a theoretical puzzle

Levy, Reali, & Griffiths, 2009, NIPS
Memory constraints: a theoretical puzzle

• Logically possible analyses grows at best exponentially in sentence length

Levy, Reali, & Griffiths, 2009, NIPS
Memory constraints: a theoretical puzzle

• # Logically possible analyses grows at best exponentially in sentence length
• Exact probabilistic inference with context-free grammars can be done efficiently in $O(n^3)$
Memory constraints: a theoretical puzzle

- # Logically possible analyses grows \textit{at best} exponentially in sentence length
- Exact probabilistic inference with context-free grammars can be done efficiently in $O(n^3)$
- But...
  - Requires \textit{probabilistic locality}, limiting conditioning context
  - Human parsing is linear—that is, $O(n)$—anyway

Levy, Reali, & Griffiths, 2009, NIPS
Memory constraints: a theoretical puzzle

• # Logically possible analyses grows \textit{at best} exponentially in sentence length
• Exact probabilistic inference with context-free grammars can be done efficiently in $O(n^3)$
• But…
  • Requires \textit{probabilistic locality}, limiting conditioning context
  • Human parsing is linear—that is, $O(n)$—anyway
• So we must be restricting attention to some subset of analyses

\textit{Levy, Reali, & Griffiths, 2009, NIPS}
Memory constraints: a theoretical puzzle

- # Logically possible analyses grows at best exponentially in sentence length
- Exact probabilistic inference with context-free grammars can be done efficiently in $O(n^3)$
- But…
  - Requires probabilistic locality, limiting conditioning context
  - Human parsing is linear—that is, $O(n)$—anyway
- So we must be restricting attention to some subset of analyses
- **Puzzle**: how to choose and manage this subset?
  - Previous efforts: $k$-best beam search

*Levy, Reali, & Griffiths, 2009, NIPS*
Memory constraints: a theoretical puzzle

- # Logically possible analyses grows **at best** exponentially in sentence length
- Exact probabilistic inference with context-free grammars can be done efficiently in $O(n^3)$
- But…
  - Requires *probabilistic locality*, limiting conditioning context
  - Human parsing is linear—that is, $O(n)$—anyway
- So we must be restricting attention to some subset of analyses
- **Puzzle**: how to choose and manage this subset?
  - Previous efforts: $k$-best beam search
- Here, we’ll explore the *particle filter* as a model of limited-parallel approximate inference

*Levy, Reali, & Griffiths, 2009, NIPS*
The particle filter: general picture

Levy, Reali, & Griffiths, 2009, NIPS
The particle filter: general picture

- Sequential Monte Carlo for incremental observations

Levy, Reali, & Griffiths, 2009, NIPS
The particle filter: general picture

- Sequential Monte Carlo for incremental observations
- Let $x_i$ be observed data, $z_i$ be unobserved states
  - For parsing: $x_i$ are words, $z_i$ are incremental structures

Levy, Reali, & Griffiths, 2009, NIPS
The particle filter: general picture

- Sequential Monte Carlo for incremental observations
- Let $x_i$ be observed data, $z_i$ be unobserved states
  - For parsing: $x_i$ are words, $z_i$ are incremental structures
- Suppose that after $n-1$ observations we have the distribution over interpretations $P(z_{n-1} | x_1...n-1)$

Levy, Reali, & Griffiths, 2009, NIPS
The particle filter: general picture

- Sequential Monte Carlo for incremental observations
- Let \( x_i \) be observed data, \( z_i \) be unobserved states
  - For parsing: \( x_i \) are words, \( z_i \) are incremental structures
- Suppose that after \( n-1 \) observations we have the distribution over interpretations \( P(z_{n-1}|x_1...n-1) \)
- After next observation \( x_n \), represent \( P(z_n|x_1...n) \) inductively:

\[
P(z_n|x_1...n) \propto P(x_n|z_n) \underbrace{P(z_n|z_{n-1})}_{\text{Hypothesized structure}} \underbrace{P(z_{n-1}|x_1...n-1)}_{\text{observation}}
\]
The particle filter: general picture

- Sequential Monte Carlo for incremental observations
- Let $x_i$ be observed data, $z_i$ be unobserved states
  - For parsing: $x_i$ are words, $z_i$ are incremental structures
- Suppose that after $n-1$ observations we have the distribution over interpretations $P(z_{n-1}|x_{1...n-1})$
- After next observation $x_n$, represent $P(z_n|x_{1...n})$ inductively:
  \[
P(z_n|x_{1...n}) \propto P(x_n|z_n)\underbrace{P(z_n|z_{n-1})}_{\text{Likelihood}}\underbrace{P(z_{n-1}|x_{1...n-1})}_{\text{Prior}}
  \]
  observation  Hypothesized
- Approximate $P(z_i|x_{1...i})$ by samples

Levy, Reali, & Griffiths, 2009, NIPS
The particle filter: general picture

- Sequential Monte Carlo for incremental observations
- Let $x_i$ be observed data, $z_i$ be unobserved states
  - For parsing: $x_i$ are words, $z_i$ are incremental structures
- Suppose that after $n-1$ observations we have the distribution over interpretations $P(z_{n-1} | x_1...n-1)$
- After next observation $x_n$, represent $P(z_n | x_1...n)$ inductively:
  \[
  P(z_n | x_1...n) \propto P(x_n | z_n) \cdot P(z_n | z_{n-1}) \cdot P(z_{n-1} | x_1...n-1)
  \]
  - Posterior \hspace{0.5cm} Likelihood \hspace{0.5cm} Prior
  - observation \hspace{0.5cm} Hypothesized
- Approximate $P(z_i | x_1...i)$ by samples
- Sample $z_n$ from $P(z_n | z_{n-1})$, and reweight by $P(x_n | z_n)$

