Modeling the effects of memory on human online sentence processing with particle filters

Matthias Zöhrer

Institut für Grundlagen der Informationsverarbeitung
TU Graz

16. Juni 2010
Content

1. sentence processing
2. garden path sentences / diggin-in phenomenon
3. particle filters
4. algorithm results
5. conclusion
what is sentence processing?

- sentences in natural language can be interpreted in more than one way (*context*)
- interpretation of a sentence $w$ can be associated with a structural description $T$
- **parse** a sentence to reveal its underlying structure
- this means inferring $T$ from $w$
structural descriptions

- context free grammars (CFGs)

```
S → NP VP
S → Aux NP VP
S → VP
NP → Det Nominal
Nominal → Noun
Nominal → Noun Nominal
NP → Proper-Noun
VP → Verb
VP → Verb NP
Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | Prefer
Aux → does
Prep → from | to | on
Proper-Noun → Houston | TWA
Nominal → Nominal PP
```

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
How do we get the underlying structure?

▷ simply parse it: top-down

1. **start** at root node
2. **build** $\eta$ trees (apply all possible rules)
3. **stop** when the correct one is found

**sentence:** “*Book that flight*”

**NOTE:** huge search-space!

*S* → NP VP
S → Aux NP VP
S → VP
NP → Det Nominal
Nominal → Noun
Nominal → Noun Nominal
NP → Proper-Noun
VP → Verb
VP → Verb NP

---

Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | Prefer
Aux → does
Prep → from | to | on
Proper-Noun → Houston | TWA
Nominal → Nominal PP

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
How do we get the underlying structure?

▷ simply parse it: **top-down**

1. **start** at root node
2. **build** \( \eta \) trees (apply all possible rules)
3. **stop** when the correct one is found

\[
S_3 \\
\mid \\
VP
\]

\[
\begin{align*}
S & \rightarrow \text{NP } \text{VP} \\
S & \rightarrow \text{Aux } \text{NP } \text{VP} \\
S & \rightarrow \text{VP} \\
\text{NP} & \rightarrow \text{Det Nominal} \\
\text{Nominal} & \rightarrow \text{Noun} \\
\text{Nominal} & \rightarrow \text{Noun Nominal} \\
\text{NP} & \rightarrow \text{Proper-Noun} \\
\text{VP} & \rightarrow \text{Verb} \\
\text{VP} & \rightarrow \text{Verb NP}
\end{align*}
\]

**sentence:** “Book that flight”

**NOTE:** huge search-space!

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
How do we get the underlying structure?

▷ simply parse it: **top-down**

1. **start** at root node
2. **build** η trees (apply all possible rules)
3. **stop** when the correct one is found

$$S \rightarrow \text{NP VP}$$
$$S \rightarrow \text{Aux NP VP}$$
$$S \rightarrow \text{VP}$$
$$\text{NP} \rightarrow \text{Det Nominal}$$
$$\text{Nominal} \rightarrow \text{Noun}$$
$$\text{Nominal} \rightarrow \text{Noun Nominal}$$
$$\text{NP} \rightarrow \text{Proper-Noun}$$
$$\text{VP} \rightarrow \text{Verb}$$
$$\text{VP} \rightarrow \text{Verb NP}$$

<table>
<thead>
<tr>
<th>Det</th>
<th>that</th>
<th>this</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noun</td>
<td>book</td>
<td>flight</td>
<td>meal</td>
</tr>
<tr>
<td>Verb</td>
<td>book</td>
<td>include</td>
<td>Prefer</td>
</tr>
<tr>
<td>Aux</td>
<td>does</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prep</td>
<td>from</td>
<td>to</td>
<td>on</td>
</tr>
<tr>
<td>Proper-Noun</td>
<td>Houston</td>
<td>TWA</td>
<td></td>
</tr>
<tr>
<td>Nominal</td>
<td>Nominal PP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**sentence:** “Book that flight”

**NOTE:** huge search-space!

*Speech and language processing, D. Jurafsky, J. Martin, Springer,p.380*
How do we get the underlying structure?

▷ simply parse it: **top-down**

1. **start** at root node
2. **build** η trees (apply all possible rules)
3. **stop** when the correct one is found

---

```
S → NP VP
S → Aux NP VP
S → VP
NP → Det Nominal
Nominal → Noun
Nominal → Noun Nominal
NP → Proper-Noun
VP → Verb
VP → Verb NP
```

```
Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | Prefer
Aux → does

Prep → from | to | on
Proper-Noun → Houston | TWA
Nominal → Nominal PP
```

**sentence:** “Book that flight”

**NOTE:** huge search-space!

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
How do we get the underlying structure?

