Binarization, Synchronous Binarization, and Target-side Binarization

Liang Huang

University of Pennsylvania

(part of this work was done at USC/ISI)

SSST workshop, NAACL 2006, Rochester, NY
Overview of Binarization

**Parsing**

CFG

\[
S \rightarrow \text{NP} \ \text{VP} \ \text{PP}
\]

**Machine Translation**

Synchronous CFG

\[
S \rightarrow \text{NP} \ \text{PP} \ \text{VP}
\]
Overview of Binarization

**parsing**

CFG

```
S → NP VP PP
```

for cubic time parsing

**machine translation**

Synchronous CFG

```
S → NP PP VP
```

for polynomial time decoding
Overview of Binarization

parsing

CFG

S → NP VP PP

for cubic time parsing

different schemes work equally well

machine translation

Synchronous CFG

S → NP PP VP

for polynomial time decoding

for cubic time parsing

different schemes work equally well
Overview of Binarization

- **Parsing**: CFG
  - $S \rightarrow NP \ VP \ PP$

- **Machine Translation**: Synchronous CFG
  - $S \rightarrow NP \ VP \ PP$

For cubic time parsing, different schemes work **equally well**. For polynomial time decoding, different schemes work **very differently**.
Overview of Binarization

machine translation

Synchronous CFG

\[ S \to \begin{array}{c} \text{NP} \quad \text{PP} \quad \text{VP} \\ \text{NP} \quad \text{VP} \quad \text{PP} \end{array} \]

for polynomial time decoding

different schemes work very differently
Overview of Binarization

(naïve) binarization

Synchronous CFG

\[ S \rightarrow NP \quad PP \quad VP \]

for polynomial time decoding

different schemes work very differently

machine translation
Overview of Binarization

Synchronous CFG

$S \rightarrow NP \ VP\ PP$

for polynomial time decoding

different schemes work very differently

machine translation
Overview of Binarization

synchronous binarization

(Zhang, Huang, Gildea, Knight, 2006)

is a principled scheme

machine translation

Synchronous CFG

\[
S \rightarrow \begin{array}{c}
\text{NP} \\
\text{PP} \\
\text{VP} \\
\text{PP-VP}
\end{array}
\]

for polynomial time decoding

different schemes work very differently
Overview of Binarization

synchronous binarization
(Zhang, Huang, Gildea, Knight, 2006)
is a principled scheme

machine translation

Synchronous CFG

for polynomial time decoding

different schemes work very differently
Overview of Binarization

Synchronous binarization (Zhang, Huang, Gildea, Knight, 2006) is a principled scheme for polynomial time decoding of synchronous CFGs. Different schemes work very differently.

For machine translation, the S → NP PP VP NP VP PP structure is used.
Overview of Binarization

synchronous binarization
(Zhang, Huang, Gildea, Knight, 2006)
is a principled scheme

machine translation
Synchronous CFG

S → NP PP VP NP VP PP

for polynomial time decoding

different schemes
work very differently

target-side binarization:
a simpler alternative
that works equally well on
tree-to-string systems
In this talk ...

- Background: SCFGs and their applications in MT
  - string-input and tree-input systems
- The Three Binarization Schemes
  - source-side, synchronous, and target-side binarizations
- Theoretical Analysis
  - translation as parsing
  - decoding with an integrated language model
- Experiments on Tree-to-String Systems
Background: SCFGs

- Synchronous Context-Free Grammar (SCFG)
- CFG in two dimensions, generating pairs of trees/strings
- co-indexed nonterminal further rewritten as a unit

\[
S \rightarrow NP^{(1)} \ PP^{(2)} \ VP^{(3)} , \quad NP^{(1)} \ VP^{(3)} \ PP^{(2)}
\]

\[
NP \rightarrow \text{Baoweier,}
\]

\[
PP \rightarrow \text{yu Shalong,}
\]

\[
VP \rightarrow \text{juxing le huitan,}
\]

\[
S \rightarrow NP \ PP \ VP
\]

\[
NP \rightarrow \text{Baoweier,}
\]