Levy, Reali, & Griffiths, 2009, NIPS
Particle filter with probabilistic grammars

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>NP VP</td>
<td>1.0</td>
<td>V</td>
</tr>
<tr>
<td>NP</td>
<td>N</td>
<td>0.8</td>
<td>V</td>
</tr>
<tr>
<td>NP</td>
<td>N RRC</td>
<td>0.2</td>
<td>V</td>
</tr>
<tr>
<td>RR</td>
<td>Part N</td>
<td>1.0</td>
<td>Part</td>
</tr>
<tr>
<td>VP</td>
<td>V N</td>
<td>1.0</td>
<td>Part</td>
</tr>
<tr>
<td>N</td>
<td>women</td>
<td>0.7</td>
<td>Part</td>
</tr>
<tr>
<td>N</td>
<td>sandwich</td>
<td>0.3</td>
<td>Adv</td>
</tr>
</tbody>
</table>

women brought sandwiches
Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Rule</th>
<th>Non-terminal</th>
<th>Probability</th>
<th>Terminal</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>NP VP</td>
<td>1.0</td>
<td>brought</td>
<td>0.4</td>
</tr>
<tr>
<td>NP</td>
<td>N</td>
<td>0.8</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>NP</td>
<td>N RRC</td>
<td>0.2</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>RR</td>
<td>Part N</td>
<td>1.0</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>VP</td>
<td>V N</td>
<td>1.0</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>women</td>
<td>0.7</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>N</td>
<td>sandwich</td>
<td>0.3</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

S

NP

women brought sandwiches
### Particle filter with probabilistic grammars

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>→</td>
<td>NP</td>
<td>VP</td>
<td>1.0</td>
<td>V</td>
<td>→</td>
</tr>
<tr>
<td>NP</td>
<td>→</td>
<td>N</td>
<td>0.8</td>
<td>V</td>
<td>→</td>
<td>broke</td>
</tr>
<tr>
<td>NP</td>
<td>→</td>
<td>N</td>
<td>RRC</td>
<td>0.2</td>
<td>V</td>
<td>→</td>
</tr>
<tr>
<td>RR</td>
<td>→</td>
<td>Part</td>
<td>N</td>
<td>1.0</td>
<td>Part</td>
<td>→</td>
</tr>
<tr>
<td>VP</td>
<td>→</td>
<td>V</td>
<td>N</td>
<td>1.0</td>
<td>Part</td>
<td>→</td>
</tr>
<tr>
<td>N</td>
<td>→</td>
<td>women</td>
<td>0.7</td>
<td>Part</td>
<td>→</td>
<td>tripped</td>
</tr>
<tr>
<td>N</td>
<td>→</td>
<td>sandwich</td>
<td>0.3</td>
<td>Adv</td>
<td>→</td>
<td>quickly</td>
</tr>
</tbody>
</table>

* women brought sandwiches
Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>S</th>
<th>→</th>
<th>NP VP</th>
<th>1.0</th>
<th>V</th>
<th>→</th>
<th>brought</th>
<th>0.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>→</td>
<td>N</td>
<td>0.8</td>
<td>V</td>
<td>→</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>NP</td>
<td>→</td>
<td>N RRC</td>
<td>0.2</td>
<td>V</td>
<td>→</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>RR</td>
<td>→</td>
<td>Part N</td>
<td>1.0</td>
<td>Part</td>
<td>→</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>VP</td>
<td>→</td>
<td>V N</td>
<td>1.0</td>
<td>Part</td>
<td>→</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>→</td>
<td>women</td>
<td>0.7</td>
<td>Part</td>
<td>→</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>N</td>
<td>→</td>
<td>sandwich</td>
<td>0.3</td>
<td>Adv</td>
<td>→</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

S
   /\NP
  /   NP
 /    /
N*    N
|
|      |
women brought sandwiches
Particle filter with probabilistic grammars

```
S → NP VP  1.0  V → brought  0.4
NP → N  0.8  V → broke  0.3
NP → N RRC  0.2  V → tripped  0.3
RR → Part N  1.0  Part → brought  0.1
VP → V N  1.0  Part → broken  0.7
N → women  0.7  Part → tripped  0.2
N → sandwich  0.3  Adv → quickly  1.0

S
  NP
    NP
      N
      *
  women  brought  sandwiches
```
Particle filter with probabilistic grammars

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>V</td>
<td></td>
<td>brought</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>N</td>
<td></td>
<td>broke</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>N</td>
<td>RRC</td>
<td>tripped</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>Part</td>
<td>N</td>
<td>brought</td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>V</td>
<td>N</td>
<td>broken</td>
<td>0.7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.7</td>
<td>Part</td>
<td>women</td>
<td>tripped</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.3</td>
<td>Part</td>
<td>sandwich</td>
<td>quickly</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S → NP VP 1.0
NP → N 0.8
NP → N RRC 0.2
RR → Part N 1.0
VP → V N 1.0
N → women 0.7
N → sandwich 0.3
Adv → quickly 1.0

S
   ├── NP*
     │   └── N
     │       └── women
     │           └── brought
     │               └── sandwiches
     │                   └── 0.7
Particle filter with probabilistic grammars

S → NP VP 1.0
NP → N 0.8
NP → N RRC 0.2
RR → Part N 1.0
VP → V N 1.0
N → women 0.7
N → sandwich 0.3

V → brought 0.4
V → broke 0.3
V → tripped 0.3
Part → brought 0.1
Part → broken 0.7
Part → tripped 0.2
Adv → quickly 1.0

S
  NP
  VP
    *
  N
    women
    brought
    sandwiches
    0.7
### Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Non-terminal</th>
<th>Production</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>NP VP</td>
<td>1.0</td>
</tr>
<tr>
<td>NP</td>
<td>N</td>
<td>0.8</td>
</tr>
<tr>
<td>NP</td>
<td>N RRC</td>
<td>0.2</td>
</tr>
<tr>
<td>RR</td>
<td>Part N</td>
<td>1.0</td>
</tr>
<tr>
<td>VP</td>
<td>V N</td>
<td>1.0</td>
</tr>
<tr>
<td>N</td>
<td>women</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>sandwich</td>
<td>0.3</td>
</tr>
</tbody>
</table>

- **S → NP VP**: S → NP VP with a probability of 1.0.
- **NP → N**: NP → N with a probability of 0.8.
- **NP → N RRC**: NP → N RRC with a probability of 0.2.
- **RR → Part N**: RR → Part N with a probability of 1.0.
- **VP → V N**: VP → V N with a probability of 1.0.
- **N → women**: N → women with a probability of 0.7.
- **N → sandwich**: N → sandwich with a probability of 0.3.

