Efficient Incremental Decoding and Rule Markov Model for Tree-to-String Translation

Haitao Mi

Institute of Computing Technology
Chinese Academy of Sciences

Information Sciences Institute
University of Southern California

with Liang Huang, Ashish Vaswani and David Chiang
### MT: Phrase-based vs. Syntax-based

<table>
<thead>
<tr>
<th></th>
<th>phrase</th>
<th>syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>hypotheses generation</td>
<td>left-to-right</td>
<td>bottom-up</td>
</tr>
<tr>
<td>in theory</td>
<td>exponential</td>
<td>polynomial</td>
</tr>
<tr>
<td>in practice</td>
<td>fast (linear with beam search)</td>
<td>slow (with LM and huge rule set)</td>
</tr>
<tr>
<td></td>
<td>phrase</td>
<td>syntax</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>hypotheses generation</td>
<td>left-to-right</td>
<td>bottom-up</td>
</tr>
<tr>
<td>in theory</td>
<td>exponential</td>
<td>polynomial</td>
</tr>
<tr>
<td>in practice</td>
<td>fast (linear with beam search)</td>
<td>slow (with LM and huge rule set)</td>
</tr>
</tbody>
</table>

Q1: borrow phrase-based decoding for tree-to-string?
## MT: Phrase-based vs. Syntax-based

<table>
<thead>
<tr>
<th></th>
<th>Phrase</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>hypotheses generation</strong></td>
<td><strong>left-to-right</strong></td>
<td>left-to-right</td>
</tr>
<tr>
<td><strong>in theory</strong></td>
<td><strong>exponential</strong></td>
<td><strong>polynomial</strong></td>
</tr>
<tr>
<td><strong>in practice</strong></td>
<td><strong>fast</strong> (linear with beam search)</td>
<td><strong>fast</strong> (with LM and huge rule set)</td>
</tr>
</tbody>
</table>

Q1: borrow phrase-based decoding for tree-to-string?
## MT: Phrase-based vs. Syntax-based

<table>
<thead>
<tr>
<th></th>
<th>phrase</th>
<th>syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>hypotheses generation</td>
<td>left-to-right</td>
<td>left-to-right</td>
</tr>
<tr>
<td>in theory</td>
<td>exponential</td>
<td>polynomial</td>
</tr>
<tr>
<td>in practice</td>
<td>fast (linear with beam search)</td>
<td>fast (with LM and huge rule set)</td>
</tr>
</tbody>
</table>

Q1: borrow phrase-based decoding for tree-to-string?
Q2: use a **small rule set, faster, a very slimmer model**?
### MT: Phrase-based vs. Syntax-based

<table>
<thead>
<tr>
<th></th>
<th>phrase</th>
<th>syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>hypotheses generation</td>
<td>left-to-right</td>
<td>left-to-right</td>
</tr>
<tr>
<td>in theory</td>
<td>exponential</td>
<td>polynomial</td>
</tr>
<tr>
<td>in practice</td>
<td>fast</td>
<td>faster</td>
</tr>
<tr>
<td></td>
<td>(linear with beam search)</td>
<td>(with LM and small rule set)</td>
</tr>
</tbody>
</table>

Q1: borrow phrase-based decoding for tree-to-string?

Q2: use a **small rule set, faster**, a very slimmer model?
Outline

• Background
  • Tree-to-String Translation
  • Phrase-based Decoding
• (Q1) Incremental Decoding for Tree-to-String Trans.
• (Q2) Rule Markov Model for Tree-to-String Trans.
• Conclusion
Tree-to-String Translation

- parse; convert to English by pattern-matching tree-to-string rules

_Bùshí yǔ Shālóng jǔxíng le huìtán_

(Galley et al., 2004; Liu et al., 2006; Huang et al., 2006)
Tree-to-String Translation

- parse; convert to English by pattern-matching tree-to-string rules

(Galley et al., 2004; Liu et al., 2006; Huang et al., 2006)
Tree-to-String Translation

- parse; convert to English by pattern-matching tree-to-string rules

\[ \text{IP}(x_1: \text{NPB}, x_2: \text{VP}) \rightarrow x_1 \ x_2 \]

(Galley et al., 2004; Liu et al., 2006; Huang et al., 2006)
Tree-to-String Translation

- recursively solve unfinished subproblems

(Liu et al. 06; Huang et al. 06)
Tree-to-String Translation

- recursively solve unfinished subproblems

(Liu et al 06; Huang et al 06)
Tree-to-String Translation

- pattern-match tree-to-string translation rules

Bush

(Liu et al. 06; Huang et al. 06)
Tree-to-String Translation

• pattern-match tree-to-string translation rules

Bush

(Liu et al 06; Huang et al 06)
Tree-to-String Translation

- pattern-match tree-to-string translation rules

Bush

(Liu et al 06; Huang et al 06)
Tree-to-String Translation

- continue pattern-matching

Bush held with (Liu et al 06; Huang et al 06)
Tree-to-String Translation

- continue pattern-matching

Bush held with

(NPB | NN | huìtán) talk

(NPB | NR | Shālóng) Sharon

(Liu et al 06; Huang et al 06)
Tree-to-String Translation

• continue pattern-matching

Bush held a talk with Sharon

(Galley et al 04; Liu et al 06; Huang et al 06)
Tree-to-String Translation

- continue pattern-matching

Bush held a talk with Sharon

really simple! and runs fast in $O(n)$-time!
Tree-to-String Translation

- continue pattern-matching

Bush held a talk with Sharon

really simple! and runs fast in $O(n)$-time!

but it becomes slower, when

integrating with language model (Q1)
and large rule set (Q2)

(Galley et al 04; Liu et al 06; Huang et al 06)
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up

```
VP
  PP  VPB
   P   VV   AS   NPB
    yǔ  NR  jǔxíng  le  NN
        Shālóng  huìtán
```
Why Decoding w/ LM is Slow?

