Forest Rescoring

Faster Decoding with Integrated Language Models

Liang Huang

David Chiang

Penn

ISI

ACL 2007, Praha, Česká republika
Statistical Machine Translation

Spanish/English Bilingual Text

Statistical Analysis

Translation model (TM) competency

Broken English

Language model (LM) fluency

English

Que hambre tengo yo

What hunger have I
Hungry I am so
Have I that hunger
I am so hungry
How hunger have I
...

I am so hungry

Huang and Chiang

(Knight and Koehn, 2003)
Statistical Machine Translation

- **Spanish/English Bilingual Text**
  - **Statistical Analysis**
  - **translation model (TM)**: competency
  - **Broken English**: English

- **English Text**
  - **Statistical Analysis**
  - **language model (LM)**: fluency
  - **I am so hungry**: English

**n-best rescoring**

- Que hambre tengo yo
- Have I that hunger
- I am so hungry
  - How hunger have I
  - I am so hungry

**Huang and Chiang**

(Knight and Koehn, 2003)
Statistical Machine Translation

- **Spanish/English Bilingual Text**
  - Statistical Analysis
  - translate model (TM) competency
  - Broken English
  - language model (LM) fluency
  - English

- **English Text**
  - Statistical Analysis
Statistical Machine Translation

Spanish/English Bilingual Text

Statistical Analysis

Translation model (TM) competency

Broken English

language model (LM) fluency

English Text

Statistical Analysis

Spanish

Que hambre tengo yo

I am so hungry

integrated decoder

computationally challenging! 😞
Statistical Machine Translation

- Spanish/English Bilingual Text
- English Text
- Que hambre tengo yo: I am so hungry

Statistical Analysis

Phrase-based TM

Syntactic-based TM

Broken English

$n$-gram LM

Integrated decoder

Computationally challenging! 😞
Forest Rescoring

Spanish/English Bilingual Text
Statistical Analysis
Spanish

English Text
Statistical Analysis
Broken English

n-gram LM

packed forest

Que hambre tengo yo
integrated decoder
I am so hungry

computationally challenging! 😞
Forest Rescoring

Statistical Analysis

Spanish/English Bilingual Text

On-the-fly rescoring

Que hambre tengo yo

Integrated decoder

Computationally challenging! 😞

I am so hungry

English Text

$n$-gram LM

English

Spanish

Broken English

Huang and Chiang
Forest Rescoring

- Spanish/English Bilingual Text
- English Text
- Statistical Analysis
- Statistical Analysis
- packed forest
- integrated decoder
- forest rescorer
- on-the-fly rescoring

Que hambre tengo yo → forest rescorer → I am so hungry

significant speed-up: 10~30 times faster! 😊

Huang and Chiang
The Forest Framework

unifying phrase- and syntax-based decoding
Phrase-based Decoding

与 沙龙 举行 了 会谈

yu Shalong juxing le huitan

held a talk with Sharon

source-side: coverage vector

held a talk

target-side: grow hypotheses

strictly left-to-right

held a talk

held a talk with Sharon
Syntax-based Translation

- synchronous context-free grammars (SCFGs)
- context-free grammar in two dimensions
- generating pairs of strings/trees simultaneously
- co-indexed nonterminal further rewritten as a unit

\[
\begin{align*}
VP & \rightarrow \text{PP}^{(1)} \text{VP}^{(2)}, & \text{VP}^{(2)} & \rightarrow \text{PP}^{(1)} \\
VP & \rightarrow \text{juxing le huitan}, & \text{held a meeting} \\
\text{PP} & \rightarrow \text{yu Shalong}, & \text{with Sharon}
\end{align*}
\]

Huang and Chiang
Translation as Parsing

- translation with SCFGs => monolingual parsing
- parse the source input with the source projection
- build the corresponding target sub-strings in parallel

\[
\begin{align*}
\text{VP} & \rightarrow \text{PP}^{(1)} \text{ VP}^{(2)}, \\
\text{VP} & \rightarrow \text{juxing le huitan,} \\
\text{PP} & \rightarrow \text{yu Shalong,}
\end{align*}
\]
Translation as Parsing

