Regular approximations through labeled bracketing (revised version)

Yli-Jyrä, Anssi Mikael

2003


http://hdl.handle.net/10138/33871

Downloaded from Helda, University of Helsinki institutional repository.

This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.
Please cite the original version.
Regular Approximations through Labeled Bracketing

Anssi YLI-JYRÄ

The PhD programme of Nordic Funding for Advanced studies (NorFA)
Department of General Linguistics, University of Helsinki
The KIT graduate school for language tecnology, Finland

FG Vienna 2003
1. motivation
   - “lost” species of regular approximations

2. labeled bracketing
   - two styles: mark rules versus mark categories
   - two models: Dyck versus $\delta$

3. previous representations based on the bracketing models

4. new representation theorem
   - explanation
   - approximations

5. concluding remarks
Some desirable properties:

**Compact representation** – Can approximations of large grammars be implemented and accessed efficiently through a compact representation?

**Exactness** – Can it assign correct CF structures up to any fixed depth of center-embedding?
Some desirable properties:

**Compact representation** – Can approximations of large grammars be implemented and accessed efficiently through a compact representation?

**Exactness** – Can it assign correct CF structures up to any fixed depth of center-embedding?

<table>
<thead>
<tr>
<th>Compact representation</th>
<th>no compact representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact</td>
<td>????????</td>
</tr>
<tr>
<td></td>
<td>approximation through PDA</td>
</tr>
<tr>
<td></td>
<td>(Johnson 1998)</td>
</tr>
<tr>
<td>not exact</td>
<td>superset approximation</td>
</tr>
<tr>
<td></td>
<td>(Mohri&amp;Nederhof 2001)</td>
</tr>
<tr>
<td></td>
<td>various approximations</td>
</tr>
<tr>
<td></td>
<td>e.g. the RTN method</td>
</tr>
</tbody>
</table>

Exactness seems to be against compactness. Is is really so?
3 – Labeled bracketing

Exactness criterion for approximations requires that structural description is assigned to strings. A classical way to do this is to decorate strings with labeled brackets of an $B_n$ to denote phrase boundaries.

Let $B_n$ be an alphabet $[1,1], [2,2], \ldots; [n,n]$ which consists of even number of letters (brackets) grouped into pairs.

Labeled bracketing can be done in two different ways:
3 – Labeled bracketing

Exactness criterion for approximations requires that structural description is assigned to strings. A classical way to do this is to decorate strings with labeled brackets of an $B_n$ to denote phrase boundaries.

Let $B_n$ be an alphabet $[1, 1], [2, 2], \ldots; [n, n]$ which consists of even number of letters (brackets) grouped into pairs.

**Labeled bracketing can be done in two different ways:**

- In the **Chomsky-Schützenberger representation** of context-free languages, each CF production correspond to different pair of brackets,

  $$[S_1 [NP_5 Bill NP_5] [VP_2 runs VP_2] S_1]$$
3 – Labeled bracketing

Exactness criterion for approximations requires that structural description is assigned to strings. A classical way to do this is to decorate strings with labeled brackets of an $B_n$ to denote phrase boundaries.

Let $B_n$ be an alphabet $[1, 1], [2, 2], \ldots; [n, n]$ which consists of even number of letters (brackets) grouped into pairs.

Labeled bracketing can be done in two different ways:

- In the **Chomsky-Schützenberger representation** of context-free languages, each CF production correspond to different pair of brackets,

  $$[S_1 [NP_5 \text{Bill} \ NP_5] [VP_2 \text{runs} \ VP_2] S_1]$$

- In a **Context-Free Bracketing Languages (CFBL)** each nonterminal corresponds to unique pair of brackets regardless of the number of productions

  $$[S [NP \text{Bill} \ NP] [VP \text{runs} \ VP] S]$$
The **Dyck language** $D_n$ over the alphabet $B_n$ is the language generated by the CF grammar

$$
S \rightarrow SS; \quad S \rightarrow \lambda; \quad S \rightarrow [1 \ S \ 1]; \quad S \rightarrow [2 \ S \ 2]; \ldots \quad S \rightarrow [k \ S \ k]
$$
The Dyck language $D_n$ over the alphabet $B_n$ is the language generated by the CF grammar

$$S \to SS; \ S \to \lambda; \ S \to [1 \ S \ 1]; \ S \to [2 \ S \ 2]; \ldots \ S \to [k \ S \ k]$$

A Dyck language $D_n$ models behavior of a PDA with an $n$-symbol stack alphabet:
The $\delta$-language (our term) models only bracketing depth in strings.