\[
PP \rightarrow \text{yu Shalong,}
\]

\[
VP \rightarrow \text{juxing le huitan,}
\]

\[
S \rightarrow NP \ VP \ PP
\]

\[
NP \rightarrow \text{Baoweier,}
\]

\[
PP \rightarrow \text{juxing le huitan,}
\]

\[
VP \rightarrow \text{held a meeting,}
\]

\[
S \rightarrow NP \ PP \ VP
\]

\[
NP \rightarrow \text{yu Shalong,}
\]

\[
VP \rightarrow \text{held a meeting,}
\]

\[
S \rightarrow NP \ VP \ PP
\]

Two Uses of SCFGs

string-input \quad (Wu, 1997; Chiang, 2005; Galley et al., 2006)

\underline{Baoweier \ yu \ Shalong \ juxing \ le \ huitan}

\underline{tree-input} \quad (Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)
Two Uses of SCFGs

string-input

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

Baoweier yu Shalong juxing le huitan

NP

NP


Powell

tree-input

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)
Two Uses of SCFGs

**string-input**
(Wu, 1997; Chiang, 2005; Galley et al., 2006)

\[
\begin{array}{ll}
\text{NP} & \text{PP} \\
\text{Baoweier} & \text{yu Shalong juxing le huitan} \\
\end{array}
\]

\[
\begin{array}{ll}
\text{NP} & \text{PP} \\
\text{Powell} & \text{with Sharon} \\
\end{array}
\]

**tree-input**
(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)
Two Uses of SCFGs

(string-input) (Wu, 1997; Chiang, 2005; Galley et al., 2006)

NP \hspace{1cm} PP \hspace{1cm} VP

Baoweier yu Shalong juxing le huitan

(np) Powell with Sharon held a meeting

(tree-input) (Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)
Two Uses of SCFGs

(string-input) 

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

NP
Baoweier

PP
yu Shalong

VP
juxing le huitan

(tree-input) 

(NP)
Powell

(PP)
with Sharon

(VP)
held a meeting

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)
Two Uses of SCFGs

string-input

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

Baoweier yu Shalong juxing le huitan

NP PP VP

tree-input

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

Powell held a meeting with Sharon

NP VP PP
Two Uses of SCFGs

string-input

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

```
S
NP  PP  VP

Baoweier  yu  Shalong  juxing  le  huitan
```

```
S
NP  VP  PP

NP  VP  PP

Powell  held a meeting  with  Sharon
```

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

tree-input

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

“syntax-directed”
Two Uses of SCFGs

string-input

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

NP  PP  VP
Baoweier  yu Shalong  juxing le huitan

(S)

tree-input

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

“syntax-directed”

NP  PP  VP
Baoweier  yu Shalong  juxing le huitan

(S)

Huang

Target-side Binarization
Two Uses of SCFGs

**string-input**

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

\[ S \]

```
NP  PP  VP
Baoweier  yu Shalong  juxing le huitan
```

**tree-input**

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

```
S
```

```
NP  PP  VP
Baoweier  yu Shalong  juxing le huitan
```

```
VP
Powell held a meeting
```

```
PP
with Sharon
```

"syntax-directed"
Two Uses of SCFGs

**string-input**

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

```
S

NP  PP  VP
Baoweier  yu Shalong  juxing le huitan
```

---

**tree-input**

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

“syntax-directed”

```
S

NP  PP  VP
Baoweier  yu Shalong  juxing le huitan
```

---

Target-side Binarization  

Huang
Two Uses of SCFGs

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

“syntax-directed”

string-input

Baoweier yu Shalong juxing le huitan

NP PP VP

S

tree-input

Baoweier yu Shalong juxing le huitan

NP PP VP

S

Powell held a meeting with Sharon

NP VP PP

S

Powell

“syntax-directed”

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)
Two Uses of SCFGs

string-input  
(Wu, 1997; Chiang, 2005; Galley et al., 2006)

```
S
  NP
  PP
  VP
Baoweier  yu Shalong  juxing le huitan
```