• reordering of sub hypotheses in bottom-up
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up

```
VP
  └── VPB
    ├── PP
    └── VPB
      ├── P
      │   └── yǔ
      │       └── Shālóng
      └── NPB
          └── NR
              └── jǔxíng
                  └── le
                      └── NN
                          └── huìtán
      └── VV
      └── AS
      └── NPB

VP(x₁:PP x₂:VPB) → x₁ x₂
```
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up

```
VP
  PP  VPB
  P   NPB  VV  AS  NPB
  yǔ NR jǔxíng le NN
  Shālóng huìtán

VP(x₁:PP x₂:VPB) → x₁ x₂
with ... talks
```

```
PP
with ... Sharon
and ... Sharon
with ... Shalong

VPB
held ... talks
held ... meeting
hold ... talk
```
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up
- each node is now split into several +LM nodes
- maintain LM signatures at both ends

\[
VP(x_1:PP \ x_2:VPB) \rightarrow x_2 \ x_1
\]

- held ... Sharon
- hold ... Shalong
- held ... talks
- held ... meeting
- hold ... talk
- with ... Sharon
- and ... Sharon
- with ... Shalong
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up
- each node is now split into several +LM nodes
- maintain LM signatures at both ends
- cross-product

\[
\text{VP}(x_1:\text{PP} \, x_2:\text{VPB}) \rightarrow x_2 \, x_1
\]

- held ... Sharon
- hold ... Shalong
- held ... talks
- held ... meeting
- hold ... talk
- with ... Sharon
- and ... Sharon
- with ... Shalong
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up
- each node is now split into several +LM nodes
- maintain LM signatures at both ends
- cross-product

\[
\text{time: } O(n V^4(m-1)) \text{ for 2-gram LM}
\]
Why Decoding w/ LM is Slow?

- reordering of sub hypotheses in bottom-up
- each node is now split into several +LM nodes
- maintain LM signatures at both ends
- cross-product

**Diagram:****

\[ VP(x_1:PP \cdot x_2:VPB) \rightarrow x_2 x_1 \]

- time: \( O(n V^4(m-1)) \) for \( m \)-gram LM

**Words:**
- held ... Sharon
- hold ... Shalong
- held ... talks
- held ... meeting
- hold ... talk
- with ... Sharon
- and ... Sharon
- with ... Shalong
Why Phrase-based is Fast?

Yu Shalong held a talk with Sharon.

- **Source-side:** Coverage vector
  - held a talk

- **Target-side:** Grow strictly left-to-right
  - held a talk

- **Hypothesis**
Why Phrase-based is Fast?

与 沙龙 举行了 会谈
yu Shalong juxing le huitan

held a talk with Sharon

hypothesis
source-side: coverage vector

held a talk

target-side: grow strictly left-to-right
Why Phrase-based is Fast?

yu Shalong held a talk with Sharon

source-side: coverage vector

held a talk

target-side: grow strictly left-to-right
Why Phrase-based is Fast?

yu Shalong juxing le huitan
held a talk with Sharon

source-side: coverage vector

hypothesis

held a talk

target-side: grow strictly left-to-right

held a talk with Sharon
Why Phrase-based is Fast?

yu Shalong juxing le huitan

held a talk with Sharon

hypothesis
source-side: coverage vector

target-side: grow strictly left-to-right

held a talk

held a talk with Sharon

time complexity: $O(2^n n^2)$
Why Decoding w/ LM is Fast?

- grow hypotheses strictly in left-to-right order
- only maintain LM signatures at rightmost
Why Decoding w/ LM is Fast?

- grow hypotheses **strictly in left-to-right** order
- only maintain LM signatures at rightmost

```
 Why Decoding w/ LM is Fast?  

- grow hypotheses **strictly in left-to-right** order  
- only maintain LM signatures at rightmost  
```

...
Why Decoding w/ LM is Fast?

- grow hypotheses **strictly in left-to-right** order
- only maintain LM signatures at rightmost

\[
time: O(2^n n^2 V^{m-1})
\]
Why Decoding w/ LM is Fast?

- grow hypotheses strictly in left-to-right order
- only maintain LM signatures at rightmost

\[ \text{time: } O(2^n n^2 V^{m-1}) \]

\[ \text{time: } O(n V^{4(m-1)}) \]
Why Decoding w/ LM is Fast?