The tree structure for the sentence "women brought sandwiches" is shown below:

- **S**
  - **NP**: women
  - **VP**: brought
    - **V**: brought
    - **Adv**: quickly
  - **NP**: sandwiches
Particle filter with probabilistic grammars

\[
\begin{align*}
S & \rightarrow \ NP \ VP & 1.0 & \quad V & \rightarrow & \text{brought} & 0.4 \\
NP & \rightarrow \ N & 0.8 & \quad V & \rightarrow & \text{broke} & 0.3 \\
NP & \rightarrow \ N \ RRC & 0.2 & \quad V & \rightarrow & \text{tripped} & 0.3 \\
RR & \rightarrow \ Part \ N & 1.0 & \quad \text{Part} & \rightarrow & \text{brought} & 0.1 \\
VP & \rightarrow \ V \ N & 1.0 & \quad \text{Part} & \rightarrow & \text{broken} & 0.7 \\
N & \rightarrow \ \text{women} & 0.7 & \quad \text{Part} & \rightarrow & \text{tripped} & 0.2 \\
N & \rightarrow \ \text{sandwich} & 0.3 & \quad \text{Adv} & \rightarrow & \text{quickly} & 1.0
\end{align*}
\]
Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th></th>
<th>Rule</th>
<th>Probability</th>
<th></th>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>$S \rightarrow NP \ VP$</td>
<td>1.0</td>
<td>V</td>
<td>$V \rightarrow$ brought</td>
<td>0.4</td>
</tr>
<tr>
<td>NP</td>
<td>$NP \rightarrow N$</td>
<td>0.8</td>
<td>V</td>
<td>$V \rightarrow$ broke</td>
<td>0.3</td>
</tr>
<tr>
<td>NP</td>
<td>$NP \rightarrow N \ RRC$</td>
<td>0.2</td>
<td>V</td>
<td>$V \rightarrow$ tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>RR</td>
<td>$RR \rightarrow Part \ N$</td>
<td>1.0</td>
<td>Part</td>
<td>$Part \rightarrow$ brought</td>
<td>0.1</td>
</tr>
<tr>
<td>VP</td>
<td>$VP \rightarrow V \ N$</td>
<td>1.0</td>
<td>Part</td>
<td>$Part \rightarrow$ broken</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>$N \rightarrow$ women</td>
<td>0.7</td>
<td>Part</td>
<td>$Part \rightarrow$ tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>N</td>
<td>$N \rightarrow$ sandwich</td>
<td>0.3</td>
<td>Adv</td>
<td>$Adv \rightarrow$ quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

```
  S
   /\    /
  NP    VP
   /\    /
  N    V*
     /
  women brought sandwiches
```

0.7 0.4
Particle filter with probabilistic grammars

\[
\begin{align*}
S & \rightarrow \text{NP VP} & 1.0 \\
\text{NP} & \rightarrow \text{N} & 0.8 \\
\text{NP} & \rightarrow \text{N RRC} & 0.2 \\
\text{RR} & \rightarrow \text{Part N} & 1.0 \\
\text{VP} & \rightarrow \text{V N} & 1.0 \\
\text{N} & \rightarrow \text{women} & 0.7 \\
\text{N} & \rightarrow \text{sandwich} & 0.3 \\
V & \rightarrow \text{brought} & 0.4 \\
V & \rightarrow \text{broke} & 0.3 \\
V & \rightarrow \text{tripped} & 0.3 \\
\text{Part} & \rightarrow \text{brought} & 0.1 \\
\text{Part} & \rightarrow \text{broken} & 0.7 \\
\text{Part} & \rightarrow \text{tripped} & 0.2 \\
\text{Adv} & \rightarrow \text{quickly} & 1.0
\end{align*}
\]
### Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Production</th>
<th>Probability</th>
<th>Example</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>1.0</td>
<td>women</td>
<td>0.7</td>
</tr>
<tr>
<td>NP</td>
<td>0.8</td>
<td>brought</td>
<td>0.4</td>
</tr>
<tr>
<td>NP</td>
<td>0.2</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>NP</td>
<td>0.2</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>RR</td>
<td>1.0</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>VP</td>
<td>1.0</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>0.7</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>N</td>
<td>0.3</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

#### Parse Tree

```
S
  NP   VP
    N   V
      V  N
        N *
          women  brought sandwiches
```

- **S**: sentence node with weight 1.0
- **NP**: noun phrase with weight 0.8
- **VP**: verb phrase with weight 0.4
- **V**: verb node with weight 0.3
- **N**: noun node with weight 0.7
- **Part**: particle node with weight 0.1
- **Adv**: adverb node with weight 1.0
Particle filter with probabilistic grammars

\[
\begin{align*}
S & \rightarrow \text{NP VP} & 1.0 \\
\text{NP} & \rightarrow \text{N} & 0.8 \\
\text{NP} & \rightarrow \text{N RRC} & 0.2 \\
\text{RR} & \rightarrow \text{Part N} & 1.0 \\
\text{VP} & \rightarrow \text{V N} & 1.0 \\
\text{N} & \rightarrow \text{women} & 0.7 \\
\text{N} & \rightarrow \text{sandwich} & 0.3 \\
\end{align*}
\]

\[
\begin{align*}
\text{V} & \rightarrow \text{brought} & 0.4 \\
\text{V} & \rightarrow \text{broke} & 0.3 \\
\text{V} & \rightarrow \text{tripped} & 0.3 \\
\text{Part} & \rightarrow \text{brought} & 0.1 \\
\text{Part} & \rightarrow \text{broken} & 0.7 \\
\text{Part} & \rightarrow \text{tripped} & 0.2 \\
\text{Adv} & \rightarrow \text{quickly} & 1.0 \\
\end{align*}
\]
### Particle filter with probabilistic grammars

| S → | NP VP | 1.0   | V → | brought | 0.4   |
| NP → | N    | 0.8   | V → | broke   | 0.3   |
| NP → | N RRC| 0.2   | V → | tripped | 0.3   |
| RR → | Part N | 1.0   | Part | brought | 0.1   |
| VP → | V N  | 1.0   | Part | broken  | 0.7   |
| N →  | women | 0.7   | Part | tripped | 0.2   |
| N →  | sandwich | 0.3 | Adv  | quickly | 1.0   |

Diagram:

```
S
  NP   VP
    N   V
      N *
        women brought sandwiches
```