- translation with SCFGs => monolingual parsing
- parse the source input with the source projection
- build the corresponding target sub-strings in parallel

\[
\begin{align*}
VP & \rightarrow PP^{(1)} \ VP^{(2)}, \quad VP^{(2)} \ PP^{(1)} \\
VP & \rightarrow juxing \ le \ huitan, \quad \text{held a meeting} \\
PP & \rightarrow \ yu \ Shalong, \quad \text{with Sharon}
\end{align*}
\]
Translation as Parsing

- translation with SCFGs => monolingual parsing
- parse the source input with the source projection
- build the corresponding target sub-strings in parallel

\[
\begin{align*}
\text{VP} & \rightarrow \text{PP}(1) \text{ VP}(2), & \text{VP}(2) \text{ PP}(1) \\
\text{VP} & \rightarrow \text{juxing le huitan}, & \text{held a meeting} \\
\text{PP} & \rightarrow \text{yu Shalong}, & \text{with Sharon} \\
\end{align*}
\]
Packed Forest

- a compact representation of all translations
- has a structure of hypergraph (graph is a special case)

phrase-based: graph  syntax-based: hypergraph
Packed Forest

• a **compact** representation of all translations

• has a structure of **hypergraph** (graph is a special case)

**phrase-based:** graph  
**syntax-based:** hypergraph
Adding a Bigram Model

Huang and Chiang
Adding a Bigram Model

Huang and Chiang
Adding a Bigram Model

Huang and Chiang

- VP1, 6
  - PP1, 3
    - with ... Sharon
    - along ... Sharon
    - with ... Shalong
  - VP3, 6
    - held ... talk
    - held ... meeting
    - hold ... talks

+LM items

- bigram
  - ... talk
  - ... meeting
  - ... Shalong
  - ... Sharon

Forest Rescoring 10
Adding a Bigram Model

Huang and Chiang

Sharon

bigram

+LM items

Held ... talk with ... Sharon

Along ... Sharon

Hold ... talks

Forest Rescoring
Huang and Chiang

Adding a Bigram Model

PP\textsubscript{1}, 3 → with ... Sharon

VP\textsubscript{1}, 6 → with ... talk

VP\textsubscript{3}, 6 → held ... talk

PP\textsubscript{1}, 3 → along ... Sharon

with ... Shalong

held ... meeting

+LM items
Adding a Bigram Model

Huang and Chiang
Adding a Bigram Model

Huang and Chiang
Conventional Beam Search

- beam search: only keep top-k +LM items at each node
- but there are many ways to derive each node
- can we avoid enumerating all combinations?
  - best-first enumeration?
Cube Pruning

monotonic grid?

\[
\begin{array}{|c|c|c|c|}
\hline
 & 1.0 & 3.0 & 8.0 \\
\hline
(VP_{3,6}^{\text{held}} \ast \text{meeting}) & 1.0 & 2.0 & 4.0 \\
(VP_{3,6}^{\text{held}} \ast \text{talk}) & 1.1 & 2.1 & 4.1 \\
(VP_{3,6}^{\text{hold}} \ast \text{conference}) & 3.5 & 4.5 & 6.5 \\
\hline
\end{array}
\]
Cube Pruning

non-monotonic grid due to LM combo costs

<table>
<thead>
<tr>
<th></th>
<th>1.0</th>
<th>3.0</th>
<th>8.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{VP}_{3,6}^{\text{held} \ast \text{meeting}} )</td>
<td>1.0</td>
<td>2.0 + 0.5</td>
<td>4.0 + 5.0</td>
</tr>
<tr>
<td>( \text{VP}_{3,6}^{\text{held} \ast \text{talk}} )</td>
<td>1.1</td>
<td>2.1 + 0.3</td>
<td>4.1 + 5.4</td>
</tr>
<tr>
<td>( \text{VP}_{3,6}^{\text{hold} \ast \text{conference}} )</td>
<td>3.5</td>
<td>4.5 + 0.6</td>
<td>6.5 + 10.5</td>
</tr>
</tbody>
</table>
Cube Pruning