- grow hypotheses **strictly in left-to-right** order
- only maintain LM signatures at rightmost

\[
time: O(2^n n^2 V^{m-1})
\]

Q1: can tree-to-string also become incremental?
Outline

• Background
  • Tree-to-String Translation
  • Phrase-based Decoding
• (Q1) Incremental Decoding for Tree-to-String Trans.
• (Q2) Rule Markov Model for Tree-to-String Trans.
• Conclusion
Incremental for Tree-to-String

- key intuition: adapt coverage-vector idea
- tree coverage-vector: which sub-trees translated
Incremental for Tree-to-String

- key intuition: adapt coverage-vector idea
- tree coverage-vector: which sub-trees translated

not that easy!

expand arbitrary
Incremental for Tree-to-String

- key intuition: adapt coverage-vector idea
- tree coverage-vector: which sub-trees translated
  not that easy!

expand arbitrary  

already know which to expand
Incremental for Tree-to-String

• key intuition: adapt coverage-vector idea

• tree coverage-vector: which sub-trees translated
  not that easy!

expand arbitrary already know which to expand

derivation history
Tree Coverage Vector as Stack
Tree Coverage Vector as Stack

- stack \textit{(active derivation history)}: \([\varepsilon \rightarrow \cdot \text{IP}] [\text{IP} \rightarrow \text{NPB} \cdot \text{VP}]\)
Tree Coverage Vector as Stack

- stack (active derivation history): \([\varepsilon \rightarrow \cdot \text{IP}] [\text{IP} \rightarrow \text{NPB} \cdot \text{VP}]\)
- expand point (blue point at the top of stack)
Tree Coverage Vector as Stack

- **stack** (*active* derivation history):  \([\varepsilon \rightarrow \cdot IP][IP \rightarrow NPB, VP]\)
- **expand point** (blue point at the top of stack)
- **three colors for nodes:**

![Diagram of a tree structure with labeled nodes and edges]

- **IP**
  - **NPB**
    - **NR**
      - **Bùshí**
    - **VP**
      - **PP**
        - **P**
          - **yǔ**
        - **NPB**
          - **NR**
          - **jǔxíng**
        - **VV**
        - **AS**
          - **le**
        - **NPB**
          - **NN**
            - **huìtán**
  - **VPB**
Tree Coverage Vector as Stack

- **stack** (*active* derivation history): \[\varepsilon \rightarrow \cdot \text{IP} \] \[\text{IP} \rightarrow \text{NPB} \cdot \text{VP}\]
- **expand point** (blue point at the top of stack)
- **three colors for nodes:**
  - white (uncovered), grey (partially covered), and black (covered)

“\(\varepsilon\) I have finished NPB subtree but not started with VP subtree”

```
IP
  └── NPB
      └── VP
          └── P
              └── NPB
                  └── PP
                      └── yǔ
                          └── Shālóng

          └── VV
              └── jǔxíng

          └── AS
              └── le

          └── NPB
              └── NN
                  └── huìtán
```
Example Incremental Decoding

[ε→<s> IP </s>]

<s>

<s> IP </s>

IP

NPB  VP

NR  PP  VPB

P  NPB  VV  AS  NPB

yǔ  NR  jǔxíng  le  NN

Bushí  Shālóng  huìtán
Example Incremental Decoding

\[ \varepsilon \xrightarrow{\langle s \rangle} \cdot \text{IP} \xrightarrow{\langle /s \rangle} \] \[ \text{[IP} \xrightarrow{\cdot \text{NPB VP}] \]

\[ \langle s \rangle \]

\[ \text{IP}(x_1: \text{NPB} \ x_2: \text{VP}) \rightarrow x_1 \ x_2 \]

action: predict (push)
Example Incremental Decoding

\[ \varepsilon \rightarrow <s> \text{ IP } \langle/s\rangle ] \ [\text{IP} \rightarrow \text{NPB VP}] \ [\text{NPB} \rightarrow \text{Bush}] \]

<s>

stack

hypothesis

action: predict (push)
Example Incremental Decoding

\[
\begin{align*}
&[\varepsilon \rightarrow \text{IP} \langle \text{s} \rangle ] \ [\text{IP} \rightarrow \text{NPB} \ \text{VP}] \ [\text{NPB} \rightarrow \text{Bush} \langle \text{s} \rangle ] \\
&\langle \text{s} \rangle \ \text{Bush}
\end{align*}
\]
Example Incremental Decoding

\[
[\epsilon \rightarrow <s> \ IP \ </s>] \ [IP \rightarrow \ NPB \ VP] \\
\]

\(<s>\) Bush

\[
[\epsilon \rightarrow <s> \ IP \ </s>] \ [IP \rightarrow \ NPB \ VP] \\
\]

\(<s>\) IP \ </s>

\[
<s> \ IP \ </s> \\
\]

action: complete (pop)

stack

hypothesis
Example Incremental Decoding

\[
[ \varepsilon \rightarrow <s> \cdot \mathrm{IP} <\!/s> ] \ [ \mathrm{IP} \rightarrow \mathrm{NPB} \cdot \mathrm{VP} ]
\]

<s> Bush

\[
\begin{array}{ll}
\text{PP} & \text{VPB} \\
\text{P} & \text{NPB} \\
\text{yǔ} & \text{NR} \\
\text{Shālóng} & \\
\text{VPB} & \\
\text{VV} & \text{AS} \\
\text{jǔxíng} & \text{le} \\
\text{huìtán} & \\
\end{array}
\]

stack

hypothesis

action: complete (pop)
Example Incremental Decoding

[ε→<s> IP </s>] [IP→ NPB•VP] [VP → •held NPB with NPB]

<s> Bush

\[ \text{action: predict (push)} \]
Example Incremental Decoding

\[
[\varepsilon \rightarrow \langle s \rangle \text{ IP } \langle /s \rangle ] \quad [\text{IP} \rightarrow \text{NPB} \cdot \text{VP}] \quad [\text{VP} \rightarrow \text{held} \cdot \text{NPB} \text{ with } \text{NPB}]
\]

