Composing Finite State Transducers on GPUs
Arturo Argueta & David Chiang
{aargueta,dciang}@nd.edu,
Computer Science and Engineering, University of Notre Dame

Abstract
Weighted finite automata\(^1\) and transducers are widely used in Natural Language Processing (NLP) to perform tasks such as morphological analysis, part-of-speech tagging, chunking, named entity recognition, speech recognition, and others.

We introduce a Graphics Processing Unit (GPU) implementation of the composition algorithm, achieving speedups of up to 6x over serial implementations and 4.5x over OpenFST.

Merrill et al.\(^2\) showed that GPUs can be used to accelerate breadth-first search to traverse up to 3.3 billion edges per second.

Composition Algorithm
The composition algorithm feeds the output symbols from transducer \(M_1 = (Q, \Sigma, \Gamma, \delta, \begin{pmatrix} 0 \end{pmatrix})\) to a transducer \(M_2 = (Q, \Gamma, \Delta, \delta, \begin{pmatrix} 0 \end{pmatrix})\) where:
- \(Q\) a set of states
- \(\Sigma\) finite input vocabulary
- \(\Gamma\) finite output/input shared vocabulary
- \(\Delta\) finite output vocabulary
- \(s\) a start state
- \(F\) a set of final states
- \(\delta\) transition function: if \(M_1\) is in state \(q\) and the next input symbol is \(a\), then \(\delta(a)[q,q']\) is the weight of going to state \(q'\).

Our approach performs a breadth-first search (BFS) of the composed FST beginning from the start state of the two input transducers, so as to avoid creating inaccessible states. As it is standard, the BFS uses two data structures, a frontier queue (A) and a visited set (V), which is always a superset of A. For each state pair \(q_1, q_2\) (representing a state from \(M_1\) and \(q_2\) from \(M_2\)) popped from A, the algorithm composes all transitions that share the same symbol from the shared vocabulary. The composed edges are added to the final transducer, and the corresponding target states are added to the frontier for future expansion.

Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Time(s)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial</td>
<td>OpenFST</td>
<td>374</td>
</tr>
<tr>
<td></td>
<td>Our Serial</td>
<td>518</td>
</tr>
<tr>
<td>Parallel</td>
<td>Our CUDA</td>
<td>82.9</td>
</tr>
</tbody>
</table>

References

Data
The probabilities \(P(f|e)\) were obtained by using GIZA++ and used to train a first transducer. Word level alignments are used to generate a word-level transducer. To generate a transducer that “reads an utterance of length \(n\) and output a translation, the automata mentioned earlier are intersected with the word level transducers. De-en and en-de transducers were trained from the Europarl corpus using 150k parallel lines.