non-monotonic grid due to LM combo costs

\[
\begin{array}{c|c|c|c}
 & 1.0 & 3.0 & 8.0 \\
\hline
(\text{VP}^{\text{hold}}_{3,6} \ast \text{meeting}) & 1.0 & 2.0 + 0.5 & 4.0 + 5.0 & 9.0 + 0.5 \\
(\text{VP}^{\text{hold}}_{3,6} \ast \text{talk}) & 1.1 & 2.1 + 0.3 & 4.1 + 5.4 & 9.1 + 0.3 \\
(\text{VP}^{\text{hold}}_{3,6} \ast \text{conference}) & 3.5 & 4.5 + 0.6 & 6.5 + 10.5 & 11.5 + 0.6 \\
\end{array}
\]

bigram \{(\text{meeting}, \text{with})\}
**Cube Pruning**

<table>
<thead>
<tr>
<th></th>
<th>(PP with * Sharon)</th>
<th>(PP along * Sharon)</th>
<th>(PP with * Shalong)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PP1,3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VP1,6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VP3,6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**non-monotonic grid due to LM combo costs**

<table>
<thead>
<tr>
<th></th>
<th>1.0</th>
<th>3.0</th>
<th>8.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>(VP_\text{held} \star \text{meeting})</td>
<td>1.0</td>
<td>2.5</td>
<td>9.0</td>
</tr>
<tr>
<td>(VP_\text{held} \star \text{talk})</td>
<td>1.1</td>
<td>2.4</td>
<td>9.5</td>
</tr>
<tr>
<td>(VP_\text{hold} \star \text{conference})</td>
<td>3.5</td>
<td>5.1</td>
<td>17.0</td>
</tr>
</tbody>
</table>
**Cube Pruning**

$k$-best parsing  
(Huang and Chiang, 2005)

- a priority queue of candidates
- extract the best candidate

<table>
<thead>
<tr>
<th></th>
<th>(PP with * Sharon )</th>
<th>(PP along * Sharon )</th>
<th>(PP with * Shalong )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.0</strong></td>
<td>1.0</td>
<td>3.0</td>
<td>8.0</td>
</tr>
<tr>
<td><strong>2.5</strong></td>
<td><strong>3.5</strong></td>
<td>9.0</td>
<td>9.5</td>
</tr>
<tr>
<td><strong>9.0</strong></td>
<td>9.5</td>
<td>9.4</td>
<td></td>
</tr>
<tr>
<td><strong>5.1</strong></td>
<td><strong>17.0</strong></td>
<td>12.1</td>
<td></td>
</tr>
</tbody>
</table>

(VP\(_{3,6}^{\text{held} \ast \text{meeting}}\))

(VP\(_{3,6}^{\text{held} \ast \text{talk}}\))

(VP\(_{3,6}^{\text{hold} \ast \text{conference}}\))

Huang and Chiang
**Cube Pruning**

$k$-best parsing
(Huang and Chiang, 2005)

- a priority queue of candidates
- extract the best candidate
- push the two successors

<table>
<thead>
<tr>
<th></th>
<th>PP with * Sharon</th>
<th>PP along * Sharon</th>
<th>PP with * Shalong</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>k</strong>-best parsing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>(VP</strong> * meeting)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>(VP</strong> * talk)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>(VP</strong> * conference)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>1.0</strong></td>
<td><strong>2.5</strong></td>
<td><strong>9.0</strong></td>
<td><strong>9.5</strong></td>
</tr>
<tr>
<td><strong>1.1</strong></td>
<td><strong>2.4</strong></td>
<td><strong>9.5</strong></td>
<td><strong>9.4</strong></td>
</tr>
<tr>
<td><strong>3.5</strong></td>
<td><strong>5.1</strong></td>
<td><strong>17.0</strong></td>
<td><strong>12.1</strong></td>
</tr>
</tbody>
</table>

