Tree Automata in Parsing and Machine Translation

Andreas Maletti

Institute for Natural Language Processing
University of Stuttgart

Highlights — September 9, 2016
Parsing

determining the syntactic structure of a sentence
subject to a given theory of syntax encoded in the training data
Parsing

- determining the syntactic structure of a sentence
- subject to a given theory of syntax \textit{encoded in the training data}

Theories

- Constituent syntax
- Dependency syntax
- …
Constituent Parsing

Penn tree bank (50,000 trees for English)

Tree Automata in Parsing and Machine Translation
Andreas Maletti
Example: We must bear in mind the Community as a whole
Constituent Parsing

Example: We must bear in mind the Community as a whole

POS-tag: part-of-speech tag, “class” of a word
Constituent Parsing

Short history

…  • Chomskyan approach (perfect analysis, poor coverage)
    • hand-crafted CFG, TAG refined via POS tags
    • corrections and selection by human annotators

90s  • Statistical approach (cheap, automatically trained)
    • Penn and WSJ tree bank (1M and 30M words)
    • automatically obtained weighted CFG

00s  • Subcategorization
    • manual, e.g., Collins (1999), Stanford (2003)
    • manual and reranking, e.g. BLLIP (2005)
    • automatic, e.g. Berkeley (2007)
Subcategorization

These CFG derivations

also admit
Subcategorization

Tags:
- official tags often conservative
  - English: ≈ 50 tags
  - German: ≫ 200 tags

ADJA-Sup-Dat-Sg-Fem
Subcategorization

My dog sleeps

Tags:
- official tags often conservative
  - English: \( \approx 50 \) tags
  - German: \( \gg 200 \) tags
- all modern parsers use refined tags \( \rightarrow \) subcategorization
Subcategorization

Tags:
- official tags often conservative
  - English: ≈ 50 tags
  - German: ≫ 200 tags
- all modern parsers use refined tags → subcategorization
- but return parse over official tags → relabeling
Definition

Tree automaton \((Q, \Sigma, I, R)\)

- finite set \(Q\) of states
- finite set \(\Sigma\) of terminals
- initial states \(I \subseteq Q\)
- finite set \(R\) of rules of the form \(q \rightarrow \sigma(q_1, \ldots, q_k)\)  
  \(\sigma \in \Sigma, k \geq 0, q, q_1, \ldots, q_k \in Q\)

Example rules

\[
\begin{align*}
q_4 & \rightarrow q_5 \, \text{VP} \, q_2 \, q_3 \\
q_0 & \rightarrow q_1 \, S \, q_4 \\
q_0 & \rightarrow q_6 \, S \, q_2
\end{align*}
\]
Tree Automaton

Derivation semantics

\( \xi \Rightarrow \zeta \) for sentential forms \( \xi, \zeta \in T_\Sigma(Q) \)
if there exist leaf position \( w \) in \( \xi \) and rule \( q \rightarrow r \) in \( R \)

\[
\xi = \xi[q]_w \quad \zeta = \xi[r]_w
\]

Recognized tree language

\[
\{ t \in T_\Sigma \mid \exists q_0 \in I: q_0 \Rightarrow^* t \}\]
Example derivation

\[ q_0 \Rightarrow S \Rightarrow q_1 \]

Example rules

\[ q_4 \Rightarrow VP \Rightarrow q_5 q_2 q_3 \]

\[ q_0 \Rightarrow S \Rightarrow q_1 q_4 \]

\[ q_0 \Rightarrow S \Rightarrow q_6 q_2 \]
Constituent Parsing

Grammar with subcategorization (grammar with relabeling)

- a grammar $G$ generating $L(G) \subseteq T_{\Sigma}(W)$ (subcategorized trees)
- a (total) mapping $\rho : \Sigma \rightarrow \Delta$ (functional relabeling)
Constituent Parsing

Grammar with subcategorization (grammar with relabeling)
- a grammar $G$ generating $L(G) \subseteq T_\Sigma(W)$ (subcategorized trees)
- a (total) mapping $\rho : \Sigma \rightarrow \Delta$ (functional relabeling)

Semantics
$L(G, \rho) = \rho(L(G)) = \{ \rho(t) | t \in L(G) \}$

Language class: REL

Tree Automata in Parsing and Machine Translation
Andreas Maletti · 11
Constituent Parsing

