Parallel Computation in a Free Merge World

Sandiway Fong
University of Arizona
with Jason Ginsburg
Osaka Kyoiku University

Acknowledgement: Dr. Nobuyoshi Asai, U. of Aizu, Japan for the test platform
Contents

• Linguistic Framework and Combinatorics
• Parallelizing the Framework and Results
• Feedback: Improving the Framework
• Parallelism: Job size
Linguistic Framework and Combinatorics
Phrase Structure Computation

[Joint work with Jason Ginsburg]
Theoretical basis:
- Chomsky (2007) and Oishi (2015)
- nominal and determiner phrase structure (n*/d*-root) parallels verbal phrase structure (v*-root)

Pair Merge (PM) analysis:
- <Determiner, Noun>
  - forced by non-head determiner
- cf. *{{d, root}, {n, root}}
  - (unlabeled Set Merge (SM))

Relabeling: Cecchetto & Donati (2015)
- my friend = {me, {'s, friend}}
- friend of mine = {friend, {me, {'s, friend}}}
Phrase Structure Computation

Example: syntactic object (SO)

- Example input: (a list of heads):
  [friend, n, [me, n,'s, d*], n*,[the, d]]

- Combinatorial Task: recursively apply operations:
  1. External Set Merge (ESM): form \{H, \alpha\}; SO: \alpha, Input: [H,...]
  2. Internal Set Merge (ISM(\beta)): form \{\beta, \alpha\}; SO: \alpha and \beta \subset^+ \alpha
  3. External Pair Merge (EPM): form <H, \alpha>; SO: \alpha, Input: [H,...]
  4. Internal Pair Merge (IPM): form <\alpha, \beta>; SO: \alpha and \beta \subset^+ \alpha
     with constraints such as:
     1. *<\beta[!F], \alpha> where !F = unvalued feature F
     2. *ISM(\beta_i) ISM(\beta_i); i.e. can’t ISM same \beta_i twice, etc.

- Example output:
  <{the, d}, {{friend, <{{me, n}, {{me, n}, 's}}, d*}, {friend, n}>, n}>*

- Example of questions answered by computation:
  A. is this the shortest derivation? YES
  B. are there other possible derivations? YES, only longer ones...
Manually Guided Derivation...

Table: sequence of operations leading to computed SO
Combinatorics for example†
† naïve version

Log10(# SOs Generated) vs. # Operations

- **logscale** y-axis:
  - e.g. $6 = 10^6 = \text{million}$
  - 15 operations deep:
    - 25 million SOs generated
    - 1 convergent SO
      - (see previous slide)
  - 16 operations deep:
    - 250 million SOs
    - 3 spurious SOs
      - (see next slides)
Parallelizing the Framework and Results
Two stages of parallel processing

Example:

• Stage 1: breadth-first derivation tree search (BFS):
  • SO₁..SO₆ are incomplete SOs that can be expanded further
  • □ represent dead-ends
  • go as deep as necessary to generate the number of starter SOs needed
  • example: going 10 deep nets us 1743 SOs
Step 1: Expose threads

Breadth-first search $n$ deep:

Blue SOs may be expanded further: these are our threads

# of threads generated @ depth
Step 2: Run threads in parallel

Our concern: load balancing
- threads binned by job size
- 1743 jobs (threads)
  - produced by initial BFS to 10 operations deep
  - each job (go 6 deep)
- 10x range in job size observed:
  - 50,000 SOs to 600,000 SOs
- 72% of jobs small:
  - belong to the 3 smallest bins, i.e. 0-150,000 SOs
Parallel Speedup

- **Runtime:**
  - Single thread: 759 (secs)
  - 32 threads: 57 (secs)

- **Speedup:** Amdahl’s Law
  - (theoretical limit)

---

**Runtime vs. # concurrent threads**

- **16 CPUs:** with HTT, 32 logical cores

**Source:** wikipedia

**13x speedup observed**

**we are somewhere here**
Is Hyper Threading (HTT) useful?

- Hyper Threading Technology:
  - each core has two sets of registers
  - hide memory latency

- Test platform:
  - Intel Xeon E5-2687W HTT-capable (2U), 128GB RAM
  - total of 16 cores (32 logical cores)

- region 8-32 threads:
  - shortest overall runtimes are all achieved by with HTT

- region 18-32 threads:
  - averages about 11.4% improvement over no HTT

- region 8-16 threads:
  - no HTT is 5.5% better
With Workspace (WS) Precomputation

• Results shown earlier, e.g. 57 (secs), were actually computed on a non-naïve model

• region $10^7$–$10^8$ SOs ("wall"):
  • too much for the test platform: approx. 4.5 hours CPU time