It equals to an inverse homomorphemic image of $D_1$:

$$\delta = h^{-1}(D_1), \text{ where}$$

$$h^{-1}([_1]) = \{[1, \ldots, [n]\},$$

$$h^{-1}(1]) = \{1], \ldots, n]\},$$

$$h^{-1}(\lambda) = (\Sigma - h([_1]) - h(1])^*$$

An example of a string in this language:

$$a [1 \ aa [6 \ a \ 2] \ a \ 19] \ aa [8 \ a \ 9] \ aa$$
The Chomsky & Schützenberger (1963) theorem. Every CFL can be represented as $h(D \cap R)$, that is, as a homomorphic image of an intersection of

- a Dyck language $D$ (accounts for non-local properties), and

- a local regular language $R$ (takes care of local properties),

Theorem by Wrathall (1977). $D_r$ equals to an intersection $C_1 \cap C_2 \cdots \cap C_r$, where $C_i$ is a constraint checking that $[i$ is closed with $i]$. Each matching constraint $C_i$ can be defined from $\delta$, which is, in turn, defined from $D_1$.

These theorems admit regular approximations through regular restrictions of the respective bracketing model ($D$ or $\delta$). The Chomsky-Schützenberger-Wrathall presentation is compact.
7 – The key contribution

The prior art. The Chomsky-Schützenberger-Wrathall presentation.

- compact representations
- the bracketings tell you which CF rules you have used - too informative.

Our new representation theorem. Every CFBL equals to an intersection of

- an “axiom” language of the form \([s_1 \delta s_1] \cup \cdots \cup [s_s \delta s_s]\), and
- a bracketing restriction constraint respectively for each bracket type.

E.g. \(S \to NP\ VP\) corresponds to restriction \([s _ s] \Rightarrow [NP \delta NP]\ [VP \delta VP]\).

We can reduce the number of different brackets by splitting the local language \(R\) into separate restrictions and by combining \(\delta\) with them.
8 – How the bracketing restrictions work?

\[
[s [np Bill np] [vp hit [np George np] vp] s] \quad \text{the "axiom" constraint: } [s \delta s] \& \delta
\]
8 – How the bracketing restrictions work?

Regular Approximations through Labeled Bracketing

Regular Approximations through Labeled Bracketing

Formal Grammar 2003
Anssi YLI-JYRÄ
8 - How the bracketing restrictions work?

The "axiom" constraint: \[ [s \delta s] \& \delta \]

Bracketing restriction constraints:

\[ [s \_ s] \Rightarrow [np \delta np][vp \delta vp] \]

\[ [vp \_ vp] \Rightarrow hit [np \delta np] \]

\[ ran \]

\[ gave [np \delta np] \]
8 – How the bracketing restrictions work?

The "axiom" constraint: \([s \delta s] \& \delta\)

Bracketing restriction constraints:

\([s - s] \Rightarrow [np \delta np][vp \delta vp]\)
\([vp - vp] \Rightarrow hit [np \delta np]\)
\(\mid ran\)
\(\mid gave [np \delta np]\)
\([np - np] \Rightarrow Bill\)
\(\mid George\)
9 – The definition of the bracketing restriction operator

Bracketing restriction constraint has (prototypically) the following syntax:

\[ \alpha \_ \beta \Rightarrow \gamma, \text{ where } \alpha \in B_{[} = \{ [1, \ldots, [n, \} \quad , \beta \in B_{]} = \{ 1], \ldots, n], \} \quad , \gamma \subseteq \delta. \]
9 – The definition of the bracketing restriction operator

Bracketing restriction constraint has (prototypically) the following syntax:
\[ \alpha \_ \beta \Rightarrow \gamma, \text{ where } \alpha \in B[ = \{ [1, \ldots, [n, ] \}, \beta \in B] = \{ 1], \ldots, n], \}, \gamma \subseteq \delta. \]

It accepts string \( w \in \Sigma^* \) if and only if it belongs to the language
\[ \Sigma^* - \Sigma^* \alpha \overline{\gamma} \beta \Sigma^* \text{ where } \overline{\gamma} = \delta - \gamma \]
The definition of the bracketing restriction operator

Bracketing restriction constraint has (prototypically) the following syntax:

\[ \alpha \_ \beta \Rightarrow \gamma, \text{ where } \alpha \in B_1 = \{1, \ldots, [n], \} \quad \beta \in B_2 = \{1, \ldots, n\}, \] \quad \gamma \subseteq \delta. \]

