Parsing
Left-Corner Parsing

Laura Kallmeyer

Heinrich-Heine-Universität Düsseldorf

Winter 2017/18
Table of contents

1 Motivation

2 Algorithm

3 Look-ahead

4 Chart Parsing
Motivation

Problems with pure TD/BU approaches:

- Top-Down does not check whether the actual input corresponds to the predictions made.
- Bottom-Up does not check whether the recognized constituents correspond to anything one might predict starting from S.

Mixed approaches help to overcome these problems:

- Left-Corner Parsing parses parts of the tree top down, parts bottom-up.
- Earley-Parsing is a chart-based combination of top-down predictions and bottom-up completions.
In a production $A \rightarrow X_1 \ldots X_k$, the first right-hand side element $X_1$ is called the **left corner** of this production.

Notation: $\langle A, X_1 \rangle \in LC$.

Idea:

- Parse the left corner bottom-up while parsing $X_2, \ldots, X_k$ top-down.

- In other words, in order to predict the subtree

\[
\begin{array}{c}
\text{A} \\
\text{X}_1 \quad \text{X}_2 \quad \ldots \quad \text{X}_k
\end{array}
\]

a parse tree for $X_1$ must already be there.
Algorithm (1)

We assume a CFG without $\varepsilon$-productions and without loops. We need the following three stacks:

- a stack $\Gamma_{compl}$ containing completed elements that can be used as potential left corners for applying new productions. Initial value: $w$

- a stack $\Gamma_{td}$ containing the top-down predicted elements of a rhs (i.e., the rhs without the left corner) Initial value: $S$

- a stack $\Gamma_{lhs}$ containing the lhs categories that are waiting to be completed. Once all the top-down predicted rhs symbols are completed, the category is moved to $\Gamma_{compl}$. Initial value: $\varepsilon$
Algorithm (2)

Item form \([\Gamma_{\text{compl}}, \Gamma_{td}, \Gamma_{lhs}]\) with

- \(\Gamma_{\text{compl}} \in (N \cup T)^*\),
- \(\Gamma_{td} \in (N \cup T \cup \{\$\})^*\) where \(\$\) is a new symbol marking the end of a rhs,
- \(\Gamma_{lhs} \in N^*\).

Whenever the symbols \(X_2, \ldots, X_k\) from a rhs are pushed onto \(\Gamma_{td}\), they are preceded by \(\$\) to mark the end of a rhs (i.e., the point where a category can be completed).

Axiom: \([w, S, \varepsilon]\)
Reduce can be applied if the top of $\Gamma_{compl}$ is the left corner $X_1$ of some rule $A \rightarrow X_1X_2 \ldots X_k$. Then $X_1$ is popped, $X_2 \ldots X_k$ is pushed onto $\Gamma_{td}$ and $A$ is pushed onto $\Gamma_{lhs}$:

Reduce: \[
\frac{[X_1\alpha, B\beta, \gamma]}{[\alpha, X_2 \ldots X_k \$, B\beta, A\gamma]} \quad A \rightarrow X_1X_2 \ldots X_k \in P, B \neq $

Once the entire righthand side has been completed (top of $\Gamma_{td}$ is $\$), the completed category is moved from $\Gamma_{lhs}$ to $\Gamma_{compl}$:

Move: \[
\frac{[\alpha, \$, \beta, A\gamma]}{[A\alpha, \beta, \gamma]} \quad A \in N
\]
Algorithm (4)

A completed category can be a left corner (then reduce is applied) or it can be the next symbol on the $\Gamma_{td}$ stack, then both can be popped:

Remove: $\frac{[X\alpha, X\beta, \gamma]}{[\alpha, \beta, \gamma]}$

The recognizer is successful if $\Gamma_{compl} = \Gamma_{td} = \Gamma_{lhs} = \varepsilon$:

Goal item: $[\varepsilon, \varepsilon, \varepsilon]$
Algorithm (5)