• Non-naïve model:
  • pre-compute sub-Workspace (WS) SOs
  • # operations required reduced
  • free Merge then is substantially easier
    “walk back from the wall”
Workspace (WS) Precomputation

- **Example:**
  instead of
  \[\text{friend, n, [me, n,’s, d*], n*, [the, d]}\]
  actually compute with
  \[\text{friend, n, \{\{me, n\}, \{me, n’, ’s\}, d*\}, n*,}
  \{\text{the, d}\}\]
  *i.e.* use pre-computed mappings:
  1. \([\text{me, n,’s, d*}] \mapsto \text{\{\{me, n\}, \{me, n’, ’s\}, d*\}}\]
  2. \([\text{the, d}] \mapsto \text{\{the, d\}}\]

- **Results:**
  - Depth 6: #SOs: 2,324; 1 solution
  - Depth 7: #SOs: 18,202; 2 solutions, etc..
Workspace (WS) Precomputation:

Manually Guided Derivation...

<table>
<thead>
<tr>
<th>Step</th>
<th>Branch</th>
<th>Op</th>
<th>SO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>the</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>esm</td>
<td>(me, n, case)</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>ism</td>
<td>(me, n, case)</td>
</tr>
</tbody>
</table>

Final output: [the]
Improve the Framework

• Parallel processing allows us to discover 5 extra analyzes at depth 9 (out of $\approx 10^6$ SOs) 10x quicker...
Extra Analyses Uncovered: Depth 7 & 8

Depth = 7
Analysis:
Extraneous ISM of \{friend,n\} to the edge of friend

\(<\phi, \phi>\) because
\{friend,n\} and friend have identical \(\phi\)-features

Depth = 8
Extra Analyses Uncovered: Depth 9
Improve the Framework: Theory Adjustment

[Joint work with Jason Ginsburg]

- block licensing of extraneous analyses
- Previously:
  - all Case valuation done through Agree
  - ‘s analyzed as a pair: root ‘s + d* (categorizer)
- Now:
  - distinguish Inherent from Structural Case
  - Inherent Case does not involve φ-features: means <φ, φ> labeling not available
  - Structural Case involves φ-feature valuation, and Nom (or Acc) Case for C/T/ (or v*/R)
  - ‘s analyzed as a single re-categorizing head: i.e. n -> d
Improve the Framework: Theory Adjustment

Depth = 6
Improve the Framework: Combinatorics

- **Orange line**: adjusted theory
  - one solution @ 6
  - no extraneous solutions @7-10
  - fewer SOs hypothesized
- **Blue line**: original theory
  - one solution @ 6
  - one solution @ 7, 8
  - five solutions @ 9
Parallelism: Job size
Parallel Processing Task Size

- **Example:**
  - say we want to search to depth 11 in parallel
  - What is the best way to divvy up the search?
  - We can perform the same search by expressing:
    - 27 threads, each 7 deep
    - 121 threads, each 6 deep
    - 610 threads, each 5 deep
    - 3750 threads, each 4 deep

- **Tradeoff:**
  - thread overhead vs. load balancing
    - (task size not a constant)
  - RAM wrt. # active threads limits task size
## Parallel Processing Task Size: Results

### Conditions:
- **blue line**: 16 CPUs used (no HTT); 16 active threads
- **green line**: same 16 CPUs + HTT; 32 active threads
- RAM: 128GB capacity

### Best results:
- HTT on
- 610 threads (from 6 deep initially), each job is 5 deep
- used ≈ 30GB RAM
  - cf. 4.7 split used ≈ 88GB
  - cf. 6.5 split used ≈ 15GB

### Table

<table>
<thead>
<tr>
<th>Split</th>
<th>Threads</th>
<th>Runtime</th>
<th>Runtime HTT</th>
<th>Walltime (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.7</td>
<td>27</td>
<td>954.64</td>
<td>848.07</td>
<td>1100.18</td>
</tr>
<tr>
<td>5.6</td>
<td>121</td>
<td>726.43</td>
<td>913.16</td>
<td>848.07</td>
</tr>
<tr>
<td>6.5</td>
<td>610</td>
<td>701.54</td>
<td>1030.29</td>
<td>913.16</td>
</tr>
<tr>
<td>7.4</td>
<td>3750</td>
<td>775.78</td>
<td>3750</td>
<td>913.16</td>
</tr>
</tbody>
</table>

![Graph showing runtime and threads vs. walltime](image-url)
Conclusions

• Application is parallel-friendly
  • search: multiple possible operations
  • speed-up results: 13x on 32 logical cores