Huang and Chiang
### Cube Pruning

**k-best parsing**
(Huang and Chiang, 2005)

- a priority queue of candidates
- extract the best candidate
- push the two successors

<table>
<thead>
<tr>
<th></th>
<th>PP with * Sharon</th>
<th>PP along * Sharon</th>
<th>PP with * Shalong</th>
</tr>
</thead>
<tbody>
<tr>
<td>(VP (3,6) held * meeting)</td>
<td>1.0</td>
<td>3.0</td>
<td>8.0</td>
</tr>
<tr>
<td>(VP (3,6) held * talk)</td>
<td>1.1</td>
<td>2.4</td>
<td>9.5</td>
</tr>
<tr>
<td>(VP (3,6) hold * conference)</td>
<td>3.5</td>
<td>5.1</td>
<td>17.0</td>
</tr>
</tbody>
</table>

Huang and Chiang
Cube Pruning

items are popped out-of-order

solution: keep a buffer of pop-ups

```
2.5  2.4  5.1
```

<table>
<thead>
<tr>
<th></th>
<th>1.0</th>
<th>3.0</th>
<th>8.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>(VP&lt;sub&gt;3,6&lt;/sub&gt; with ⋆ Sharon)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(PP&lt;sub&gt;1,3&lt;/sub&gt; along ⋆ Sharon)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(PP&lt;sub&gt;1,3&lt;/sub&gt; with ⋆ Shalong)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
(VP<sub>3,6</sub> held ⋆ meeting) 1.0  2.5  9.0  9.5
(VP<sub>3,6</sub> held ⋆ talk) 1.1  2.4  9.5  9.4
(VP<sub>3,6</sub> hold ⋆ conference) 3.5  5.1 17.0 12.1
```
**Cube Pruning**

Items are popped out-of-order

**solution:** keep a buffer of pop-ups

2.5 2.4 5.1

Finally re-sort the buffer and return inorder:

2.4 2.5 5.1

<table>
<thead>
<tr>
<th></th>
<th>1.0</th>
<th>3.0</th>
<th>8.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>(VP \text{\textsuperscript{held} \ast meeting})</td>
<td>1.0</td>
<td>2.5</td>
<td>9.0</td>
</tr>
<tr>
<td>(VP \text{\textsuperscript{held} \ast talk})</td>
<td>1.1</td>
<td>2.4</td>
<td>9.5</td>
</tr>
<tr>
<td>(VP \text{\textsuperscript{hold} \ast conference})</td>
<td>3.5</td>
<td>5.1</td>
<td>17.0</td>
</tr>
</tbody>
</table>
Across Hyperedges

$k$-best parsing
(Huang and Chiang, 2005)

hyperedge

process all hyperedges *simultaneously*!
significant savings of computation
Across Hyperedges

\( k \)-best parsing
(Huang and Chiang, 2005)

on-the-fly rescoring at each node,
instead of only at the root node

process all hyperedges \textit{simultaneously}!

significant savings of computation
Cube Growing

- an even faster variant of cube pruning
- motivation
  - why do we have a fixed beam of size $k$ at each node?
    - why don’t we on-the-fly figure out the minimum $k$?
- cube growing uses
  - lazy $k$-best parsing (Huang and Chiang, 2005, Algorithm 3)
  - on-demand computation
- but harder to implement
### Cube Pruning vs. Growing

<table>
<thead>
<tr>
<th></th>
<th>Cube Pruning</th>
<th>Cube Growing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Based on k-best</strong></td>
<td>Algorithm 2</td>
<td>Algorithm 3</td>
</tr>
<tr>
<td><strong>Traversal</strong></td>
<td>bottom-up</td>
<td>top-down</td>
</tr>
<tr>
<td><strong>Multipass</strong></td>
<td>implicit</td>
<td>explicit</td>
</tr>
<tr>
<td><strong>Need heuristic?</strong></td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td><strong>Implementation</strong></td>
<td>easy</td>
<td>hard</td>
</tr>
<tr>
<td><strong>Optimality</strong></td>
<td>no guarantee</td>
<td>if admiss. heuristic and no hard limit</td>
</tr>
</tbody>
</table>

Liang Huang (Penn) 21 Faster Decoding for Syntax MT