▷ simply parse it: **top-down**

1. **start** at root node
2. **build** \( \eta \) trees (apply all possible rules)
3. **stop** when the correct one is found

\[
S \rightarrow NP \ VP \\
S \rightarrow Aux \ NP \ VP \\
S \rightarrow VP \\
NP \rightarrow Det \ Nominal \\
\quad Nominal \rightarrow Noun \\
\quad Nominal \rightarrow Noun \ Nominal \\
NP \rightarrow Proper-Noun \\
VP \rightarrow Verb \\
VP \rightarrow Verb \ NP \\
Det \rightarrow that | this | a \\
Noun \rightarrow book | flight | meal | money \\
Verb \rightarrow book | include | Prefer \\
Aux \rightarrow does \\
Prep \rightarrow from | to | on \\
Proper-Noun \rightarrow Houston | TWA \\
Nominal \rightarrow Nominal \ PP
\]

**sentence:** “Book that flight”

**NOTE:** huge search-space!

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
How do we get the underlying structure?

▷ simply parse it: **top-down**

1. **start** at root node
2. **build** η trees (apply all possible rules)
3. **stop** when the correct one is found


(sentence: “Book that flight”)

**NOTE**: huge search-space!

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
How do we get the underlying structure?

▷ simply parse it: **top-down**

1. **start** at root node
2. **build** η trees (apply all possible rules)
3. **stop** when the correct one is found

\[
S_3 \\
| \\
VP
\]

- \[S \rightarrow NP \; VP\]
- \[S \rightarrow Aux \; NP \; VP\]
- \[S \rightarrow VP\]
- \[NP \rightarrow Det \; Nominal\]
- **Nominal** → **Noun**
  - \[Nominal \rightarrow Noun \; Nominal\]
  - \[NP \rightarrow Proper-Noun\]
- \[VP \rightarrow Verb\]
- \[VP \rightarrow Verb \; NP\]
- \[Det \rightarrow that \; | \; this \; | \; a\]
- \[Noun \rightarrow book \; | \; flight \; | \; meal \; | \; money\]
- \[Verb \rightarrow book \; | \; include \; | \; Prefer\]
- \[Aux \rightarrow does\]
- \[Prep \rightarrow from \; | \; to \; | \; on\]
- \[Proper-Noun \rightarrow Houston \; | \; TWA\]
- \[Nominal \rightarrow Nominal \; PP\]

**sentence:** “Book that flight”

**NOTE:** huge search-space!

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
How do we get the underlying structure?

▷ simply parse it: top-down

1. **start** at root node
2. **build** $\eta$ trees (apply all possible rules)
3. **stop** when the correct one is found

```plaintext
S \rightarrow NP \ VP \\
S \rightarrow Aux \ NP \ VP \\
S \rightarrow VP \\
NP \rightarrow Det \ Nominal \\
Nominal \rightarrow Noun \\
Nominal \rightarrow Noun \ Nominal \\
NP \rightarrow Proper-Noun \\
VP \rightarrow Verb \\
VP \rightarrow Verb \ NP \\
Det \rightarrow that | this | a \\
Noun \rightarrow book | flight | meal | money \\
Verb \rightarrow book | include | Prefer \\
Aux \rightarrow does \\
Prep \rightarrow from | to | on \\
Proper-Noun \rightarrow Houston | TWA \\
Nominal \rightarrow Nominal \ PP \\
```

**sentence:** “Book that flight”

**NOTE:** huge search-space!

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
How do we get the underlying structure?

▷ simply parse it: **bottom-up**

1. **start** at left node
2. **build** \( \eta \) trees (apply all possible rules)
3. **stop** when the correct one is found

1\(_{st}\) iteration:

- **Noun** that flight
- | | |
- Book Det Noun

2\(_{nd}\) iteration:

- **NOM** that **NOM**
- | | |
- Noun Det flight
- | |
- Book Noun

\[
\begin{align*}
S & \rightarrow NP \ VP \\
S & \rightarrow Aux \ NP \ VP \\
S & \rightarrow VP \\
NP & \rightarrow Det \ Nominal \\
Nominal & \rightarrow Noun \\
Nominal & \rightarrow Noun \ Nominal \\
NP & \rightarrow Proper-Noun \\
VP & \rightarrow Verb \\
VP & \rightarrow Verb \ NP
\end{align*}
\]

\[
\begin{align*}
Det & \rightarrow that \mid this \mid a \\
Noun & \rightarrow book \mid flight \mid meal \mid money \\
Verb & \rightarrow book \mid include \mid Prefer \\
Aux & \rightarrow does \\
Prep & \rightarrow from \mid to \mid on \\
Proper-Noun & \rightarrow Houston \mid TWA \\
Nominal & \rightarrow Nominal PP
\end{align*}
\]

**sentence:** “Book that flight”

**NOTE:** huge search-space!

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.380*
structural descriptions