Grammar with subcategorization (grammar with relabeling)

- a grammar $G$ generating $L(G) \subseteq T_{\Sigma}(W)$ (subcategorized trees)
- a (total) mapping $\rho: \Sigma \rightarrow \Delta$ (functional relabeling)

My dog sleeps

Semantics $L(G, \rho) = \rho(L(G)) = \{\rho(t) | t \in L(G)\}$

Language class: REL ($L$)

Tree Automata in Parsing and Machine Translation
Andreas Maletti · 11
Constituent Parsing

Grammar with subcategorization (grammar with relabeling)
- a grammar $G$ generating $L(G) \subseteq T_{\Sigma}(W)$ (subcategorized trees)
- a (total) mapping $\rho : \Sigma \rightarrow \Delta$ (functional relabeling)

Semantics

$$L(G, \rho) = \rho(L(G)) = \{ \rho(t) \mid t \in L(G) \}$$

Language class: $\text{REL}(\mathcal{L})$ for language class $\mathcal{L}$
Constituent Parsing

**Theorem**

\[ \text{REL}(\text{wCFL}) = \text{REL}(\text{wTSL}) = \text{wRTL} \]

\[
\begin{align*}
\text{wRTL} & \quad [\text{wTA}] \\
\text{REL}(\text{wTSL}) & \quad [\text{wTSG}_{\text{sub}}] \\
\text{wTSL} & \quad [\text{wTSG}] \\
\text{wCFL} & \quad [\text{wCFG}] \\
\end{align*}
\]

Hence: subcategorization \( \approx \) finite-state Tree Automata in Parsing and Machine Translation

Andreas Maletti · 12
Constituent Parsing

Theorem

\[ \text{REL}(w_{\text{CFL}}) = \text{REL}(w_{\text{TSL}}) = w_{\text{RTL}} \]
Constituent Parsing

Theorem

\[ \text{REL}(w_{\text{CFL}}) = \text{REL}(w_{\text{TSL}}) = w_{\text{RTL}} \]

Hence: subcategorization \( \approx \) finite-state
Constituent Parsing

Comparison

1 rule of subcategorized grammar:

\[ S-1 \rightarrow \text{ADJP-2 } S-1 \quad \text{weight: } 0.003545 \]

with relabeling \( \rho(S-1) = S, \ldots \)
Constituent Parsing

Comparison

1. rule of subcategorized grammar:

\[ S-1 \rightarrow \text{ADJP-2 } S-1 \quad \text{weight: } 0.003545 \]

with relabeling \( \rho(S-1) = S, \ldots \)

2. corresponding rule of tree automaton

\[ S-1 \rightarrow S(\text{ADJP-2}, S-1) \quad \text{weight: } 0.003545 \]
Constituent Parsing

State-of-the-art models

- CFG with subcategorization (CFG$_{sub}$)
- Tree substitution grammars with subcategorization (TSG$_{sub}$) [Shindo et al., 2012]

(both as expressive as weighted tree automata)
- Other models
Constituent Parsing

State-of-the-art models

- CFG with subcategorization ($\text{CFG}_{\text{sub}}$)
- Tree substitution grammars with subcategorization ($\text{TSG}_{\text{sub}}$) [Shindo et al., 2012]
- (both as expressive as weighted tree automata)
- Other models

<table>
<thead>
<tr>
<th>grammar model</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{wCFG}$</td>
<td>72.6</td>
</tr>
<tr>
<td>$\text{wTSG}$ [Cohn et al., 2010]</td>
<td>84.7</td>
</tr>
<tr>
<td>$\text{wCFG}_{\text{sub}}$ [Petrov, 2010]</td>
<td>91.8</td>
</tr>
<tr>
<td>$\text{wTSG}_{\text{sub}}$ [Shindo et al., 2012]</td>
<td>92.4</td>
</tr>
</tbody>
</table>
Advances in NLP

- best learning algorithms from positive data (state splitting & EM)
- fastest evaluators of weighted tree automata (coarse-to-fine parsing)
- fastest $n$-best derivation extraction
- …
Literature

Thatcher & Wright (1968): Generalized Finite Automata Theory with an Application to a Decision Problem of Second-Order Logic.


Gécseg & Steinby (1997): Tree Languages.
In Handbook of Formal Languages 3: 1–68, Springer
Dependency Parsing

John saw a dog yesterday which was a Yorkshire Terrier.