---

tree-input  
(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)  
“syntax-directed”

```
S
  NP
  PP
  VP
Baoweier  yu Shalong  juxing le huitan
```

```
S
  NP
  VP
  PP
Powell  held a meeting  with Sharon
```
Two Uses of SCFGs

**string-input**

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

\[ S \]

- **NP**
  - Baoweier

- **PP**
  - yu Shalong

- **VP**
  - juxing le huitan

**tree-input**

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

“syntax-directed”

\[ S \]

- **NP**
  - Powell

- **VP**
  - held a meeting

- **PP**
  - with Sharon

---

Huang

Target-side Binarization 6
Two Uses of SCFGs

(string-input) (Wu, 1997; Chiang, 2005; Galley et al., 2006)

Baoweier yu Shalong juxing le huitan

(tree-input) (Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006) “syntax-directed”

Baoweier yu Shalong juxing le huitan

Powell held a meeting with Sharon
Two Uses of SCFGs

String-input

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

Baoweier yu Shalong juxing le huitan

Powell held a meeting w/ Sharon

Tree-input

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006) “syntax-directed”

Baoweier yu Shalong juxing le huitan

Powell held a meeting with Sharon

Huang

Target-side Binarization
Two Uses of SCFGs

**string-input**

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

Baoweier yu Shalong juxing le huitan

Powell held a meeting w/ Sharon

**tree-input**

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

“syntax-directed”

Huang

Target-side Binarization 7
Two Uses of SCFGs

string-input

(Wu, 1997; Chiang, 2005; Galley et al., 2006)

Baoweier yu Shalong juxing le huitan

Powell held a meeting w/ Sharon

(Liu et al., 2006; Huang et al., 2006; Cowan et al., 2006)

“syntax-directed”

larger locality: STSG, STAG, ...

details later...

Huang
Three Binarization Schemes

source-side

synchronous

target-side

NP-PP

PP-VP

NP-VP

NP-PP

PP

NP

NP-VP

VP

VP

VP

NP

gap

gap

gap
Three Binarization Schemes

- **source-side**
  - NP-PP
  - NP-PP
  - contiguous on: source only
  - applicability: 100%
  - implementation: trivial

- **synchronous**
  - PP-VP
  - PP-VP
  - contiguous on: both sides
  - applicability: high, but not always
  - implementation: involved

- **target-side**
  - PP
  - NP-VP
  - contiguous on: target only
  - applicability: 100%
  - implementation: trivial
Three Binarization Schemes

- **source-side**
- **synchronous**
- **target-side**

**contiguous on:**
- source only
- both sides
- target only

**applicability:**
- 100%
- high, but not always
- 100%

**implementation:**
- trivial
- involved
- trivial

**string-input:**
- very bad
- good
- bad

**tree-input:**
- very bad
- good
- very good

Huang
Source-side Binarization

- translation model (TM) only
- same as monolingual parsing
- cubic on the source side
- with a language model \((\text{Wu}, 1996)\)
- exponential due to gaps \((\text{Huang et al.}, 2005)\)

\(O(|w|^{3+2(t+1)(m−1)})\)

\textit{m-gram, target discontinuity \(t\)}

Baoweier yu Shalong juxing le huitan
Source-side Binarization

- translation model (TM) only
- same as monolingual parsing
- cubic on the source side
- with a language model \((\text{Wu, 1996})\)
- exponential due to gaps \((\text{Huang et al., 2005})\)

\[
O(|w|^{3+2(t+1)(m-1)})
\]

\(m\)-gram, target discontinuity \(t\)
**Source-side Binarization**

- translation model (TM) only
- same as monolingual parsing
- cubic on the source side
- with a language model \((Wu, 1996)\)
- exponential due to gaps \((Huang et al., 2005)\)

\[
O(|w|^{3+2(t+1)(m-1)})
\]

- \(m\)-gram, target discontinuity \(t\)

**Target-side Binarization**
Synchronous Binarization

- contiguous on both sides
- cubic on the source side
- polynomial on the target side
- but ...

\[ O(|w|^{3+4(m-1)}) \]

Baoweier yu Shalong juxing le huitan
Binarizability

- not all rules are synchronously binarizable
- Chinese-English: vast majority is binarizable
- but binarizability ratio decreases on freer word-order languages (Wellington et al., 2006)

“will meet on the same day with Mishira, secretary...”