0.7  0.4  0.3
### Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Rule</th>
<th>Non-terminal</th>
<th>Probability</th>
<th>Action</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>$NP ; VP$</td>
<td>1.0</td>
<td>$V$</td>
<td>0.4</td>
</tr>
<tr>
<td>$NP$</td>
<td>$N$</td>
<td>0.8</td>
<td>$V$</td>
<td>0.3</td>
</tr>
<tr>
<td>$NP$</td>
<td>$N ; RRC$</td>
<td>0.2</td>
<td>$V$</td>
<td>0.3</td>
</tr>
<tr>
<td>$RR$</td>
<td>$Part ; N$</td>
<td>1.0</td>
<td>$Part$</td>
<td>0.1</td>
</tr>
<tr>
<td>$VP$</td>
<td>$V ; N$</td>
<td>1.0</td>
<td>$Part$</td>
<td>0.7</td>
</tr>
<tr>
<td>$N$</td>
<td>$women$</td>
<td>0.7</td>
<td>$Part$</td>
<td>0.2</td>
</tr>
<tr>
<td>$N$</td>
<td>$sandwich$</td>
<td>0.3</td>
<td>$Adv$</td>
<td>1.0</td>
</tr>
</tbody>
</table>

```
S
   /\    \
  NP   VP
     /\     \
    N   V     N*
    /\    |      |
  women brought sandwiches
```

```
women brought sandwiches
$$0.7 \quad 0.4 \quad 0.3$$
```
Particle filter with probabilistic grammars

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>→</td>
<td>NP</td>
<td>VP</td>
<td>1.0</td>
<td>V</td>
<td>→</td>
<td>brought</td>
</tr>
<tr>
<td>NP</td>
<td>→</td>
<td>N</td>
<td>0.8</td>
<td>V</td>
<td>→</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>NP</td>
<td>→</td>
<td>N RRC</td>
<td>0.2</td>
<td>V</td>
<td>→</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>RR</td>
<td>→</td>
<td>Part</td>
<td>N</td>
<td>1.0</td>
<td>Part</td>
<td>→</td>
<td>brought</td>
</tr>
<tr>
<td>VP</td>
<td>→</td>
<td>V</td>
<td>N</td>
<td>1.0</td>
<td>Part</td>
<td>→</td>
<td>broken</td>
</tr>
<tr>
<td>N</td>
<td>→</td>
<td>women</td>
<td>0.7</td>
<td>Part</td>
<td>→</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>N</td>
<td>→</td>
<td>sandwich</td>
<td>0.3</td>
<td>Adv</td>
<td>→</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

```
S
  NP
    N
      women
      brought
      sandwiches

NP
  VP
    N
      brought

S
  NP
    *
      women
      brought
      sandwiches
```

Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Rule</th>
<th>Non-terminal</th>
<th>Probability</th>
<th>Rule</th>
<th>Non-terminal</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>→ NP VP</td>
<td>1.0</td>
<td>V</td>
<td>→ brought</td>
<td>0.4</td>
</tr>
<tr>
<td>NP</td>
<td>→ N</td>
<td>0.8</td>
<td>V</td>
<td>→ broke</td>
<td>0.3</td>
</tr>
<tr>
<td>NP</td>
<td>→ N RRC</td>
<td>0.2</td>
<td>V</td>
<td>→ tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>RR</td>
<td>→ Part N</td>
<td>1.0</td>
<td>Part</td>
<td>→ brought</td>
<td>0.1</td>
</tr>
<tr>
<td>VP</td>
<td>→ V N</td>
<td>1.0</td>
<td>Part</td>
<td>→ broken</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>→ women</td>
<td>0.7</td>
<td>Part</td>
<td>→ tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>N</td>
<td>→ sandwich</td>
<td>0.3</td>
<td>Adv</td>
<td>→ quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Graph representation:

- **S**
  - **NP**
    - **N**
    - **V**
    - **N**

- **NP**
  - **women**
  - **brought**
  - **sandwiches**

- **VP**
  - **women**
  - **brought**
  - **sandwiches**
<table>
<thead>
<tr>
<th>S</th>
<th>NP VP</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>N</td>
<td>0.8</td>
</tr>
<tr>
<td>NP</td>
<td>N RRC</td>
<td>0.2</td>
</tr>
<tr>
<td>RR</td>
<td>Part N</td>
<td>1.0</td>
</tr>
<tr>
<td>VP</td>
<td>V N</td>
<td>1.0</td>
</tr>
<tr>
<td>N</td>
<td>women</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>sandwich</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>brought</td>
<td>0.4</td>
</tr>
<tr>
<td>V</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>Part</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>Part</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>Part</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>Adv</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Diagram:**

```
S
  NP
    N
      V
        N *
          women
          brought
          sandwiches
          0.7
          0.4
          0.3

S
  NP
    N *
      N
        V
          N *
            women
            brought
            sandwiches
```
Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Production</th>
<th>Probability</th>
<th>Production</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → NP VP</td>
<td>1.0</td>
<td>V → brought</td>
<td>0.4</td>
</tr>
<tr>
<td>NP → N</td>
<td>0.8</td>
<td>V → broke</td>
<td>0.3</td>
</tr>
<tr>
<td>NP → N RRC</td>
<td>0.2</td>
<td>V → tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>RR → Part N</td>
<td>1.0</td>
<td>Part → brought</td>
<td>0.1</td>
</tr>
<tr>
<td>VP → V N</td>
<td>1.0</td>
<td>Part → broken</td>
<td>0.7</td>
</tr>
<tr>
<td>N → women</td>
<td>0.7</td>
<td>Part → tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>N → sandwich</td>
<td>0.3</td>
<td>Adv → quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

```
S
  NP
    N
      women
      brought
      sandwiches
    VP
      V
        brought
      N
        women
        brought
        sandwiches
```

```
S
  NP
    N*
      women
      brought
      sandwiches
```
Particle filter with probabilistic grammars

\[
\begin{align*}
S & \rightarrow \ NP \ VP \quad 1.0 \\
NP & \rightarrow \ N \quad 0.8 \\
NP & \rightarrow \ N \ RRC \quad 0.2 \\
RR & \rightarrow \ Part \ N \quad 1.0 \\
VP & \rightarrow \ V \ N \quad 1.0 \\
N & \rightarrow \ women \quad 0.7 \\
N & \rightarrow \ sandwich \quad 0.3 \\
S & \rightarrow \ women \ brought \ sandwiches \\
V & \rightarrow \ brought \quad 0.4 \\
V & \rightarrow \ broke \quad 0.3 \\
V & \rightarrow \ tripped \quad 0.3 \\
Part & \rightarrow \ brought \quad 0.1 \\
Part & \rightarrow \ broken \quad 0.7 \\
Part & \rightarrow \ tripped \quad 0.2 \\
Adv & \rightarrow \ quickly \quad 1.0
\end{align*}
\]
Particle filter with probabilistic grammars