Problems

Given edge-weighted directed graph, extract “best” edge cover

- (general) [Edmonds, 1965]
- that is a tree [Chu-Liu & Edmonds, 1965–1967]
- that is projective tree [Eisner, 1996]
- that is acyclic
- that is a tree with page-number 2 [Gómez-Rodríguez & Nivre, 2013]
- that has page-number $k \geq 2$ NP-hard [Kuhlmann & Jonsson, 2015]
- that is a tree with page-number $k \geq 3$ open
The lexicon generates string language $\mathcal{L}$ with $\mathcal{L} \cap c^+ d^+ e^+ = \{c^i d^i e^i | i \geq 1\}$ for goal item $D$.

$L(c) = \{C\}$

$L(d) = \{D/E\backslash C, D/E/D\backslash C\}$

$L(e) = \{E\}$
## Problems

Under a suitable relabeling, characterize the set of valid proof trees:

- for just applications → sub-regular tree languages
- for compositions of order 1 → open (probably still regular)
- for compositions of order \( k \geq 2 \) → open
- for arbitrary compositions → context-free tree language
1. The room it is not narrowly was a simple, bathtub was also attached.
2. Wi-fi, TV and I was available.
3. Church looked When morning awake open the curtain.
4. When looking at often, wives, went out and is invited to try to go [...].
5. But was a little cold, morning walks was good.
1. The room it is not narrowly was a simple, bathtub was also attached.
2. Wi-fi, TV and I was available.
3. Church looked When morning awake open the curtain.
4. When looking at often, wives, went out and is invited to try to go [...].
5. But was a little cold, morning walks was good.
Short History

Timeline

1960  -  Dark age
- rule-based systems (e.g., SYSTRAN)
- Chomskyan approach (perfect translation, poor coverage)

1991  -  Reformation
- phrase-based and syntax-based systems
- statistical approach (cheap, automatically trained)

2016  -  Potential future
- semantics-based systems (e.g., FrameNet-based)
- semi-supervised, statistical approach
- basic understanding of (translated) text
Machine Translation

Vauquois triangle:

Translation model: string-to-string
Machine Translation

Vauquois triangle:

Translation model: string-to-tree
Machine Translation

Vauquois triangle:

Translation model: tree-to-tree
### Training data

- parallel corpus
- word alignments
- parse trees

(For syntax-based systems)
Machine Translation

Training data
- parallel corpus
- word alignments
- parse trees (for syntax-based systems)

Parallel corpus
linguistic resource containing (sentence-by-sentence) example translations
I would like your advice about Rule 143 concerning inadmissibility.

Können Sie mir eine Auskunft zu Artikel 143 im Zusammenhang mit der Unzulässigkeit geben.
parallel corpus, word alignments, parse tree

via GIZA++ [Och, Ney: A systematic comparison of various statistical alignment models. Computational Linguistics 29(1), 2003]
Machine Translation

parallel corpus, word alignments, parse tree

Könnten Sie mir eine Auskunft zu Artikel 143 im Zusammenhang mit der Unzulässigkeit geben

Weighted Synchronous Grammars

Synchronous tree substitution grammar: productions $N \rightarrow (r, r_1)$

- nonterminal $N$
- right-hand side $r$ of context-free grammar production
- right-hand side $r_1$ of tree substitution grammar production

Weighted Synchronous Grammars

Synchronous tree substitution grammar: productions $N \rightarrow (r, r_1)$

- nonterminal $N$
- right-hand side $r$ of context-free grammar production
- right-hand side $r_1$ of tree substitution grammar production
- (bijective) synchronization of nonterminals

Synchronous Grammars

Production application

Selection of synchronous nonterminals
Synchronous Grammars

Production application

Selection of synchronous nonterminals
Synchronous Grammars

Production application

1. Selection of synchronous nonterminals
2. Selection of suitable production

S → PPER KOUS PPER advice PP

KOUS →

would like

Könnten

Tree Automata in Parsing and Machine Translation

Andreas Maletti · 26
Synchronous Grammars

Production application

1. Selection of synchronous nonterminals
2. Selection of suitable production
3. Replacement on both sides
Synchronous Grammars

Production application

1 synchronous nonterminals
Synchronous Grammars

Production application

1 synchronous nonterminals
Synchronous Grammars

Production application
1 synchronous nonterminals
2 suitable production
Synchronous Grammars