(Huang, Zhang, Gildea, Knight, 2007)
Target-side Binarization

- contiguous on target-side
- polynomial for LM
- but gaps on the source-side!
- can not use CKY
- exponential for discont. parsing (cf. TAGs)

- trade-off is worthwhile
- because LM complexity is higher

- related: (Watanabe et al., 2006)
  (Venugopal et al., 2007)

\[ O(|w|(2s+1)+4(m−1)) \]
## Summary: string-input systems

$m$-gram, source discontinuity $s$, target discontinuity $t$

<table>
<thead>
<tr>
<th></th>
<th>TM</th>
<th>LM</th>
<th>overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>source-side</td>
<td>3</td>
<td>$2(t+1)(m-1)$</td>
<td>$3+2(t+1)(m-1)$</td>
</tr>
<tr>
<td>synchronous</td>
<td>3</td>
<td>$4(m-1)$</td>
<td>$3+4(m-1)$</td>
</tr>
<tr>
<td>target-side</td>
<td>$2s+1$</td>
<td>$4(m-1)$</td>
<td>$(2s+1)+4(m-1)$</td>
</tr>
</tbody>
</table>
Tree-input Systems

• pattern-match the input tree with the source-proj.
• key difference
  • TM-only decoding does not need binarization!
  • always linear-time decoding regardless of arity
• source-side binarization: still exponential time
Tree-input Systems

- pattern-match the input tree with the source-proj.
- **key difference**
  - TM-only decoding does **not** need binarization!
  - always **linear-time** decoding regardless of arity
- source-side binarization: still exponential time

```
S
  NP
  PP
  VP

Baoweier yu Shalong juxing le huitan
```

Huang
Tree-input Systems

- pattern-match the input tree with the source-proj.
- **key difference**
  - TM-only decoding does **not** need binarization!
  - always **linear-time** decoding regardless of arity
- source-side binarization: still exponential time
Tree-input Systems

- pattern-match the input tree with the source-proj.
- key difference
  - TM-only decoding does not need binarization!
  - always linear-time decoding regardless of arity
- source-side binarization: still exponential time
Tree-input Systems

- pattern-match the input tree with the source-proj.
- **key difference**
  - TM-only decoding does **not** need binarization!
  - always **linear-time** decoding regardless of arity
- source-side binarization: still exponential time

Baoweier [yu Shalong] juxing le huitan

Powell
with Sharon

Huang
(on-the-fly) left-to-right generation on the target-side

also memoize virtual non-terminals

=> target-side binarization for tree-input systems

now it does not suffer from discontiguous parsing, and still enjoys contiguous generation
• (on-the-fly) left-to-right generation on the target-side
• also memoize virtual non-terminals
  • => target-side binarization for tree-input systems
• now it does not suffer from discontiguous parsing, and still enjoys contiguous generation
(on-the-fly) left-to-right generation on the target-side
also memoize virtual non-terminals
=> target-side binarization for tree-input systems
now it does not suffer from discontiguous parsing, and still enjoys contiguous generation
• (on-the-fly) left-to-right generation on the target-side
• also memoize virtual non-terminals
  • => target-side binarization for tree-input systems
• now it does not suffer from discontiguous parsing, and still enjoys contiguous generation
(on-the-fly) left-to-right generation on the target-side

Also memoize virtual non-terminals

$\Rightarrow$ target-side binarization for tree-input systems

Now it does not suffer from discontiguous parsing, and still enjoys contiguous generation
(on-the-fly) left-to-right generation on the target-side

also memoize virtual non-terminals

=> target-side binarization for tree-input systems

now it does not suffer from discontiguous parsing, and still enjoys contiguous generation
Target-side vs. Synchronous

- synchronous binarization: unnecessarily complicated
Target-side vs. Synchronous

- synchronous binarization: unnecessarily complicated
Target-side vs. Synchronous