Example: Left Corner Parsing

<table>
<thead>
<tr>
<th>Productions:</th>
<th>$\Gamma_{compl}$</th>
<th>$\Gamma_{td}$</th>
<th>$\Gamma_{lhs}$</th>
<th>operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow aSa</td>
<td>bSb</td>
<td>c$</td>
<td>abcba</td>
<td>$S$</td>
</tr>
<tr>
<td>input $w = abcba$</td>
<td>bcba</td>
<td>$Sa$</td>
<td>$S$</td>
<td>$S$</td>
</tr>
<tr>
<td></td>
<td>cba</td>
<td>$SbSa$</td>
<td>$SS$</td>
<td>$SS$</td>
</tr>
<tr>
<td></td>
<td>ba</td>
<td>$SbSa$</td>
<td>$SS$</td>
<td>$SS$</td>
</tr>
<tr>
<td></td>
<td>Sba</td>
<td>$SbSa$</td>
<td>$SS$</td>
<td>$SS$</td>
</tr>
<tr>
<td></td>
<td>ba</td>
<td>$bSa$</td>
<td>$SS$</td>
<td>$SS$</td>
</tr>
<tr>
<td></td>
<td>a</td>
<td>$S$</td>
<td>$SS$</td>
<td>$SS$</td>
</tr>
<tr>
<td></td>
<td>Sa</td>
<td>$S$</td>
<td>$S$</td>
<td>$S$</td>
</tr>
<tr>
<td></td>
<td>a</td>
<td>$aS$</td>
<td>$S$</td>
<td>$S$</td>
</tr>
<tr>
<td></td>
<td>$\varepsilon$</td>
<td>$S$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td>$S$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td></td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
</tr>
</tbody>
</table>
Problematic for left-corner parsing:

- \( \varepsilon \)-productions: there is no left corner that can trigger a reduce step with an \( \varepsilon \)-production. If we allow \( \varepsilon \)-productions to be predicted in reduce steps without a left corner, we would add them an infinite number of times.

- loops: as in the LL-parsing case, loops can cause an infinite sequence of reduce and move steps. This problem is already avoided with the item-based formulation since we would only try to create the same items again.

Both problems can be overcome using the chart-based version with dotted productions described later.
Look-ahead (1)

Idea:

- build the reflexive transitive closure $LC^*$ of the left corner relation $LC$,
- before applying reduce, check whether the top of $\Gamma_{td}$ stands in the relation $LC^*$ to the lhs of the new production we predict:

Reduce: \[
\frac{[X_1\alpha, B\beta, \gamma]}{[\alpha, X_2 \ldots X_k B\beta, A\gamma]} \quad A \rightarrow X_1X_2 \ldots X_k \in P, \langle B, A \rangle \in LC^*
\]

Difference between $LC^*$ and First: $LC^*$ for a given non-terminal can be non-terminals and terminals, while the First sets contain only terminals.

$LC^* = \{ \langle A, X \rangle | A \Rightarrow X\alpha \}$
Example:

\[
\begin{align*}
VP & \rightarrow V \ NP, \ VP \rightarrow VP \ PP, \ V \rightarrow \text{sees}, \\
NP & \rightarrow \text{Det} \ N, \ \text{Det} \rightarrow \text{the}, \ N \rightarrow N \ PP, \ N \rightarrow \text{girl}, \ N \rightarrow \text{telescope}, \\
PP & \rightarrow P \ NP, \ P \rightarrow \text{with}
\end{align*}
\]

\[
LC:
\langle \text{VP, V} \rangle, \langle \text{VP, VP} \rangle, \langle \text{V, sees} \rangle \langle \text{NP, Det} \rangle, \langle \text{Det, the} \rangle, \\
\langle \text{N, N} \rangle, \langle \text{N, girl} \rangle, \langle \text{N, telescope} \rangle, \langle \text{PP, P} \rangle, \langle \text{P, with} \rangle
\]

\[LC^* = LC \cup:
\{ \langle \text{VP, sees} \rangle, \langle \text{V, V} \rangle, \langle \text{NP, NP} \rangle, \langle \text{NP, the} \rangle, \\
\langle \text{PP, PP} \rangle, \langle \text{PP, with} \rangle, \langle \text{P, P} \rangle \} \]
Chart Parsing (1)

Problem of left corner parsing: non-deterministic.

In order to avoid computing partial results several times, we can use tabulation, i.e., adopt chart parsing.

Items we need to tabulate:

- Completely recognized categories: passive items $[X, i, l]$
- Partially recognized productions: active items $[A \rightarrow \alpha \bullet \beta, i, l]$
  with $\alpha \in (N \cup T)^+, \beta \in (N \cup T)^*$

($i$ index of first terminal in yield, $l$ length of the yield)
Let us again assume a CFG without $\varepsilon$-productions. We start with the initial items $[w_i, i, 1]$. The operations reduce, remove and move are then as follows:

- **Reduce:** If $[X_1, i, l]$ and $A \rightarrow X_1X_2 \ldots X_k \in P$, then we add $[A \rightarrow X_1 \bullet X_2 \ldots X_k, i, l]$.