Production application

1. synchronous nonterminals
2. suitable production
3. replacement
I would like your advice about Rule 143 concerning inadmissibility following [Galley, Hopkins, Knight, Marcu: What’s in a translation rule? Proc. NAACL, 2004]
would like your advice about Rule 143 concerning inadmissibility

I would like your advice about Rule 143 concerning inadmissibility following [Galley, Hopkins, Knight, Marcu: What’s in a translation rule? Proc. NAACL, 2004]
I would like your advice about Rule 143 concerning inadmissibility following [Galley, Hopkins, Knight, Marcu: What’s in a translation rule? Proc. NAACL, 2004]
would like your advice about Rule 143 concerning inadmissibility

Removal of extractable production:

I would like your advice about Rule 143 concerning inadmissibility.
Removing extractable production:

Können Sie eine Auskunft zu Artikel 143 geben?
PPER would like your advice about Rule 143

Could you give an answer to Article 143?
KÖNNTEN Sie eine Auskunft zu Artikel 143 geben.
Könnten Sie eine Auskunft zu Artikel 143 geben?
PPER would like your advice about Rule 143.
Könnten Sie eine Auskunft zu Artikel 143 geben?
PPER would like your advice about Rule 143.

Könnten Sie eine Auskunft zu Artikel 143 geben?

Tree Automata in Parsing and Machine Translation  
Andreas Maletti  ·  30
Synchronous Tree Substitution Grammars

Advantages

- very simple
- implemented in framework ‘Moses’
  
- “context-free”
Synchronous Tree Substitution Grammars

Advantages

- very simple
- implemented in framework ‘Moses’
- “context-free”

Disadvantages

- problems with discontinuities
- composition and binarization not possible
- “context-free”
### Evaluation

**English → German translation task:** (higher BLEU is better)

<table>
<thead>
<tr>
<th>Type</th>
<th>System</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>vanilla</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vanilla WMT 2013</td>
<td>15.2</td>
<td>19.4</td>
</tr>
<tr>
<td>vanilla WMT 2015</td>
<td>14.5</td>
<td>—</td>
</tr>
</tbody>
</table>

Observation

- syntax-based systems competitive with manual adjustments
- much less so for vanilla systems
- very unfortunate situation [more supervision yields lower scores]
very specific production

every production for ‘advice’ contains sentence structure

(syntax “in the way”)
Synchronous Grammars

Synchronous multi tree substitution grammar: \( N \rightarrow (r, \langle r_1, \ldots, r_n \rangle) \)


- nonterminal \( N \)
- right-hand side \( r \) of context-free grammar production
- right-hand sides \( r_1, \ldots, r_n \) of regular tree grammar production
Synchronous Grammars

Synchronous multi tree substitution grammar: $N \rightarrow (r, \langle r_1, \ldots, r_n \rangle)$

- nonterminal $N$
- right-hand side $r$ of context-free grammar production
- right-hand sides $r_1, \ldots, r_n$ of regular tree grammar production


ART-NN-VV →

- eine
- Auskunft
- geben

ART

NN

VV
Synchronous Grammars

Synchronous multi tree substitution grammar: $N \rightarrow (r, \langle r_1, \ldots, r_n \rangle)$

- nonterminal $N$
- right-hand side $r$ of context-free grammar production
- right-hand sides $r_1, \ldots, r_n$ of regular tree grammar production