\langle s \rangle \text{ Bush held}

action: scan
Example Incremental Decoding

\[ \varepsilon \rightarrow <s> \text{IP } <s> \] \[ \text{IP} \rightarrow \text{NPB} \cdot \text{VP} \] \[ \text{VP} \rightarrow \text{held} \cdot \text{NPB} \text{ with NPB} \] \[ \text{NPB} \rightarrow \text{talks} \]

\(<s>\) Bush held

\(<s>\) IP <s>

\text{NPB}
\text{VP}
\text{PP}
\text{VPB}

\text{Bush}
\text{held}
\text{NPB}
\text{with}
\text{NPB}

\text{action: predict (push)}
Example Incremental Decoding

\[
[\varepsilon \rightarrow \langle s \rangle \text{ IP } \langle /s \rangle ]
[\text{IP} \rightarrow \text{NPB} \cdot \text{VP}]
[\text{VP} \rightarrow \text{held} \cdot \text{NPB} \text{ with NPB}]
[\text{NPB} \rightarrow \text{talks} ]
\]

\(<s>\) Bush held talks

action: scan
Example Incremental Decoding

\[ \varepsilon \rightarrow \langle s \rangle \quad \text{IP} \langle /s \rangle \quad [\text{IP} \rightarrow \text{NPB} \cdot \text{VP}] \quad [\text{VP} \rightarrow \text{held} \quad \text{NPB} \cdot \text{with} \quad \text{NPB}] \]

\langle s \rangle \quad \text{Bush held talks}

\textit{action: complete (pop)}
Example Incremental Decoding

\[ \varepsilon \rightarrow <s> \cdot \text{IP} <s> ] \ [ \text{IP} \rightarrow \text{NPB} \cdot \text{VP} ] \ [ \text{VP} \rightarrow \text{held} \text{NPB} \text{ with } \text{NPB} ] \]

<s> Bush held talks with

\textbf{action: scan}
Example Incremental Decoding

[ε→<s> ·IP </s>] [IP→ NPB•VP] [VP → held NPB with • NPB][NPB→•Sharon]

<s> Bush held talks with

action: predict (push)
Example Incremental Decoding

\[ \varepsilon \rightarrow <s> \ IP \ <s> \] \ [IP \rightarrow NPB \cdot VP] \ [VP \rightarrow \text{held NPB with NPB}] \ [NPB \rightarrow \text{Sharon}] \]

<s> Bush held talks with Sharon

action: scan
Example Incremental Decoding

\[
\begin{align*}
\varepsilon \rightarrow <s> & \text{ IP } \langle/s\rangle \ [\text{IP} \rightarrow \text{NPB} \bullet \text{VP}] \\
& [\text{VP} \rightarrow \text{held NPB with NPB} \bullet ]
\end{align*}
\]

\(<s>\) Bush held talks with Sharon

\(\text{action: complete (pop)}\)
Example Incremental Decoding

\[
[\varepsilon \rightarrow \text{IP} \leftarrow \text{IP} \rightarrow \text{NPB} \leftarrow \text{VP} \leftarrow \text{s}]
\]

\text{<s> Bush held talks with Sharon}

action: complete (pop)
Example Incremental Decoding

\[ \varepsilon \rightarrow <s> \text{ IP . } </s> \]

\(<s>\text{ Bush held talks with Sharon}\)

action: complete (pop)
Example Incremental Decoding

\[ \varepsilon \rightarrow <s> \text{ IP } </s> \cdot \]

\(<s>\) Bush held talks with Sharon <</s>\)

**action:** scan
Example Incremental Decoding

[ε→<s> IP </s>•]

<s> Bush held talks with Sharon </s>
Beam Search in Practice

- very similar to phrase-based beam search
- coverage-vectors => derivation stacks
Beam Search in Practice

- very similar to phrase-based beam search
- coverage-vectors => derivation stacks
- beaming: # of Chinese tree nodes in black or grey
Beam Search in Practice

- very similar to phrase-based beam search
- coverage-vectors => derivation stacks
- beaming: # of Chinese tree nodes in black or grey

Time: $O(nbc)$  \[ c: \# \text{ of edges at each node} \]
Beam Search in Practice

- very similar to phrase-based beam search
- coverage-vectors => derivation stacks
- beaming: # of Chinese tree nodes in black or grey

Time: $O(nbc)$  \( c \): # of edges at each node

linear-time if $c$ is a constant!
Beam Search in Practice

- very similar to phrase-based beam search
- coverage-vectors => derivation stacks
- beaming: # of Chinese tree nodes in black or grey

$\text{time: } O(nbc) \; \quad c: \# \text{ of edges at each node}$

linear-time if $c$ is a constant!  

phrased-based: linear-time
Experiments
Experimental Setup

- Chinese-to-English translation
  - on a Python implementation of tree-to-string system
- 1.5M sentence pairs (38M/32M words in Chn/Eng)
- dev: NIST 2006 (616 sent); test: NIST 2008 (691 sent)
- Chinese-side parsed by Berkeley parser (Petrov & Klein, 07)
- rules extracted using GHKM algorithm (Galley et al, 04; 06)
- trigram language model trained on the English side
- feature weights tuned using MERT (Och, 03)
Comparison with Moses

- we train/tune Moses with various distortion limits
- our incremental tree-to-string is \( \sim 30 \) times faster
- this includes parsing time (0.2s per sentence)