• Speed-up allows us to search deeper
  • beyond a basic analysis

• Improve the theory
  • eliminate extraneous analyzes
Extra Analyses Uncovered

Depth = 9
\[ d^*P = \{\{\text{me,n}\},\{\{\text{me,n}\},\text{'s}\},\text{d}^*\} \]

ESM \( \text{friend,n} \)
ISM \( \text{'s} \)
ISM \( \{\text{me,n}\},\{\text{me,n}\},\text{'s}\}\) (need a d categorizer)
ISM \( \{\text{me,n}\},\{\{\text{me,n}\},\text{'s}\}\} \)
ISM \( \{\text{me,n}\},\text{'s}\}
ISM \( \{'s,\{\text{friend,n}\},d^*P\} \)
Extra Analyses Uncovered

```
\text{Depth} = 9
```

```
\text{LLC:\{friend.a\}case.\{\{me.a\},\{\{me.a\},\{s\}\}\}d^*},n^*case.\{the,d\}\text{ Derivation \#1}

\begin{array}{|c|c|c|}
\hline
\text{Step} & \text{Branch} & \text{Op} \text{ } \text{SO} \\
\hline
1 & - & - friend \\
2 & 1 & esm \{friend.a\}case \\
3 & 3 & esm \{\{friend.a\}case.\{\{me.a\},\{\{me.a\},\{s\}\}\}d^*\} \\
4 & 1 & ism \{\{s\},\{\{friend.a\}case.\{\{me.a\},\{\{me.a\},\{s\}\}\}d^*\}\} \\
5 & 5 & ism \{\{friend,a\}case.\{\{s\},\{\{friend.a\}case.\{\{me.a\},\{\{me.a\},\{s\}\}\}d^*\}\}\} \\
6 & 1 & ism \{\{me.a\},\{s\},\{\{friend,a\}case.\{\{s\},\{\{friend,a\}case.\{\{me.a\},\{\{me,a\},\{s\}\}\}d^*\}\}\}\} \\
7 & 4 & ism \{\{s\},\{\{friend,a\}case.\{\{me.a\},\{\{me.a\},\{s\}\}\}d^*\}\},\{\{me.a\},\{s\},\{friend,a\}case.\{\{me.a\},\{me.a\},\{s\}\}\}d^*\}\} \\
8 & 6 & esm \{\{s\},\{\{friend,a\}case.\{\{me.a\},\{\{me,a\},\{s\}\}\}d^*\}\},\{\{me.a\},\{s\},\{\{friend,a\}case.\{\{me,a\},\{me.a\},\{s\}\}\}d^*\}\} \\
9 & 1 & epm <\{the,d\},\{\{s\},\{\{friend,a\}case.\{\{me.a\},\{\{me,a\},\{s\}\}\}d^*\}\},\{\{me,a\},\{s\},\{\{friend,a\}case.\{\{me,a\},\{me.a\},\{s\}\}\}d^*\}\}n^*case > \\
\hline
\end{array}
```

Spelling heads: \{the,s,me\}

Final output: \{the,s,me\}
Extra Analyses Uncovered

**Derivation #1**

<table>
<thead>
<tr>
<th>Step</th>
<th>Branch</th>
<th>Op</th>
<th>SO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>Friend</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>esm</td>
<td>friend,n\n</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>esm</td>
<td>friend\n</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>esm</td>
<td>friend\n</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>esm</td>
<td>friend\n</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>esm</td>
<td>friend\n</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>esm</td>
<td>friend\n</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>esm</td>
<td>friend\n</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>cpm</td>
<td>friend\n</td>
</tr>
</tbody>
</table>

**Spelling Heads:** [the,friend,s,of,me]

**Final Output:** (the,friend,s,of,me)
Extra Analyses Uncovered

**Depth = 9**

---

**LIs (friend,n|case,{{(me,a),{(me,a),s}}},d*},n*|case,(the,d)) Derivation #1**

<table>
<thead>
<tr>
<th>Step</th>
<th>Branch</th>
<th>Op</th>
<th>SO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>friend</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>esm</td>
<td>(friend,n</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>esm</td>
<td>{{(me,a),{(me,a),s}}},d*}</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>esm</td>
<td>(friend,(friend,n</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>esm</td>
<td>{{(me,a),{(me,a),s}}},friend,(friend,n</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>esm</td>
<td>{{friend,n</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>ism</td>
<td>((friend,(friend,n</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>esm</td>
<td>((friend,(friend,n</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>epm</td>
<td>&lt;the,d&gt;,{{friend,(friend,n</td>
</tr>
</tbody>
</table>

Spellout heads: [the,friend|s,of,me]  
Final output: [the,friend|s,of,me]
Extra Analyses Uncovered

Depth = 9