**ART-NN-VV** →

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ART</td>
<td>NN</td>
<td>VV</td>
</tr>
<tr>
<td></td>
<td>eine Auskunft geben</td>
<td></td>
</tr>
</tbody>
</table>
```

advice
Synchronous Grammars

Synchronous multi tree substitution grammar: \( N \rightarrow (r, \langle r_1, \ldots, r_n \rangle) \)


- nonterminal \( N \)
- right-hand side \( r \) of context-free grammar production
- right-hand sides \( r_1, \ldots, r_n \) of regular tree grammar production
Synchronous Grammars

Synchronous multi tree substitution grammar: \( N \rightarrow (r, \langle r_1, \ldots, r_n \rangle) \)

- nonterminal \( N \)
- right-hand side \( r \) of context-free grammar production
- right-hand sides \( r_1, \ldots, r_n \) of regular tree grammar production
- synchronization via map \( NT r_1, \ldots, r_n \) to \( NT r \)
Production application

1. synchronous nonterminals
Production application

1 synchronous nonterminals
Synchronous Grammars

Production application

1. synchronous nonterminals
2. suitable production
Production application

1. synchronous nonterminals
2. suitable production
3. replacement
PPER would like your advice about Rule 143

variant of [M.: How to train your multi bottom-up tree transducer. *Proc. ACL, 2011*]
PPER would like your advice about Rule 143.

variant of [M.: How to train your multi bottom-up tree transducer. *Proc. ACL, 2011*]
PPER would like your advice about Rule 143.

variant of [M.: How to train your multi bottom-up tree transducer. *Proc. ACL, 2011*]
PPER would like your advice about Rule 143

Könnten Sie eine Auskunft zu Artikel 143 geben

variant of [M.: How to train your multi bottom-up tree transducer. Proc. ACL, 2011]
Synchronous Multi Tree Substitution Grammars

Advantages

- complicated discontinuities
- implemented in framework ‘Moses’
- binarizable, composable
Synchronous Multi Tree Substitution Grammars

Advantages

- complicated discontinuities
- implemented in framework ‘Moses’
- binarizable, composable

Disadvantages

- output non-regular (trees) or non-context-free (strings)
- not symmetric (input context-free; output not)
He bought a new and fuel-efficient car.
## Evaluation

<table>
<thead>
<tr>
<th>System</th>
<th>Number of productions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E.-to-German</td>
</tr>
<tr>
<td>t-to-t STSG</td>
<td>7M</td>
</tr>
<tr>
<td>t-to-t SMTSG</td>
<td>41M</td>
</tr>
<tr>
<td>s-to-t STSG</td>
<td>14M</td>
</tr>
<tr>
<td>s-to-t SMTSG</td>
<td>144M</td>
</tr>
</tbody>
</table>

### Evaluation

**String-to-tree systems:**

<table>
<thead>
<tr>
<th>Task</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>STSG</td>
</tr>
<tr>
<td>English → German</td>
<td>15.0</td>
</tr>
<tr>
<td>English → Arabic</td>
<td>48.2</td>
</tr>
<tr>
<td>English → Chinese</td>
<td>17.7</td>
</tr>
<tr>
<td>English → Polish</td>
<td>21.3</td>
</tr>
<tr>
<td>English → Russian</td>
<td>24.7</td>
</tr>
</tbody>
</table>

Conclusions

- consistent improvements
- 1 magnitude more productions
- SMTSG alleviate some of the problems of syntax-based systems
Synchronous Grammars

Notes
- tree-to-tree models easier for theoretical investigation
- strongly related to tree transducers
- we disallow trivial input sides of just a nonterminal ($\varepsilon$-free)

Synchronous grammar:

Tree transducer:
Synchronous Grammars

Major linear tree transducers:

<table>
<thead>
<tr>
<th>synchronization</th>
<th>bijective</th>
<th>injective</th>
</tr>
</thead>
<tbody>
<tr>
<td>input sides</td>
<td></td>
<td>(output → input)</td>
</tr>
<tr>
<td>shallow</td>
<td>nondeleting top-down …</td>
<td>top-down …</td>
</tr>
<tr>
<td>general</td>
<td>nondeleting extended …</td>
<td>extended …</td>
</tr>
</tbody>
</table>

Further distinction

- allow productions on disconnected input nonterminals
  → regular look-ahead
- allow arbitrary trees for disconnected input nonterminals
  → no look-ahead
Synchronous Grammars

Illustration

- no look-ahead: can plug any (terminal) tree for \( N_{MD} \)
  [e.g., \( NP(DT(\text{the}), NN(\text{tower})) \)]
Synchronous Grammars

Illustration

- **no look-ahead**: can plug any (terminal) tree for $N_{MD}$ [e.g., $NP(DT(\text{the}), NN(\text{tower}))$]
- **regular look-ahead**: use special “no-output”-productions $N \rightarrow (r)$ [e.g., $N_{MD} \rightarrow (MD(\text{should}))$]
Synchronous Grammars

- **no look-ahead**: can plug any (terminal) tree for $N_{MD}$
  
  [e.g., $NP(DT(the), NN(tower))]$

- **regular look-ahead**: use special “no-output”-productions $N \rightarrow (r)$
  
  [e.g., $N_{MD} \rightarrow (MD(should))]$

- **SMTSG always have regular look-ahead**
  (any number of components includes 0)
Synchronous Grammars

Evaluation criteria

(rotations implementable?) (for arbitrary $t_1, t_2, t_3$)

symmetric?
domain regular?
range regular?
closed under composition?


Icons by interactivemania (http://www.interactivemania.com/) and UN Office for the Coordination of Humanitarian Affairs
Synchronous Grammars

Illustration of rotations

Tree Automata in Parsing and Machine Translation
Andreas Maletti · 47
Top-down Tree Transducer

Hasse diagram with composition closure indicated in subscript:

```
  TOP
  |   |
  TOP R1    s-TOP R1
  |   |
  TOP R2    s-TOP R2
  |   |
  s-TOP R2    n-TOP (R)
  |   |
  ns-TOP (R)    n-TOP (R)
```
### Top-down Tree Transducer

<table>
<thead>
<tr>
<th>Model \ Criterion</th>
<th>Rotation</th>
<th>Symmetry</th>
<th>Homogeneity</th>
<th>Composition</th>
<th>Synthesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>ns-TOP</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>n-TOP</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>s-TOP</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>s-TOP&lt;sup&gt;R&lt;/sup&gt;</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TOP</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TOP&lt;sup&gt;R&lt;/sup&gt;</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Synchronous Tree Substitution Grammars

Hasse diagram with the composition closure indicated in subscript:

composition closures by

Tree Automata in Parsing and Machine Translation
Synchronous Tree Substitution Grammars

<table>
<thead>
<tr>
<th>Model \ Criterion</th>
<th>⬡</th>
<th>🎨</th>
<th>🔧</th>
<th>🛠</th>
</tr>
</thead>
<tbody>
<tr>
<td>n-TOP</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TOP</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TOP^R</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ns-STSG</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>n-STSG</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>s-STSG^R</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>STSG</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>STSG^R</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Tree Automata in Parsing and Machine Translation

Andreas Maletti

51
Synchronous Multi Tree Substitution Grammars

**Advantages of SMTSG**

- always have regular look-ahead
- can always be made nondeleting & shallow
- closed under composition

Synchronous Multi Tree Substitution Grammars

Advantages of SMTSG

- always have regular look-ahead
- can always be made nondeleting & shallow
- closed under composition

Disadvantages of SMTSG

- non-regular range

Synchronous Multi Tree Substitution Grammars

Hasse diagram with the composition closure indicated in subscript:
## Synchronous Multi Tree Substitution Grammars

<table>
<thead>
<tr>
<th>Model \ Criterion</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>n-TOP</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TOP</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>x2</td>
</tr>
<tr>
<td>TOPR</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ns-STSG</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>x2</td>
</tr>
<tr>
<td>n-STSG</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>x∞</td>
</tr>
<tr>
<td>s-STSG(R)</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>x2</td>
</tr>
<tr>
<td>STSG</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>x4</td>
</tr>
<tr>
<td>STSGR</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>x3</td>
</tr>
<tr>
<td>(n)s-SMTSG(R)</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>(n)-SMTSG(R)</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>reg.-range SMTSG</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>symmetric SMTSG</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

(string-level) range characterization by

Theorem

\((\text{STSG}^R)^3 \subsetneq \text{reg.-range SMTSG}\)

abstracts a well-known linguistic transformation called *topicalization*

implementable by SMTSG, but not by any composition of STSG
Illustration of topicalization

- It rained *yesterday night*.
  
  **Topicalized**: *Yesterday night*, it rained.
Illustration of topicalization

- It rained *yesterday night*.
  
  *Topicalized:* *Yesterday night*, it rained.

- We toiled all day yesterday at the restaurant that charges extra for clean plates.
  
  *Topicalized:* At the restaurant that charges extra for clean plates, we toiled all day yesterday.
On the tree level

Synchronous Multi Tree Substitution Grammars
### Summary

**Parsing**
- weighted tree automata = CFG with subcategorization
- state-of-the-art models for many languages
- wealth of open problems for non-constituent parsing

**Machine translation**
- many grammar-based translation models in use
- expressive power ill-understood
- combination of parser grammar and translation model only understood for weighted tree automata
- “generalized” models might even work in practice
Thank you for the attention.