<table>
<thead>
<tr>
<th>C++</th>
<th>Moses (ttable=100, distortion=10)</th>
<th>BLEU</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>29.4</td>
<td>10.8s</td>
</tr>
<tr>
<td>Python</td>
<td>tree-to-string: incremental (b=10)</td>
<td>29.5</td>
<td>0.3s</td>
</tr>
</tbody>
</table>
Comparison with Moses

- we train/tune Moses with various distortion limits
- our incremental tree-to-string is ~30 times faster
- this includes parsing time (0.2s per sentence)

<table>
<thead>
<tr>
<th></th>
<th>C++</th>
<th>Python</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Moses (ttable=100, distortion=10)</td>
<td>tree-to-string: incremental (b=10)</td>
</tr>
<tr>
<td>BLEU</td>
<td>29.4</td>
<td>29.5</td>
</tr>
<tr>
<td>time</td>
<td>10.8s</td>
<td>0.3s</td>
</tr>
</tbody>
</table>
Comparison with Moses

- incremental tree-to-string is linear-time in practice
Comparison with Cube Pruning

- incremental is slightly faster than cube pruning
- note they are very different (orthogonal) techniques
- we envision their combination will be even faster
Comparison with Cube Pruning

- incremental is slightly faster than cube pruning
- note they are very different (orthogonal) techniques
- we envision their combination will be even faster
Comparison with Cube Pruning

- incremental is slightly faster than cube pruning
- note they are very different (orthogonal) techniques
- we envision their combination will be even faster
<table>
<thead>
<tr>
<th></th>
<th>phrase</th>
<th>syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>hypotheses</td>
<td>left-to-right</td>
<td>left-to-right</td>
</tr>
<tr>
<td>generation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in theory</td>
<td>exponential</td>
<td>polynomial</td>
</tr>
<tr>
<td>in practice</td>
<td>fast (O(n^2) or O(n) with beam search)</td>
<td>fast (with LM and huge rule set)</td>
</tr>
</tbody>
</table>

(A1): incremental decoding for tree-to-string translation

Q2: use a **small rule set, faster, a very slimmer model?**

39
Outline

- Background
  - Tree-to-String Translation
  - Phrase-based Decoding
- (Q1) Incremental Decoding for Tree-to-String Trans.
- (Q2) Rule Markov Model for Tree-to-String Trans.
- Conclusion
Where Rules Come From?

(Galley et al., 2004)
Where Rules Come From?

- extract word aligned source parse tree and target string

(Galley et al., 2004)
Where Rules Come From?

- extract word aligned source parse tree and target string
- cut into pieces, each of which will form a rule

(Galley et al., 2004)
Where Rules Come From?

- extract word aligned source parse tree and target string
- cut into pieces, each of which will form a rule
- where to cut?

(Galley et al., 2004)
Where Rules Come From?

- extract word aligned source parse tree and target string
- cut into pieces, each of which will form a rule
- where to cut?

(Galley et al., 2004)
Where Rules Come From?

- extract word aligned source parse tree and target string
- cut into pieces, each of which will form a rule
- where to cut?

(Galley et al., 2004)
Where Rules Come From?

- extract word aligned source parse tree and target string
- cut into pieces, each of which will form a rule
- where to cut?
Where Rules Come From?

- extract word aligned source parse tree and target string
- cut into pieces, each of which will form a rule
- cut points

(Galley et al., 2004)
• cut into pieces, each of which will form a rule
Where Rules Come From?

- cut into pieces, each of which will form a rule

(Galley et al., 2004)
Where Rules Come From?

- cut into pieces, each of which will form a rule

(Galley et al., 2004)
Where Rules Come From?

- cut into pieces, each of which will form a rule

(Galley et al., 2004)
Where Rules Come From?

- cut into pieces, each of which will form a rule

minimal rules: \( r_1' \) ... \( r_6' \)

(Galley et al., 2004)
Where Rules Come From?

Derivation tree:

composed rules:

Vertically

(NP)

“Bush with Sharon”

(NPB)

“Bush”

(Bùshí)

Bush

(NPB)

“Sharon”

(Shālóng)

(dp)

VPB

“held .. meeting”

(VV)

“held”

( jǔxíng)

le

(f)

hùtàn

NPB

“a meeting”

(NPB)

“held”

(“held .. meeting”)

(IP)

“Bush .. Sharon”

(Galley et al., 2004; 2006)
Where Rules Come From?

(Agalley et al., 2004; 2006)

composed rules:

$\text{r}_1' + \text{r}_2'$

vertically
Where Rules Come From?

 derivation tree

composed rules: $r_1' + r_2' + r_6'$

vertically

(Galley et al., 2004; 2006)
Where Rules Come From?

derivation tree

composed rules:
\[ r_1' + r_2' + r_5' + r_6' \]

(Galley et al., 2004; 2006)
Huge Composed Rules

<table>
<thead>
<tr>
<th>rule type</th>
<th>Max com. rule height</th>
<th># of min. + com. (million)</th>
<th>% of min. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.</td>
<td>3</td>
<td>3.9+245.1</td>
<td>1.57</td>
</tr>
<tr>
<td>com.</td>
<td>4</td>
<td>3.9+361.0</td>
<td>1.07</td>
</tr>
<tr>
<td>com.</td>
<td>7</td>
<td>3.9+444.8</td>
<td>0.87</td>
</tr>
</tbody>
</table>

- huge and redundant composed rules:
  - +2~3 BLEU points
  - decoding slow

GHKM (Galley et al. 2004; 2006)
Huge Composed Rules

<table>
<thead>
<tr>
<th>rule type</th>
<th>Max com. rule height</th>
<th># of min. + com. (million)</th>
<th>% of min. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.</td>
<td>3</td>
<td>3.9+245.1</td>
<td>1.57</td>
</tr>
<tr>
<td>com.</td>
<td>4</td>
<td>3.9+361.0</td>
<td>1.07</td>
</tr>
<tr>
<td>com.</td>
<td>7</td>
<td>3.9+444.8</td>
<td>0.87</td>
</tr>
</tbody>
</table>

- huge and redundant composed rules:
- +2~3 BLEU points
- decoding slow

Q2: use min. rules only, yet get same quality as com. rules?