S → NP VP 1.0
NP → N 0.8
NP → N RRC 0.2
RR → Part N 1.0
VP → V N 1.0
N → women 0.7
N → sandwich 0.3
V → brought 0.4
V → broke 0.3
V → tripped 0.3
Part → brought 0.1
Part → broken 0.7
Part → tripped 0.2
Adv → quickly 1.0

S
  └── NP
      └── VP
          └── N
              └── V
                  └── N
                      └── women
                          └── V
                              └── N
                                  └── brought
                                      └── sandwiches
                                          └── 0.7

NP
  └── RRC
      └── Part
          └── S
              └── N
                  └── N
                      └── women
                          └── V
                              └── Part
                                  └── brought
                                      └── sandwiches
                                          └── 0.7
Particle filter with probabilistic grammars

S $\rightarrow$ NP VP 1.0
NP $\rightarrow$ N 0.8
NP $\rightarrow$ N RRC 0.2
RR $\rightarrow$ Part N 1.0
VP $\rightarrow$ V N 1.0
N $\rightarrow$ women 0.7
N $\rightarrow$ sandwich 0.3

V $\rightarrow$ brought 0.4
V $\rightarrow$ broke 0.3
V $\rightarrow$ tripped 0.3
Part $\rightarrow$ brought 0.1
Part $\rightarrow$ broken 0.7
Part $\rightarrow$ tripped 0.2
Adv $\rightarrow$ quickly 1.0

S

NP

NP

VP

N

V

N

women

brought

sandwiches

0.7

0.4

0.3

S

NP

NP

RRC

Part *

women

brought

sandwiches

0.7
**Particle filter with probabilistic grammars**

<table>
<thead>
<tr>
<th>S</th>
<th>→ NP VP</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>→ N</td>
<td>0.8</td>
</tr>
<tr>
<td>NP</td>
<td>→ N RRC</td>
<td>0.2</td>
</tr>
<tr>
<td>RR</td>
<td>→ Part N</td>
<td>1.0</td>
</tr>
<tr>
<td>VP</td>
<td>→ V N</td>
<td>1.0</td>
</tr>
<tr>
<td>N</td>
<td>→ women</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>→ sandwich</td>
<td>0.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>V</th>
<th>→ brought</th>
<th>0.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>→ broke</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>→ tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>Part</td>
<td>→ brought</td>
<td>0.1</td>
</tr>
<tr>
<td>Part</td>
<td>→ broken</td>
<td>0.7</td>
</tr>
<tr>
<td>Part</td>
<td>→ tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>Adv</td>
<td>→ quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Example sentences:**

- `S -> NP VP 1.0` produces `women brought sandwiches 0.7 0.4 0.3`
- `S -> NP VP 1.0`

**Tree diagrams:**

- Left: `S -> NP VP 1.0` with `women brought sandwiches 0.7 0.4 0.3`
- Right: `S -> NP RRC 1.0` with `women brought sandwiches 0.7 0.1`
Particle filter with probabilistic grammars

S \rightarrow NP \ VP \ 1.0
NP \rightarrow N \ 0.8
NP \rightarrow N \ RRC \ 0.2
RR \rightarrow Part \ N \ 1.0
VP \rightarrow V \ N \ 1.0
N \rightarrow women \ 0.7
N \rightarrow sandwich \ 0.3

V \rightarrow brought \ 0.4
V \rightarrow broke \ 0.3
V \rightarrow tripped \ 0.3
Part \rightarrow brought \ 0.1
Part \rightarrow broken \ 0.7
Part \rightarrow tripped \ 0.2
Adv \rightarrow quickly \ 1.0
Particle filter with probabilistic grammars

S → NP VP 1.0 V → brought 0.4
NP → N 0.8 V → broke 0.3
NP → N RRC 0.2 V → tripped 0.3
RR → Part N 1.0 Part → brought 0.1
VP → V N 1.0 Part → broken 0.7
N → women 0.7 Part → tripped 0.2
N → sandwich 0.3 Adv → quickly 1.0
Particle filter with probabilistic grammars

S → NP VP 1.0
NP → N 0.8
NP → N RRC 0.2
RR → Part N 1.0
VP → V N 1.0
N → women 0.7
N → sandwich 0.3
V → brought 0.4
V → broke 0.3
V → tripped 0.3
Part → brought 0.1
Part → broken 0.7
Part → tripped 0.2
Adv → quickly 1.0
Particle filter with probabilistic grammars

S → NP VP 1.0
NP → N 0.8
NP → N RRC 0.2
RR → Part N 1.0
VP → V N 1.0
N → women 0.7
N → sandwich 0.3
V → brought 0.4
V → broke 0.3
V → tripped 0.3
Part → brought 0.1
Part → broken 0.7
Part → tripped 0.2
Adv → quickly 1.0

Tree 1:
S
  / 
NP  VP
  / 
N  V
  / 
  women  brought
  /   /  
sandwiches tripped

Tree 2:
S
  / 
NP
  / 
N  RRC
  / 
  Part
  / 
  women
  / 
sandwiches

0.7 0.4 0.3
0.7 0.1 0.3
### Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>S</th>
<th>NP VP</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>N</td>
<td>0.8</td>
</tr>
<tr>
<td>NP</td>
<td>N RRC</td>
<td>0.2</td>
</tr>
<tr>
<td>RR</td>
<td>Part N</td>
<td>1.0</td>
</tr>
<tr>
<td>VP</td>
<td>V N</td>
<td>1.0</td>
</tr>
<tr>
<td>N</td>
<td>women</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>sandwich</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>brought</td>
<td>0.4</td>
</tr>
<tr>
<td>V</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>Part</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>Part</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>Part</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>Adv</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Diagram:**

```
S
  /\   /
 NP   VP
 /\   /
 N   V  N
 \   /  \
 women brought sandwiches tripped
```

```
S
  /\   /
 NP   RRC
 /\  /\  /
 N  Part N
 \  /  /  \
 women brought sandwiches
```

```
S
  /\   /
 NP   VP
 /\   /
 N   V  N
 \   /  \
 women brought sandwiches tripped
```

```
S
  /\   /
 NP   RRC
 /\  /\  /
 N  Part N
 \  /  /  \
 women brought sandwiches
```
Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>NP VP</td>
<td>1.0</td>
</tr>
<tr>
<td>NP</td>
<td>N</td>
<td>0.8</td>
</tr>
<tr>
<td>NP</td>
<td>N RRC</td>
<td>0.2</td>
</tr>
<tr>
<td>RR</td>
<td>Part N</td>
<td>1.0</td>
</tr>
<tr>
<td>VP</td>
<td>V N</td>
<td>1.0</td>
</tr>
<tr>
<td>N</td>
<td>women</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>sandwich</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>brought</td>
<td>0.4</td>
</tr>
<tr>
<td>V</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>Part</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>Part</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>Part</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>Adv</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Particle filter with probabilistic grammars