- synchronous binarization: unnecessarily complicated

Huang
• synchronous binarization: unnecessarily complicated
Target-side vs. Synchronous

- synchronous binarization: unnecessarily complicated
### Complexity Summary

$m$-gram, source discontinuity $s$, target discontinuity $t$

<table>
<thead>
<tr>
<th></th>
<th>string-input</th>
<th>tree-input *</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>source-side</strong></td>
<td>$3 + 2(t+1)(m-1)$</td>
<td>$1 + 2(t+1)(m-1)$</td>
</tr>
<tr>
<td><strong>synchronous</strong></td>
<td>$3 + 4(m-1)$</td>
<td>$1 + 4(m-1)$</td>
</tr>
<tr>
<td><strong>target-side</strong></td>
<td>$2s + 1 + 4(m-1)$</td>
<td>$1 + 4(m-1)$</td>
</tr>
</tbody>
</table>

* does not include parsing time
### Complexity Summary

$m$-gram, source discontinuity $s$, target discontinuity $t$

<table>
<thead>
<tr>
<th></th>
<th>string-input</th>
<th>tree-input *</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>source-side</strong></td>
<td>☹</td>
<td>☹</td>
</tr>
<tr>
<td><strong>synchronous</strong></td>
<td>☻</td>
<td>☻</td>
</tr>
<tr>
<td><strong>target-side</strong></td>
<td>☹</td>
<td>☻</td>
</tr>
</tbody>
</table>
Experiments
Tree-to-String System

- syntax-directed English to Chinese \((\text{Huang, Knight, Joshi, 2006})\)
- the reverse direction is found in \((\text{Liu et al., 2006})\)

Extended domain of locality:
synchronous tree-substitution grammars (STSG)
\((\text{Galley et al., 2004; Eisner, 2003})\)

Related to
STAG \((\text{Shieber/Schabes, 90})\)
STIG \((\text{Nesson et al., 06})\)
Huang

Target-side Binarization

21

TM-only Decoding

- depth-first-search (DFS)
- for each tree node
- try all rules applicable
- recursion on subtrees
- top-down memoization
- linear-time
- dynamic programming
- soft decisions
Data

- trained on a parallel corpus of 28.3 M words on the English-side
- tested on 140 sentences (9-36 words)
- English-side parsed by a variant of Collins parser
- GIZA++ alignment
- extracted 24.7 M rules using (Galley et al., 2004)
- slightly better than Pharaoh (Koehn, 2004) in BLEU
- saved forests for these 140 sentences
  - non-binarizable ratio: 0.25%   cf. 0.3% in (Zhang et al., 06)
Forest Size

- Original forest
- Target-side binarization
- On-the-fly generation

Number of nodes in the forest vs. length of the input sentence.
Decoding

The graph shows the relationship between beam size and various metrics. The x-axis represents the beam size, while the y-axis represents the average number of LM items per sentence. The legend includes lines for BLEU score, on-the-fly generation, and target-side binarization. The graph demonstrates an increasing trend for all metrics as the beam size increases.
Decoding

![Graph showing BLEU score and average number of +LM items per sentence against beam size. The graph includes two lines representing BLEU score and average number of +LM items per sentence. The BLEU score line is red and the average number of +LM items per sentence line is green. There are two markers indicating on-the-fly generation and target-side binarization. The x-axis represents beam size ranging from 5 to 20, and the y-axis represents BLEU score from 25 to 26.2 and average number of +LM items per sentence from 0 to 120,000.](image)
Decoding

Pharaoh: 23.54

Huang

Target-side Binarization
Decoding

ACL paper: faster algorithms for integrated decoding

Pharaoh: 23.54

Huang
Conclusion

- a simpler alternative binarization scheme
- theoretical comparison of the three binarization schemes in two popular types of systems
  - string-input: synchronous; tree-input: target-side
- empirical evidence on a tree-to-string system
- tree-input ("syntax-directed") is a promising direction
  - linear-time decoding
  - easy binarization
  - decoupled parsing and transduction grammars
Thanks!