GHKM (Galley et al. 2004; 2006)
How to do it?

- composed rules
- combine minimal rules

![Derivation tree diagram](image)
How to do it?

- composed rules
- combine minimal rules
- use independently

![Derivation Tree](image)
How to do it?

- composed rules
  - combine minimal rules
  - use independently
- rule Markov model (RMM)
  - model the composition
  - generate rules conditioned on their ancestors
    - only take into account their ancestors (vertical context)
  - \( P(T) = \prod_{r \in T} P(r | \text{anc}(r)) \)
How to do it?

- composed rules
  - combine minimal rules
  - use independently
- rule Markov model (RMM)
  - model the composition
  - generate rules conditioned on their ancestors
    - only take into account their ancestors (vertical context)

\[
P(T) = \prod_{r \in T} P(r \mid \text{anc}(r))
\]

\[
= P(r_1 \mid \varepsilon) \cdot P(r_2 \mid r_1) \cdot P(r_3 \mid r_1)
\cdot P(r_4 \mid r_1) \cdot P(r_5 \mid r_1) \cdot P(r_6 \mid r_1, r_2)
\]
Incremental Decoding with RMM

very easy!

derivation tree

<s> IP </s>
Incremental Decoding with RMM

derivation tree \( r_1 \)

\[ P(r_1|\epsilon) \]

action: predict (push)
Incremental Decoding with RMM

\[ P(r_2|r_1) \]

derivation tree \[ r_1 \]

\[ <s> \text{IP} \text{ </s>} \]

\[ \text{action: predict (push)} \]

\[
\begin{aligned}
&\text{NPB} \\
&\downarrow \text{VP} \\
&\text{IP} \\
&\downarrow \text{NPB} \\
&\text{NR} \\
&\text{Bush} \\
\end{aligned}
\]

\[
\begin{aligned}
&\text{PP} \\
&P \\
&\text{NPB} \\
&\downarrow \text{VV} \\
&\text{jǔxíng} \\
&\downarrow \text{AS} \\
&\text{le} \\
&\text{NPB} \\
&\downarrow \text{NN} \\
&\text{Shālǒng} \\
&\downarrow \text{hùtán} \\
\end{aligned}
\]
Incremental Decoding with RMM

action: scan
Incremental Decoding with RMM

action: complete (pop)
Incremental Decoding with RMM

Derivation tree:

\[ P(r_3|r_1) \]

\[ \text{action: predict (push)} \]

Example sentence:

&lt;s&gt; IP &lt;/s&gt;

NPB  VP
   PP  VPB
      |    |
     NPB  NPB
        |    |
       P  NPB
          |    |
         yǔ  NPB
            |    |
           jǔxíng  AS
              |    |
             le  NPB
                 |    |
                Shālóng  NNP
                    |    |
                   huìtán

NPB  VP
   NPB  with  NPB
Incremental Decoding with RMM

Derivation tree:

- \( r_1 \)
- \( r_2 \) with \( r_3 \)

Action: scan
Incremental Decoding with RMM

\[
P(r_4|r_1,r_3)
\]

Derivation tree:

\[
\begin{array}{c}
r_1 \\
\downarrow \\
r_2 \\
\downarrow \\
r_3 \\
\downarrow \\
r_4
\end{array}
\]

Action: predict (push)

<s> IP </s>

NPB

VP

NPB

Bush

held

NPB

with

NPB

action: predict (push)
Incremental Decoding with RMM

Derivation Tree:

```
<seq> IP </seq>
```

```
NPB
```

```
VP
```

```
PP
```

```
VPB
```

```
NPB
```

```
NR
```

```
Bush
```

```
held
```

```
NPB
```

```
with
```

```
NPB
```

```
talks
```

```
le
```

```
Shālóng
```

```
yǔ
```

```
jǔxíng
```

```
NN
```

```
huìtán
```

**Action:** scan
Incremental Decoding with RMM

action: complete (pop)
Incremental Decoding with RMM

derivation tree

<s> IP </s>

action: scan
Incremental Decoding with RMM

action: predict (push)
Incremental Decoding with RMM

action: scan

derivation tree

<s> IP </s>

NPB

VP

Bush

NPB

VP

NPB

with

NPB

Sharon

action: scan

<IP>

PP

NPB

VPB

NPB

VP

NPB

Bush

NPB

VP

NPB

Sharon
Incremental Decoding with RMM

**derivation tree**

```
<\s> IP <\s>

NPB  \\
| VP

PP  \\
| VPB

P  \\
| NPB  \\
| NR  \\
| Bùshì

VV  \\
| AS

yǔ  \\
| jǔxíng

le

NPB  \\
| NNP  \\
| Shālóng

NPB

action: complete (pop)
```