- probabilistic context free grammars (PCFGs)

```
S → Aux NP VP  .15
NP → Pro       .40
VP → V NP NP    .05
NP → Nom       .05
NP → PNoun     .35
Nom → Noun     .75
Aux → Can      .40
NP → Pro       .40
Pro → you      .40
Verb → book    .30
PNoun → TWA    .40
Noun → flights .50
```

**sentence**: “Can you book TWA flights”

**NOTE**: search-space still huge! (pruning)  

*Speech and language processing, D. Jurafsky, J. Martin, Springer, p.447*
developed parsing methods

- CFGs
  - early algorithm (*top down parsing method*) [Early, 1970]
  - Cocke-Younger-Kasami algorithm (CYK) (*bottom up parsing method*) [Cocke, Younger, Kasami, 1960]
  - k-best algorithm [Huang, Chiang, 2005]

- PCFGs
  - probabilistic earley algorithm [Stolcke, 1995]
  - probabilistic CYK [Ney, 1991]
  - k-best algorithm [Huang, Chiang, 2005]
known problems ...

- **infinite loops**: left recursion

  \[
  S \Rightarrow S \\
  NP \quad VP \\
  S \Rightarrow S \\
  NP \quad VP \\
  NP \quad PP \quad VP \\
  S \Rightarrow ... \\
  S \Rightarrow ... \\
  NP \quad VP \\
  NP \quad PP \\
  NP \quad PP
  \]

- **ambiguity**: one sentence $\Rightarrow$ two parse trees

- **huge search space**: inefficient reparsing of subtrees $[O(n^3)]$
does the human brain work like this?

- **algorithms:**
  - run time of all shown algorithms is **super linear** in sentence length
  - strict **locality conditions** on the probabilistic dependence between events

- **humans:**
  - human processing time is **linear** in sentence length
  - seem to make use of **arbitrary features** (extra-linguistic context)
    - *(incremental posterior expectations)*
  - **some mechanisms** are **hard to explain** purely in probabilistic terms
    - *(garden-path-sentences, digging in effects)*
QUIZ - what does this mean?

"The"
QUIZ - *what does this mean?*

"The woman"
QUIZ - *what does this mean?*

"The woman *brought*“
QUIZ - *what does this mean?*

"The woman brought the"
QUIZ - what does this mean?

"The woman brought the sandwich"
QUIZ - *what does this mean?*

"The woman brought the sandwich from"
QUIZ - *what does this mean?*

"The woman brought the sandwich from the"
QUIZ - *what does this mean?*

"The woman brought the sandwich from the *kitchen*"
QUIZ - what does this mean?

"The woman brought the sandwich from kitchen tripped"
QUIZ - *what does this mean?*

"The woman brought the sandwich from kitchen *tripped*"

... that’s a *garden-path sentence*

▷ dynamic programming algorithms *recover* the correct interpretation in a single pass

▷ many readers will *fail* completely on their *first attempt*
Diggin-in *effect*

"While the man hunted *the deer* that was brown and graceful ran into the woods“

*the deer* initially seems to be the object of the preceding verb (*high probability*)

*the deer* could be the *noun* of the *main clause* as well (*low probability*)
"While the man hunted the deer that was brown and graceful ran into the woods"".

- the deer initially seems to be the object of the preceding verb (high probability)
- the deer could be the Noun as well (low probability)
Diggin-in effect

"While the man hunted the deer that was brown and graceful ran into the woods"

the deer initially seems to be the object of the preceding verb (high probability)
the deer could be the Noun as well (low probability)
Diggin-in effect

"While the man hunted the deer that was brown and graceful ran into the woods"

the deer initially seems to be the object of the preceding verb (high probability)
the deer could be the Noun as well (low probability)
"While the man hunted the deer that was brown and graceful ran into the woods"

the deer initially seems to be the object of the preceding verb (high probability) (70%)
the deer could be the Noun as well (low probability) (30%)
"While the man hunted the deer that was brown and graceful ran into the woods“

the deer initially seems to be the object of the preceding verb (high probability) (80%)
the deer could be the Noun as well (low probability) (10%)
Diggin-in effect

"While the man hunted the deer that was brown and graceful ran into the woods“

the deer initially seems to be the object of the preceding verb (high probability (20%
the deer could be the Noun as well (low probability) (80%)
Diggin-in *effect*

"While the man hunted, the deer that was brown and graceful ran *into the woods*“

*the deer* initially seems to be the object of the preceding verb (*high probability*) (10%)

*the deer* could be the Noun as well (*low probability*) (90%)
**Diggin-in effect**

"While the man hunted, the deer that was brown and graceful ran into the woods“

-the deer initially seems to be the object of the preceeding verb (**high probability**)  
-the deer could be the Noun as well (**low probability**)  