\[
\begin{align*}
S & \rightarrow \text{NP VP} \quad 1.0 \\
\text{NP} & \rightarrow \text{N} \quad 0.8 \\
\text{NP} & \rightarrow \text{N RRC} \quad 0.2 \\
\text{RR} & \rightarrow \text{Part N} \quad 1.0 \\
\text{VP} & \rightarrow \text{V N} \quad 1.0 \\
\text{N} & \rightarrow \text{women} \quad 0.7 \\
\text{N} & \rightarrow \text{sandwich} \quad 0.3 \\
\text{V} & \rightarrow \text{brought} \quad 0.4 \\
\text{V} & \rightarrow \text{broke} \quad 0.3 \\
\text{V} & \rightarrow \text{tripped} \quad 0.3 \\
\text{Part} & \rightarrow \text{brought} \quad 0.1 \\
\text{Part} & \rightarrow \text{broken} \quad 0.7 \\
\text{Part} & \rightarrow \text{tripped} \quad 0.2 \\
\text{Adv} & \rightarrow \text{quickly} \quad 1.0 
\end{align*}
\]
Particle filter with probabilistic grammars

S → NP VP 1.0  V → brought 0.4
NP → N 0.8  V → broke 0.3
NP → N RRC 0.2  V → tripped 0.3
RR → Part N 1.0  Part → brought 0.1
VP → V N 1.0  Part → broken 0.7
N → women 0.7  Part → tripped 0.2
N → sandwich 0.3  Adv → quickly 1.0

S
    /\ NP
   /   /
  VP   N
  /\   /\ V   N
 women brought sandwiches tripped

S
   /\ NP
  /   /
 N   RRC
    /   /
 Part   N
   /\   /\ *
 women brought sandwiches
### Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th></th>
<th>S → NP VP</th>
<th>V → brought</th>
<th>1.0</th>
<th>V → broke</th>
<th>0.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP → N</td>
<td>V → broke</td>
<td>0.8</td>
<td>V → tripped</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>NP → N RRC</td>
<td>V → tripped</td>
<td>0.2</td>
<td>V → tripped</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>RR → Part N</td>
<td>Part → brought</td>
<td>1.0</td>
<td>Part → brought</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>VP → V N</td>
<td>Part → broken</td>
<td>1.0</td>
<td>Part → tripped</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>N → women</td>
<td>Part → tripped</td>
<td>0.7</td>
<td>Part → tripped</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>N → sandwich</td>
<td>Adv → quickly</td>
<td>0.3</td>
<td>Adv → quickly</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>

![Parsing tree](image1)

![Parsing tree](image2)
### Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
<th>Action</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → NP VP</td>
<td>1.0</td>
<td>brought</td>
<td>0.4</td>
</tr>
<tr>
<td>NP → N</td>
<td>0.8</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>NP → N RRC</td>
<td>0.2</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>RR → Part N</td>
<td>1.0</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>VP → V N</td>
<td>1.0</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>N → women</td>
<td>0.7</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>N → sandwich</td>
<td>0.3</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

#### Examples

**Example 1:**
- S → NP VP
- NP → N
- N → women
- V → brought

**Example 2:**
- S → NP VP
- NP → N RRC
- N → sandwich
- V → tripped
## Particle filter with probabilistic grammars

<table>
<thead>
<tr>
<th>Non-terminal</th>
<th>Production Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>NP VP</td>
<td>1.0</td>
</tr>
<tr>
<td>NP</td>
<td>N</td>
<td>0.8</td>
</tr>
<tr>
<td>NP</td>
<td>N RRC</td>
<td>0.2</td>
</tr>
<tr>
<td>RR</td>
<td>Part N</td>
<td>1.0</td>
</tr>
<tr>
<td>VP</td>
<td>V N</td>
<td>1.0</td>
</tr>
<tr>
<td>N</td>
<td>women</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>sandwich</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>brought</td>
<td>0.4</td>
</tr>
<tr>
<td>V</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>Part</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>Part</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>Part</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>Adv</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Particle filter with probabilistic grammars

S → NP VP  1.0
NP → N    0.8
NP → N RRC 0.2
RR → Part N 1.0
VP → V N  1.0
N → women 0.7
N → sandwich 0.3

V → brought 0.4
V → broke 0.3
V → tripped 0.3
Part → brought 0.1
Part → broken 0.7
Part → tripped 0.2
Adv → quickly 1.0

S
  └── NP
      ├── V
      │   └── N
      │       └── RRC
      │           └── Part
      │               └── N
      └── VP
          └── NP
              └── V
                  ├── N
                  │   └── Adv
                  └── N
                      └── RRC
                          └── Part
                              └── N

women brought sandwiches tripped
0.7 0.4 0.3

women brought sandwiches tripped
0.7 0.1 0.3
*Particle filter with probabilistic grammars*