- **NPB**: noun phrase boundary
- **VP**: verb phrase
- **IP**: (internal) parse
- **PP**: prepositional phrase
- **NN**: noun
- **NR**: noun root
- **VV**: verb
- **AS**: argument structure
Incremental Decoding with RMM

Derivation tree:

<s> IP </s>

NPB

VP

IP

NPB

VP

action: complete (pop)
Incremental Decoding with RMM

**derivation tree**

```
  r1
    r2
      r3
        r4
          r5
```
Experiments
Experimental Setup

- same as the incremental decoding experiments
  - Chinese-to-English translation
  - ...... 
- differences
  - use all tree-to-string rules (c>0)
  - don’t use bilingual phrase
  - larger pruning threshold for translation forest
  - larger beam
- slower than previous experiments
## Results

- baseline

<table>
<thead>
<tr>
<th>grammar</th>
<th>RMM</th>
<th>Max rule height</th>
<th>parameters (million)</th>
<th>BLEU</th>
<th>time</th>
<th>PM</th>
<th>decoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>min.</td>
<td>None</td>
<td>3</td>
<td>3.9</td>
<td>26.2</td>
<td>1.28s</td>
<td>0.37s</td>
<td></td>
</tr>
</tbody>
</table>
## Results

- rule Markov model
- improves translation quality significantly

<table>
<thead>
<tr>
<th>grammar</th>
<th>RMM</th>
<th>Max rule height</th>
<th>parameters (million)</th>
<th>BLEU</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>min.</td>
<td>None</td>
<td>3</td>
<td>3.9</td>
<td>26.2</td>
<td>0.37s</td>
</tr>
<tr>
<td></td>
<td>bigram</td>
<td></td>
<td>3.9+4.7</td>
<td>27.5</td>
<td>1.28s</td>
</tr>
<tr>
<td></td>
<td>trigram</td>
<td></td>
<td>3.9+7.6</td>
<td>28.4</td>
<td>1.44s</td>
</tr>
<tr>
<td></td>
<td>7%</td>
<td></td>
<td>448.7+7.6</td>
<td>30.0</td>
<td>2.54s</td>
</tr>
</tbody>
</table>
# Results

- rule Markov model
- improves translation quality significantly
- runs faster, yet same quality than vertically com.

<table>
<thead>
<tr>
<th>grammar</th>
<th>RMM</th>
<th>Max rule height</th>
<th>parameters (million)</th>
<th>BLEU</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>min.</td>
<td>None</td>
<td></td>
<td>3.9</td>
<td>26.2</td>
<td></td>
</tr>
<tr>
<td>bigram</td>
<td>3</td>
<td></td>
<td>3.9+4.7</td>
<td>27.5</td>
<td>0.37s</td>
</tr>
<tr>
<td>trigram</td>
<td>3</td>
<td></td>
<td>3.9+7.6</td>
<td>28.4</td>
<td>0.37s</td>
</tr>
<tr>
<td>vertical com.</td>
<td>None</td>
<td>7</td>
<td>176.8</td>
<td>28.5</td>
<td>1.57s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>time</th>
<th>PM</th>
<th>decoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>min.</td>
<td>1.28s</td>
<td>1.44s</td>
</tr>
<tr>
<td>bigram</td>
<td>0.37s</td>
<td>1.44s</td>
</tr>
<tr>
<td>trigram</td>
<td>0.37s</td>
<td>2.54s</td>
</tr>
<tr>
<td>vertical com.</td>
<td>1.57s</td>
<td>1.62s</td>
</tr>
</tbody>
</table>
## Results

- rule Markov model
- faster, same translation quality as composed rules

<table>
<thead>
<tr>
<th>grammar</th>
<th>RMM</th>
<th>Max rule height</th>
<th>parameters (million)</th>
<th>BLEU</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PM</td>
</tr>
<tr>
<td>min.</td>
<td>None</td>
<td></td>
<td>3.9</td>
<td>26.2</td>
<td>0.37s</td>
</tr>
<tr>
<td></td>
<td>bigram</td>
<td>3</td>
<td>3.9+4.7</td>
<td>27.5</td>
<td>1.28s</td>
</tr>
<tr>
<td></td>
<td>trigram</td>
<td></td>
<td>3.9+7.6</td>
<td>28.4</td>
<td>1.57s</td>
</tr>
<tr>
<td>vertical com.</td>
<td>None</td>
<td>7</td>
<td>176.8</td>
<td>28.5</td>
<td>1.57s</td>
</tr>
<tr>
<td>com.</td>
<td>None</td>
<td>3</td>
<td>249.0</td>
<td>28.4</td>
<td>1.5s</td>
</tr>
</tbody>
</table>
## Results

- rule Markov model
- still lower than composed rules with higher height

<table>
<thead>
<tr>
<th>grammar</th>
<th>RMM</th>
<th>Max rule height</th>
<th>parameters (million)</th>
<th>BLEU</th>
<th>time</th>
<th>PM</th>
<th>decoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>min.</td>
<td>None</td>
<td></td>
<td>3.9</td>
<td>26.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bigram</td>
<td>None</td>
<td>3</td>
<td>3.9+4.7</td>
<td>27.5</td>
<td>0.37</td>
<td>1.28</td>
<td></td>
</tr>
<tr>
<td>trigram</td>
<td>None</td>
<td></td>
<td>3.9+7.6</td>
<td><strong>28.4</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vertical com.</td>
<td>None</td>
<td>7</td>
<td>176.8</td>
<td>28.5</td>
<td>1.57</td>
<td>1.62</td>
<td></td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>3</td>
<td>249.0</td>
<td>28.4</td>
<td>1.5</td>
<td>1.48</td>
<td></td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>7</td>
<td>448.7</td>
<td><strong>29.4</strong></td>
<td>5.46</td>
<td>1.30</td>
<td></td>
</tr>
</tbody>
</table>
## Results