▷ **increasing** the duration of the local syntactic ambiguity **increases** the difficulty to recover the context  
▷ there seems to be an **internal feedback** mechanism  
▷ preferred **alternative** in a syntactic ambiguity seems to **grow more attractive over** the time, even if there is a strong disambiguating information  
▷ **hard** to **explain** with **local dependencies**
Particle *filters*

- generate particles
- compute weights
- normalize
- resample
Particle filters

- generate particles
- compute weights
- normalize
- resample
Particle filters

- generate particles
- compute weights
- normalize
- resample
Particle filters

- generate particles
- compute weights
- normalize
- resample
Parsing with particle filters

- implement top-down incremental particle filter parsing
- words are our observable variables
- grammar rules are our latent variables

Problem: we have discrete values!
Parsing with particle filters

- implement top-down incremental particle filter parsing
- words are our observable variables
- grammar rules are our latent variables

**Problem:** we have discrete values!

- so we use a transition table
- and an observation table

<table>
<thead>
<tr>
<th>rule</th>
<th>transition model*</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S → Aux NP VP</td>
</tr>
<tr>
<td>2</td>
<td>NP → Pro</td>
</tr>
<tr>
<td>3</td>
<td>VP → V NP NP</td>
</tr>
<tr>
<td>4</td>
<td>NP → Nom</td>
</tr>
<tr>
<td>5</td>
<td>NP → PNoun</td>
</tr>
<tr>
<td>6</td>
<td>Nom → Noun</td>
</tr>
<tr>
<td>7</td>
<td>Aux → Can</td>
</tr>
<tr>
<td>8</td>
<td>NP → Pro</td>
</tr>
<tr>
<td>9</td>
<td>Pro → you</td>
</tr>
<tr>
<td>10</td>
<td>Verb → book</td>
</tr>
<tr>
<td>11</td>
<td>PNoun → TWA</td>
</tr>
<tr>
<td>12</td>
<td>Noun → flights</td>
</tr>
</tbody>
</table>

*could be taken from the Brown corpus
Parsing *with particle filters*

- implement *top-down incremental* particle filter parsing
- **words** are our **observable** variables
- **grammar rules** are our **latent** variables

**Problem:** *we have discrete values!*

- so we use a **transition table**
- and an **observation table**

<table>
<thead>
<tr>
<th>rule</th>
<th>observation model* [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.15</td>
</tr>
<tr>
<td>2</td>
<td>.40</td>
</tr>
<tr>
<td>3</td>
<td>.05</td>
</tr>
<tr>
<td>4</td>
<td>.05</td>
</tr>
<tr>
<td>5</td>
<td>.35</td>
</tr>
<tr>
<td>6</td>
<td>.75</td>
</tr>
<tr>
<td>7</td>
<td>.40</td>
</tr>
<tr>
<td>8</td>
<td>.40</td>
</tr>
<tr>
<td>9</td>
<td>.40</td>
</tr>
<tr>
<td>10</td>
<td>.30</td>
</tr>
<tr>
<td>11</td>
<td>.40</td>
</tr>
<tr>
<td>12</td>
<td>.50</td>
</tr>
</tbody>
</table>

*could be taken from the Brown corpus*
Parsing with particle filters

sentence: "The woman brought the sandwich from the kitchen tripped."
Algorithm results

- tested 32 native English speakers who rate the complexity of sentences
- 20 NP/S sentences (structural ambiguity → presence/absence of “that”)
- 20 NP/Z sentences (structural ambiguity → absence/presence of a “comma”)
- rating from “very easy to understand (0)” to “very difficult to understand (10)
basic **NP/S unambiguous** controls example:

Long (U-L): Tom heard that the gossip about the neighbors wasn’t true.

Short (U-S): Tom heard that the gossip wasn’t true.

basic **NP/Z unambiguous** controls example:

Long (U-L): While the man hunted, the deer that was brown and graceful ran into the woods.

Short (U-S): While the man hunted, the deer ran into the woods.
conclusion

- presented a new incremental parsing algorithm based on particle filters
- provides a novel solution for the "diggin-in" effect problem
- provides a useful foundation for human sentence comprehension, because of the limited memory model
- must not convergence to global optimum because of sampling
A Kehler D Jurafsky, JH Martin.  
*Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.*  

Jay Earley.  
An efficient context-free parsing algorithm.  

Liang Huang and David Chiang.  
Better k-best parsing.  
Association for Computational Linguistics.

T Griffiths R Levy, F Reali, editor.  
*Modeling the effects of memory on human online sentence processing with particle filters*, Department of Psychology University of California, Berkeley, 2009.  
Proceedings of NIPS.

Andreas Stolcke.
An efficient probabilistic context-free parsing algorithm that computes prefix probabilities.


Whitney Tabor, Bruno Galantucci, and Daniel Richardson. Evidence for self-organized sentence processing: Local coherence effects.