Syntax-based Experiments
Tree-to-String System

- syntax-directed, English to Chinese (Huang, Knight, Joshi, 2006)
- first parse input, and then recursively transfer

synchronous tree-substitution grammars (STSG) (Galley et al., 2004; Eisner, 2003)

search space still a hypergraph

tested on 140 sentences slightly better BLEU scores than Pharaoh

Huang and Chiang
Tree-to-String System

- syntax-directed, English to Chinese (Huang, Knight, Joshi, 2006)
- first parse input, and then recursively transfer

synchronous tree-substitution grammars (STSG)
(Galley et al., 2004; Eisner, 2003)

search space still a hypergraph

tested on 140 sentences slightly better BLEU scores than Pharaoh
Speed vs. Search Quality

![Graph showing the relationship between speed and search quality. The graph compares full-integration, cube pruning, and cube growing methods. The x-axis represents the average number of +LM items explored per sentence, and the y-axis represents the average model cost. The legend indicates that speed is improved with the increase in search quality.](image-url)
Speed vs. Search Quality

- Speed ++
- Quality ++

10 times faster

Huang and Chiang
Speed vs. Search Quality

![Graph showing the relationship between average model cost and average number of +LM items explored per sentence. The graph compares full-integration, cube pruning, and cube growing methods, showing that full-integration is 10 times faster while maintaining quality ++.]

- speed ++
- quality ++

**Legend:**
- full-integration
- cube pruning
- cube growing

**Axes:**
- Y-axis: (−log Prob)
- X-axis: average number of +LM items explored per sentence
Speed vs. Search Quality

- speed ++
- quality ++

10 times faster

Huang and Chiang

Forest Rescoring
Speed vs. Search Quality

- 10 times faster
- speed ++
- quality ++

The graph shows the trade-off between speed and search quality for different rescoring methods. The x-axis represents the average number of +LM items explored per sentence, while the y-axis shows the average model cost. The graph includes lines for full-integration, cube pruning, and cube growing, all with the same parameters.

- 10 times faster

Huang and Chiang

Forest Rescoring 24
Speed vs. Translation Accuracy

- Speed ++
- Quality ++

Graph showing the relationship between BLEU score and the average number of +LM items explored per sentence for different methods:

- Full-integration
- Cube pruning
- Cube growing

Huang and Chiang

Forest Rescoring
Cube-Pruning for
Phrase-based Decoding
Syntax vs. Phrase-based

- VP
  - PP1, 3
  - VP3, 6
  - PP1, 4
  - VP4, 6

- NP1, 4
- VP4, 6

With Sharon

- Talk
- Meeting
- Talks

- Held
- Shalong
- Minister

- Held
- Hold
- Did

... talk
... meeting
... talks

... Sharon
... Shalong
... minister

... held
... hold
... did
Syntax vs. Phrase-based

Huang and Chiang
Alternative Phrase-Pairs

grouping into hyperedge bundles

... talk
... meeting
... talks

... Sharon
... Shalong
... minister

... held
... hold
... did

Huang and Chiang

Forest Rescoring 28
Alternative Phrase-Pairs

grouping into hyperedge bundles

Pharaoh would explore all cells

... held
... hold
... did

... Shalong
... minister

... Sharon

... hold a meeting
... held a talk
... conference a meeting a talk

... talk
... meeting
... talks

with Ariel Sharon
and Sharon
with Sharon

Huang and Chiang