- rule Markov model
- +composed rules
- can improve translation quality further

<table>
<thead>
<tr>
<th>grammar</th>
<th>RMM</th>
<th>Max rule height</th>
<th>parameters (million)</th>
<th>BLEU</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PM</td>
</tr>
<tr>
<td>min.</td>
<td>None</td>
<td></td>
<td>3.9</td>
<td>26.2</td>
<td>0.37s</td>
</tr>
<tr>
<td></td>
<td>bigram</td>
<td>3</td>
<td>3.9+4.7</td>
<td>27.5</td>
<td>1.44s</td>
</tr>
<tr>
<td></td>
<td>trigram</td>
<td></td>
<td>3.9+7.6</td>
<td>28.4</td>
<td>2.54s</td>
</tr>
<tr>
<td>vertical com.</td>
<td>None</td>
<td>7</td>
<td>176.8</td>
<td>28.5</td>
<td>1.57s</td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>3</td>
<td>249.0</td>
<td>28.4</td>
<td>1.5s</td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>7</td>
<td>448.7</td>
<td>29.4</td>
<td>5.46s</td>
</tr>
<tr>
<td></td>
<td>trigram</td>
<td>7</td>
<td>448.7+7.6</td>
<td>30.0</td>
<td>5.46s</td>
</tr>
</tbody>
</table>
## Results

- a slimmer model (only use minimal rules)
- with same Max rule height for PM
- faster, same translation quality as composed rules

<table>
<thead>
<tr>
<th>grammar</th>
<th>RMM</th>
<th>Max rule height</th>
<th>parameters (million)</th>
<th>BLEU</th>
<th>PM</th>
<th>decoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>min.</td>
<td>None</td>
<td>3</td>
<td>3.9</td>
<td>26.2</td>
<td>0.37s</td>
<td>1.28s</td>
</tr>
<tr>
<td></td>
<td>bigram</td>
<td>3</td>
<td>3.9+4.7</td>
<td>27.5</td>
<td>1.57s</td>
<td>1.44s</td>
</tr>
<tr>
<td></td>
<td>trigram</td>
<td>3</td>
<td>3.9+7.6</td>
<td>28.4</td>
<td>1.5s</td>
<td>2.54s</td>
</tr>
<tr>
<td>vertical com.</td>
<td>None</td>
<td>7</td>
<td>176.8</td>
<td>28.5</td>
<td>1.5s</td>
<td>1.62s</td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>3</td>
<td>249.0</td>
<td>28.4</td>
<td>1.5s</td>
<td>1.48s</td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>7</td>
<td>448.7</td>
<td>29.4</td>
<td>5.46s</td>
<td>1.30s</td>
</tr>
<tr>
<td></td>
<td>trigram</td>
<td>7</td>
<td>448.7+7.6</td>
<td>30.0</td>
<td>5.46s</td>
<td>2.80s</td>
</tr>
</tbody>
</table>
### Summary

<table>
<thead>
<tr>
<th>model</th>
<th>reordering</th>
<th>target gen.</th>
<th>rules</th>
<th>in theory</th>
<th>beam search</th>
</tr>
</thead>
<tbody>
<tr>
<td>phrase</td>
<td>local</td>
<td>incremental</td>
<td>phrases</td>
<td>$O(2^n n^2 V^{m-1})$</td>
<td>$O(n b d_{\text{max}})$</td>
</tr>
<tr>
<td>string-to-tree</td>
<td>global</td>
<td>bottom-up</td>
<td>com.</td>
<td>$O(n^3 c V^{4(m-1)})$</td>
<td>$O(n^3 c b^2)$</td>
</tr>
<tr>
<td>tree-to-string</td>
<td>global</td>
<td>bottom-up</td>
<td>com.</td>
<td>$O(n c V^{4(m-1)})$</td>
<td>$O(n c b^2)$</td>
</tr>
<tr>
<td></td>
<td>global</td>
<td>incremental</td>
<td>min.</td>
<td>$O(n^{\log(c)} c V^{m-1})$</td>
<td>$O(n c b)$</td>
</tr>
</tbody>
</table>

$n$: sentence length; $V$: English vocabulary size; $b$: beam size; $m$: order of LM; $c$: # of hyperedges at each node;
Conclusion

- **incremental** decoding for tree-to-string translation
- generating target hypothesis strictly left-to-right
- polynomial-time in theory, linear-time in practice

<table>
<thead>
<tr>
<th></th>
<th><em>in theory</em></th>
<th><em>in practice</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>phrase-based</td>
<td>exponential</td>
<td>linear</td>
</tr>
<tr>
<td>tree-to-string</td>
<td>polynomial</td>
<td>linear</td>
</tr>
</tbody>
</table>

- **rule Markov model** for incremental tree-to-string
  - a slimmer model (only use minimal rules)
  - faster, same translation quality as composed rules
  - (with same Max rule height)
非常 感谢！
Thank you very much!

非常 感谢！