- **Move:** If $[A \rightarrow X_1X_2 \ldots X_k \bullet, i, l]$, then we add $[A, i, l]$. 

- **Remove:** If $[X, i, l]$ and $[A \rightarrow \alpha \bullet X\beta, j, i - j]$ then we add $[A \rightarrow \alpha X \bullet \beta, j, i - j + l]$. 

Chart Parsing (3)

Parsing Schema:

Scan: \([w_i, i, 1] \quad 1 \leq i \leq n\)

Reduce: \(\frac{[X, i, l]}{[A \rightarrow X \bullet \alpha, i, l]} \quad A \rightarrow X\alpha \in P\)

Remove: \(\frac{[A \rightarrow \alpha \bullet X\beta, i, l_1], [X, j, l_2]}{[A \rightarrow \alpha X \bullet \beta, i, l_1 + l_2]} \quad j = i + l_1\)

Move: \(\frac{[A \rightarrow \alpha X\bullet, i, l]}{[A, i, l]}\)

Goal item: \([S, 1, n].\)

(This is actually the same algo as the CYK with dotted productions seen earlier in the course, except for different names of the rules and a different use of indices.)
## Chart Parsing (4)

### Example: Left Corner Chart Parsing

Productions: \( S \rightarrow aSa \mid bSb \mid c \), input \( w = abcba \).

<table>
<thead>
<tr>
<th>item(s)</th>
<th>rule</th>
<th>antecedens items</th>
</tr>
</thead>
<tbody>
<tr>
<td>([a, 1, 1], [b, 2, 1], [c, 3, 1], [b, 4, 1], [a, 5, 1]) (axioms)</td>
<td>reduce</td>
<td>([a, 1, 1])</td>
</tr>
<tr>
<td>(S \rightarrow a \bullet Sa, 1, 1)</td>
<td>reduce</td>
<td>([b, 2, 1])</td>
</tr>
<tr>
<td>(S \rightarrow b \bullet Sb, 2, 1)</td>
<td>reduce</td>
<td>([c, 3, 1])</td>
</tr>
<tr>
<td>(S \rightarrow c \bullet , 3, 1)</td>
<td>reduce</td>
<td>([b, 4, 1])</td>
</tr>
<tr>
<td>(S \rightarrow a \bullet Sb, 4, 1)</td>
<td>reduce</td>
<td>([a, 5, 1])</td>
</tr>
<tr>
<td>(S \rightarrow b \bullet Sa, 5, 1)</td>
<td>move</td>
<td>([S \rightarrow c \bullet , 3, 1])</td>
</tr>
<tr>
<td>([S, 3, 1])</td>
<td>remove</td>
<td>([S \rightarrow b \bullet Sb, 2, 1], [S, 3, 1])</td>
</tr>
<tr>
<td>(S \rightarrow bS \bullet b, 2, 2)</td>
<td>remove</td>
<td>([S \rightarrow bS \bullet b, 2, 2], [b, 4, 1])</td>
</tr>
<tr>
<td>(S \rightarrow bSb \bullet , 2, 3)</td>
<td>move</td>
<td>([S \rightarrow bSb \bullet , 2, 3])</td>
</tr>
<tr>
<td>([S, 2, 3])</td>
<td>remove</td>
<td>([S \rightarrow a \bullet Sa, 1, 1], [S, 2, 3])</td>
</tr>
<tr>
<td>(S \rightarrow aS \bullet a, 1, 4)</td>
<td>remove</td>
<td>([S \rightarrow aS \bullet a, 1, 4], [a, 5, 1])</td>
</tr>
<tr>
<td>(S \rightarrow aSa \bullet , 1, 5)</td>
<td>move</td>
<td>([S \rightarrow aSa \bullet , 1, 5])</td>
</tr>
<tr>
<td>([S, 1, 5])</td>
<td>move</td>
<td>([S \rightarrow aSa \bullet , 1, 5])</td>
</tr>
</tbody>
</table>
Conclusion

- The left corner of a production is the first element of its rhs.
- Predict a production only if its left corner has already been found.
- In general non-deterministic.
- Problematic for $\varepsilon$-productions and loops.
- Can be implemented as a chart parser with passive and active items.
- In the chart parser, $\varepsilon$-productions can be dealt with (they require an additional Scan-$\varepsilon$ rule) and loops are no longer a problem.