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Grammar Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>NP VP</td>
<td>1.0</td>
</tr>
<tr>
<td>NP</td>
<td>N</td>
<td>0.8</td>
</tr>
<tr>
<td>NP</td>
<td>N RRC</td>
<td>0.2</td>
</tr>
<tr>
<td>RR</td>
<td>Part N</td>
<td>1.0</td>
</tr>
<tr>
<td>VP</td>
<td>V N</td>
<td>1.0</td>
</tr>
<tr>
<td>N</td>
<td>women</td>
<td>0.7</td>
</tr>
<tr>
<td>N</td>
<td>sandwich</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>brought</td>
<td>0.4</td>
</tr>
<tr>
<td>V</td>
<td>broke</td>
<td>0.3</td>
</tr>
<tr>
<td>V</td>
<td>tripped</td>
<td>0.3</td>
</tr>
<tr>
<td>Part</td>
<td>brought</td>
<td>0.1</td>
</tr>
<tr>
<td>Part</td>
<td>broken</td>
<td>0.7</td>
</tr>
<tr>
<td>Part</td>
<td>tripped</td>
<td>0.2</td>
</tr>
<tr>
<td>Adv</td>
<td>quickly</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Diagram:**

```
  S
 /\   \
NP  VP
  /\   \
 N  V N
   /   \
 women brought sandwiches tripped
```

```
  S
 /\   \
NP  VP
  /\   \
 N  V N
   /   \
 women brought sandwiches tripped
```

```
  S
 /\   \
NP  RRC
  /\   \
 Part  N
   /   \
 women  tripped
```

```
  S
 /\   \
NP  VP
  /\   \
 Part  N
   /   \
 women  tripped
```

```
  S
 /\   \
NP  VP
  /\   \
 N  V N
   /   \
 women  quicksandwiches  tripped
```

```
  S
 /\   \
NP  VP
  /\   \
 Part  N
   /   \
 women  quicksandwiches
```

```
  S
 /\   \
NP  VP
  /\   \
 N  V N
   /   \
 women  quicksandwiches  tripped
```

```
  S
 /\   \
NP  VP
  /\   \
 Part  N
   /   \
 women  broken
```

```
  S
 /\   \
NP  VP
  /\   \
 Part  N
   /   \
 women  broken
```

```
  S
 /\   \
NP  VP
  /\   \
 N  V N
   /   \
 women  quicksandwiches  tripped
```

```
  S
 /\   \
NP  VP
  /\   \
 Part  N
   /   \
 women  broken
```
Particle filter with probabilistic grammars

\[ S \rightarrow NP \ VP \ 1.0 \quad V \rightarrow \ \text{brought} \quad 0.4 \]
\[ NP \rightarrow N \ 0.8 \quad V \rightarrow \ \text{broke} \quad 0.3 \]
\[ NP \rightarrow N RRC \ 0.2 \quad V \rightarrow \ \text{tripped} \quad 0.3 \]
\[ RR \rightarrow \ \text{Part} \ N \ 1.0 \quad \text{Part} \rightarrow \ \text{brought} \quad 0.1 \]
\[ VP \rightarrow V \ N \ 1.0 \quad \text{Part} \rightarrow \ \text{broken} \quad 0.7 \]
\[ N \rightarrow \ \text{women} \ 0.7 \quad \text{Part} \rightarrow \ \text{tripped} \quad 0.2 \]
\[ N \rightarrow \ \text{sandwiches} \ 0.3 \quad \text{Adv} \rightarrow \ \text{quickly} \quad 1.0 \]
Particle filter with probabilistic grammars

S → NP VP 1.0
NP → N 0.8
NP → N RRC 0.2
RR → Part N 1.0
VP → V N 1.0
N → women 0.7
N → sandwich 0.3

V → brought 0.4
V → broke 0.3
V → tripped 0.3
Part → brought 0.1
Part → broken 0.7
Part → tripped 0.2
Adv → quickly 1.0

S
 NP VP
  N V N
   women brought sandwiches tripped

0.7 0.4 0.3

S
 NP
  N RRC
   Part N
    women brought sandwiches tripped

0.7 0.1 0.3 0.3
Simple garden-path sentences
Simple garden-path sentences

*The woman brought the sandwich from the kitchen tripped*
Simple garden-path sentences

*The woman brought the sandwich from the kitchen tripped*

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

_The woman brought the sandwich from the kitchen tripped_

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

- Comprehender is initially misled away from ultimately correct interpretation
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)

- Comprehender is initially misled away from ultimately correct interpretation
- With finitely many hypotheses, recovery is not always successful
Resampling in the particle filter

- With the naïve particle filter, inferences are highly dependent on initial choices
  - Most particles wind up with small weights
  - Region of dense posterior poorly explored
- Especially bad for parsing
  - Space of possible parses grows (at best) exponentially with input length

*input*
Resampling in the particle filter

- With the naïve particle filter, inferences are highly dependent on initial choices
  - Most particles wind up with small weights
  - Region of dense posterior poorly explored
- Especially bad for parsing
  - Space of possible parses grows (at best) exponentially with input length
Resampling in the particle filter

- With the naïve particle filter, inferences are highly dependent on initial choices
  - Most particles wind up with small weights
  - Region of dense posterior poorly explored
- Especially bad for parsing
  - Space of possible parses grows (at best) exponentially with input length
Resampling in the particle filter

• With the naïve particle filter, inferences are highly dependent on initial choices
  • Most particles wind up with small weights
  • Region of dense posterior poorly explored
• Especially bad for parsing
  • Space of possible parses grows (at best) exponentially with input length
Resampling in the particle filter

- With the naïve particle filter, inferences are highly dependent on initial choices
  - Most particles wind up with small weights
  - Region of dense posterior poorly explored
- Especially bad for parsing
  - Space of possible parses grows (at best) exponentially with input length
Resampling in the particle filter

• With the naïve particle filter, inferences are highly dependent on initial choices
  • Most particles wind up with small weights
  • Region of dense posterior poorly explored
• Especially bad for parsing
  • Space of possible parses grows (at best) exponentially with input length
• We handle this by resampling at each input word
Resampling in the particle filter

- With the naïve particle filter, inferences are highly dependent on initial choices
  - Most particles wind up with small weights
  - Region of dense posterior poorly explored
- Especially bad for parsing
  - Space of possible parses grows (at best) exponentially with input length
- We handle this by resampling at each input word
Resampling in the particle filter

- With the naïve particle filter, inferences are highly dependent on initial choices
  - Most particles wind up with small weights
  - Region of dense posterior poorly explored
- Especially bad for parsing
  - Space of possible parses grows (at best) exponentially with input length
- We handle this by resampling at each input word
Resampling in the particle filter

- With the naïve particle filter, inferences are highly dependent on initial choices
  - Most particles wind up with small weights
  - Region of dense posterior poorly explored
- Especially bad for parsing
  - Space of possible parses grows (at best) exponentially with input length
- We handle this by resampling at each input word
Resampling in the particle filter

• With the naïve particle filter, inferences are highly dependent on initial choices
  • Most particles wind up with small weights
  • Region of dense posterior poorly explored
• Especially bad for parsing
  • Space of possible parses grows (at best) exponentially with input length
• We handle this by resampling at each input word
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

**MAIN VERB** (it was the woman who brought the sandwich)

**REduced Relative** (the woman was brought the sandwich)

- Posterior initially misled away from ultimately correct interpretation
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

