PDTgcomp: Compilation Framework for Data Transformation Kernels on GPU

Tri Nguyen and Michela Becchi (tnguye7, mbecchi@ncsu.edu)

Motivation
- Data transformation kernels are at the core of many data analytics, data processing, and scientific applications.
- Custom CPU and GPU libraries and hardware accelerators for specific data transformation tasks provide efficiency but lack generality and extensibility.
- Accelerating the computational abstraction at the core of data transformation can benefit a broad spectrum of applications.

Objective
- Compilation framework that generates accelerated GPU kernels for data transformation tasks expressed using Pushdown Transducers (PDTs).

Pushdown Transducers (PDT)

Example (CSV parsing): PDT that extracts the name of all female members from a CSV file containing "name, gender, occupation" tuples.

Conversion Steps
1. Convert the list of states into an if-else block in which each if-condition guards the content of a state.
2. For each state:
   a. convert the associated arithmetic operation (if any) to the appropriate arithmetic operation on stack values.
   b. convert the list of outgoing transitions into an if-else block, with an if-statement per transition. Generate if-conditions and statements according to the appropriate input, output and stack read/write operation.
3. Allocate stacks in shared memory.
4. Store the remaining context information (pointers in I/O streams, active state indicator) into local variables to be stored in the register file.

Basic Code

Optimizations
- Reduce code size
- Reduce control flow operations
- Improve memory access patterns
No optimizations

User Interface

PDT IR

GPU code

Performance

Experimental Setup

<table>
<thead>
<tr>
<th>Task</th>
<th>Data</th>
<th>Eri Dec</th>
<th>Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack</td>
<td>Tulsa Corpus</td>
<td>Parquet (1)</td>
<td>Nvdia Thrust (4)</td>
</tr>
<tr>
<td>Transformations</td>
<td>Texas A&amp;M Sparse Matrix</td>
<td>MKL (3)</td>
<td>Nvdia outparus (4)</td>
</tr>
<tr>
<td>Histogram</td>
<td>RDU Accident and Crime Report</td>
<td>GSL Histogram (2)</td>
<td>Nvdia Cub (4)</td>
</tr>
<tr>
<td>Query</td>
<td>RDU Accident and Crime Report</td>
<td>Pandas (5)</td>
<td>Rapids AI (6)</td>
</tr>
</tbody>
</table>

CPU
- 2x Intel Xeon ES-2630 2.2GHz

GPU
- NVIDIA TITAN XP 12GB, 305Mx

OS/CUDA
Ubuntu 18.04, CUDA toolkit 11.7

Conclusion and Future Work
- We demonstrated a method to generate efficient GPU code implementing data transformation tasks expressed using PDT.
- Future work encompasses more compiler optimizations to further improve performance.

References and Acknowledgement
- This work was supported by National Science Foundation awards CNS-1812727 and CCF-1907863.