but we explore the grids
in a best-first fashion

... talk
... meeting
... talks

... Sharon
... Shalong
... minister

... held
... hold
... did

... held a reunion
... held a meeting
... held a talk

conference
a meeting
a talk

with Ariel Sharon
with Sharon
and Sharon
but we explore the grids in a best-first fashion

in practice we use per-bin pruning as in Pharaoh

---

... talk
... meeting
... talks

... Sharon
... Shalong
... minister

Huang and Chiang
Pharaoh hypothesis expansion

Pharaoh
(forward)

Equivalent version
(backward)
In Practice: per-bin Pruning

Pharaoh
(forward)

1

2

3

4

5
In Practice: per-bin Pruning

Pharaoh (forward)

Cube Pruning (backward)
In Practice: per-bin Pruning

Cube Pruning (backward)

Huang and Chiang
In Practice: per-bin Pruning

Cube Pruning (backward)

1.0 4.0 7.0

1.0 2.5 8.3 8.5

1.1 2.4 9.5 8.4

3.5 9.2 17.0 15.2

Huang and Chiang

Forest Rescoring 31
In Practice: per-bin Pruning

Cube Pruning (backward)

hyperedge bundles

with Sharon and Sharon

with Ariel Sharon

1.0  4.0  7.0

1.0  2.5  8.3  8.5

1.1  2.4  9.5  8.4

3.5  9.2  17.0  15.2
In Practice: per-bin Pruning

Cube Pruning (backward)

Hyperedge bundles

<table>
<thead>
<tr>
<th>Huang and Chiang</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1.0</th>
<th>2.5</th>
<th>8.3</th>
<th>8.5</th>
<th>9.2</th>
<th>17.0</th>
<th>15.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>meeting</td>
<td>1.0</td>
<td>2.4</td>
<td>9.5</td>
<td>8.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>talk</td>
<td>1.1</td>
<td></td>
<td></td>
<td></td>
<td>9.2</td>
<td>17.0</td>
<td>15.2</td>
</tr>
<tr>
<td>conference</td>
<td>3.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Forest Rescoring 31
In Practice: per-bin Pruning

Cube Pruning
(backward)

hyperedge bundles

with Sharon
and Sharon
with Ariel Sharon

1.0 4.0 7.0

(_●●● meeting) 1.0 2.5 8.3
(_●●● talk) 1.1 2.4
(_●●● conference) 3.5

Huang and Chiang

Forest Rescoring
In Practice: per-bin Pruning

Hyperedge bundles

Cube Pruning (backward)

1.0 2.5 8.3
1.1 2.4
3.5

Huang and Chiang
Huang and Chiang

Forest Rescoring

Speed vs. Search Quality

tested on our faithful clone of Pharaoh

- speed ++
- quality ++

Graph showing the relationship between average number of hypotheses per sentence and average model cost for full-integration and cube pruning methods.
Speed vs. Search Quality

tested on our faithful clone of Pharaoh

32 times faster

Huang and Chiang

Forest Rescoring
Speed vs. Search Quality

tested on our faithful clone of Pharaoh

- speed ++
- quality ++

32 times faster

Huang and Chiang

Forest Rescoring
Speed vs. Translation Accuracy

Huang and Chiang

Forest Rescoring 33
Speed vs. Translation Accuracy

~100 times faster

BLEU score vs. average number of hypotheses per sentence

- Pharaoh
- full-integration
- cube pruning

speed ++

quality ++
Conclusions

- forest-rescoring: cube pruning and cube growing
  - on-the-fly rescoring using $k$-best parsing
  - applicable to both phrase- and syntax-based systems
  - significant speed-up against conventional beam search
- general technique for reducing search spaces
  - effectiveness depends on scale of non-monotonicity
- future work
  - forest-reranking: parsing with non-local features
Thanks!

try out **Cubit**

a cube pruning decoder for phrase-based translation