It accepts string \( w \in \Sigma^* \) if and only if it belongs to the language

\[ \Sigma^* - \Sigma^* \alpha \bar{\gamma} \beta \Sigma^* \quad \text{where} \quad \bar{\gamma} = \delta - \gamma \]

The subset \( (\alpha \_ \beta \Rightarrow \gamma) \cap \delta \) is generated by a CFG with the following set of productions:

\[
\{ S \rightarrow \lambda; S \rightarrow SS; S \rightarrow T; S \rightarrow \alpha \gamma' \beta \} \cup \{ S \rightarrow \alpha' S \beta' \mid \alpha \beta \neq \alpha' \beta' \in B_1 B_2 \}
\]

where \( T = \Sigma - B_1 - B_2 \) and

\( \gamma' \) is obtained syntactically from \( \gamma \)-expression by substituting each \( \delta \)-symbol with \( S \).
10 – Why the obtained presentation is compact?

The complexity of a set of approximated bracketing restrictions has the following parameters:

- \( n \) = the number of nonterminals in the CF grammar being approximated
- \( k \) = the depth of the nested (square) brackets
- \( c \) = the maximal state complexity of the \( \gamma \), where \( \delta \) is a symbol rather than being substituted with a language

The approximated grammar can be represented as \( n \) bracketing restriction constraints. The state complexity of each constraint is independent of \( n \).

**Lazy grammar compilation.** We suspend computing the intersection of bracketing restriction constraints to the parsing time.
The total size of the compact representation is exponential to $k$ (depth), but linear to $n$ (nonterminals).
12 – State complexity of individual constraints

<table>
<thead>
<tr>
<th>$\delta^k$</th>
<th>definition</th>
<th>$\gamma = [1\delta_1] \ldots [2\delta_2]$</th>
<th>$s = 4$</th>
<th>$s = 8$</th>
<th>$s = 16$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta^2$</td>
<td>$\delta (L\delta^{-1} R\delta^{-1})^*$</td>
<td>12 (8)</td>
<td>16 (12)</td>
<td>(20)</td>
<td>(36)</td>
</tr>
<tr>
<td>$\delta^3$</td>
<td>$\delta^2 (L\delta^{-2} R\delta^{-2})^*$</td>
<td>34 (16)</td>
<td>50 (26)</td>
<td>(46)</td>
<td>(86)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$\delta^6$</td>
<td>$\delta^5 (L\delta^{-5} R\delta^{-5})^*$</td>
<td>724 (148)</td>
<td>1’778 (516)</td>
<td>(2’332)</td>
<td>(12’972)</td>
</tr>
<tr>
<td>$\delta^7$</td>
<td>$\delta^6 (L\delta^{-6} R\delta^{-6})^*$</td>
<td>(296)</td>
<td>(1’282)</td>
<td>(7’094)</td>
<td>(46’174)</td>
</tr>
</tbody>
</table>

The state complexity can be reduced with an optimization (in parenthesis).

Conjecture: the state complexity the bracketing restriction constraint depend polynomially from the state complexity of its right-hand side language $\gamma$. 

Regular Approximations through Labeled Bracketing

Anssi YLI-JYRÄ

Formal Grammar 2003
13 – Further topics

- Supplements at www.ling.helsinki.fi/~ayliijyra/
  - Errata (for the paper)
  - Experiments (a small grammar for a fragment of English using XFST)
13 – Further topics

- Supplements at www.ling.helsinki.fi/~ayliijyra/
  - **Errata** (for the paper)
  - **Experiments** (a small grammar for a fragment of English using XFST)

- Other issues:
  - **Coverage** - We handled initial and final embedding
  - **Efficiency** - substantial optimizations are available
  - **Star-Freeness** - this property hold for the approximation
The main contributions:

- the Chomsky-Schützenberger style representation theorem
  - the bracketing restriction operator
  - characterizing sets of CF bracketings without derivation trees
14 – Conclusions

The main contributions:

• the Chomsky-Schützenberger style representation theorem
  – the bracketing restriction operator
  – characterizing sets of CF bracketings without derivation trees

• regular approximations with a new compact representation
  – combine exactness to the compact representation
  – initial and final embedding handled
14 – Conclusions

The main contributions:

- the Chomsky-Schützenberger style representation theorem
  - the bracketing restriction operator
  - characterizing sets of CF bracketings without derivation trees
- regular approximations with a new compact representation
  - combine exactness to the compact representation
  - initial and final embedding handled

Future

- more serious experiments
- more strong generative power