**MAIN VERB** (it was the woman who brought the sandwich)

**REDUCED RELATIVE** (the woman was brought the sandwich)

- Posterior initially misled away from ultimately correct interpretation
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)

- Posterior initially misled away from ultimately correct interpretation
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)

- Posterior initially misled away from ultimately correct interpretation
Simple garden-path sentences

The woman brought the sandwich from the kitchen tripped

MAIN VERB (it was the woman who brought the sandwich)

REDUCED RELATIVE (the woman was brought the sandwich)

- Posterior initially misled away from ultimately correct interpretation
- With finite # of particles, recovery is not always successful
Returning to the puzzle ("NP/S")

A-S  Tom heard the gossip wasn’t true.
A-L  Tom heard the gossip about the neighbors wasn’t true.
U-S  Tom heard that the gossip wasn’t true.
U-L  Tom heard that the gossip about the neighbors wasn’t true.

Frazier & Rayner, 1982; Tabor & Hutchins, 2004
Returning to the puzzle (“NP/S”)

A-S  Tom heard the gossip wasn’t true.
A-L  Tom heard the gossip about the neighbors wasn’t true.
U-S  Tom heard that the gossip wasn’t true.
U-L  Tom heard that the gossip about the neighbors wasn’t true.

• Previous empirical finding: ambiguity induces difficulty…

Frazier & Rayner, 1982; Tabor & Hutchins, 2004
Returning to the puzzle ("NP/S")

A-S  Tom heard the gossip wasn’t true.
A-L  Tom heard the gossip about the neighbors wasn’t true.
U-S  Tom heard that the gossip wasn’t true.
U-L  Tom heard that the gossip about the neighbors wasn’t true.

• Previous empirical finding: ambiguity induces difficulty…

Frazier & Rayner, 1982; Tabor & Hutchins, 2004
Returning to the puzzle ("NP/S")

A-S  Tom heard the gossip wasn’t true.
A-L  Tom heard the gossip about the neighbors wasn’t true.
U-S  Tom heard that the gossip wasn’t true.
U-L  Tom heard that the gossip about the neighbors wasn’t true.

• Previous empirical finding: ambiguity induces difficulty…
• …but so does the length of the ambiguous region

Frazier & Rayner, 1982; Tabor & Hutchins, 2004
Returning to the puzzle (“NP/S”)

A-S  Tom heard the gossip wasn’t true.
A-L  Tom heard the gossip about the neighbors wasn’t true.
U-S  Tom heard that the gossip wasn’t true.
U-L  Tom heard that the gossip about the neighbors wasn’t true.

• Previous empirical finding: ambiguity induces difficulty...
• …but so does the length of the ambiguous region

Frazier & Rayner, 1982; Tabor & Hutchins, 2004
Returning to the puzzle ("NP/S")

A-S  Tom heard the gossip wasn’t true.
A-L  Tom heard the gossip about the neighbors wasn’t true.
U-S  Tom heard that the gossip wasn’t true.
U-L  Tom heard that the gossip about the neighbors wasn’t true.

• Previous empirical finding: ambiguity induces difficulty…
• …but so does the length of the ambiguous region

• Our linking hypothesis:

  Proportion of parse failures at the disambiguating region should increase with sentence difficulty

Frazier & Rayner, 1982; Tabor & Hutchins, 2004
Returning to the puzzle ("NP/S")

A-S  Tom heard the gossip wasn’t true.
A-L  Tom heard the gossip about the neighbors wasn’t true.
U-S  Tom heard that the gossip wasn’t true.
U-L  Tom heard that the gossip about the neighbors wasn’t true.

• Previous empirical finding: ambiguity induces difficulty …
• …but so does the length of the ambiguous region

• Our linking hypothesis:

Proportion of parse failures at the disambiguating region should increase with sentence difficulty

Frazier & Rayner, 1982; Tabor & Hutchins, 2004
Another example (Tabor & Hutchins 2004)

Trans/Short  As the author wrote the essay the book grew.
Intr/Short   As the author wrote the book grew.
Trans/Long   As the author wrote the essay the book describing Babylon grew.
Intr/Long    As the author wrote the book describing Babylon grew.

“NP/Z”
Another example (Tabor & Hutchins 2004)

Trans/Short  As the author wrote the essay the book grew.
Intr/Short  As the author wrote the book grew.
Trans/Long  As the author wrote the essay the book describing Babylon grew.
Intr/Long  As the author wrote the book describing Babylon grew.

“NP/Z”
Resampling-induced drift
Resampling-induced drift

- In ambiguous region, observed words aren’t strongly informative ($P(x_i|z_i)$ similar across different $z_i$)
Resampling-induced drift

• In ambiguous region, observed words aren’t strongly informative ($P(x_i|z_i)$ similar across different $z_i$)

• But due to resampling, $P(z_i|x_i)$ will drift
Resampling-induced drift

- In ambiguous region, observed words aren’t strongly informative ($P(x_i|z_i)$ similar across different $z_i$)
- But due to resampling, $P(z_i|x_i)$ will drift
- One of the interpretations may be lost
Resampling-induced drift

- In ambiguous region, observed words aren’t strongly informative ($P(x_i|z_i)$ similar across different $z_i$)
- But due to resampling, $P(z_i|x_i)$ will drift
- One of the interpretations may be lost
Resampling-induced drift

• In ambiguous region, observed words aren’t strongly informative (\( P(x_i|z_i) \) similar across different \( z_i \))

• But due to resampling, \( P(z_i|x_i) \) will drift

• One of the interpretations may be lost
Resampling-induced drift

- In ambiguous region, observed words aren’t strongly informative \( P(x_i|z_i) \) similar across different \( z_i \)
- But due to resampling, \( P(z_i|x_i) \) will drift
- One of the interpretations may be lost
- The longer the ambiguous region, the more likely this is
Model results

Here are the graphs showing the proportion of parse successes at disambiguator as a function of the number of particles for different models: U-S, A-S, U-L, and A-L. The graphs are labeled NP/S and NP/Z, respectively.
Summary for today

• Probabilistic syntactic models and surprisal give a broad account of
  • Offline disambiguation
  • Online effects
    • Garden-pathing
    • Syntactic prediction in the absence of ambiguity
• Broadly supported by experimental and “corpus” behavioral data
• Adding algorithmic nuance broadens their range of coverage