ASPN: A Scalable In-SRAM Architecture for Pushdown Automata

Kevin Angstadt1, Arun Subramaniyan1, Elaheh Sadredini1, Reza Rahimi1
Kevin Skadron1, Westley Weimer1, Reetuparna Das1
1University of Michigan
{angstadt, arunsub, weimerw, reetudas} @umich.edu
1University of Virginia
{elaheh, rahimi, skadron} @virginia.edu

Problem
- Processing tree-structured or recursively-nested data (e.g., parsing and tree mining) are common data analyses
- Data sets continue to grow in size, as does the demand for real-time analyses
- Existing CPU-based data parsers exhibit high branching and low data reuse
- Automata Processing and Regular Expression Acceleration have aided other big data analyses
- Limited expressive power does not support recognizing recursively-nested data
- Goal: scalable and high-performance techniques for processing data to keep up with industrial demand

Our Approach: ASPEN
- We developed the Accelerated In-SRAM Pushdown ENgine
  - Scalable processing engine that uses LLC slices to accelerate Pushdown Automata computation
  - Custom five-stage datapath using SRAM lookups can process up to one token per cycle
- We support existing grammars for data parsers and hand-crafted PDAs with a custom compiler
  - Parsers transformed to homogeneous deterministic pushdown automata for execution on ASPEN
  - Optimizations improve hardware resource usage and reduce number of processing stalls

Advantages of LLC
- Supports tight coupling with CPU for processing that is part of a workflow
- LLC supports SRAM lookup operations needed in ASPEN datapath
- High clock frequency supports high data throughputs
- ASPEN provides additional cache when not in use

Compiler Optimizations

Epsilon Merging
- \([A-Z] \times \text{Pop 0} \\text{No Push}\)
- \([A-Z] \times \text{Pop 1} \\text{Push 'a'}\)
- \(\varepsilon \times \text{Pop 0} \text{No Push}\)

Multipop
- \(\varepsilon \times \text{Pop 1} \text{No Push}\)
- \(\varepsilon \times \text{Pop 1} \text{No Push}\)
- \(\varepsilon \times \text{Pop 1} \text{No Push}\)
- \(\varepsilon \times \text{Pop 1} \text{No Push}\)
- \(\varepsilon \times \text{Pop 4} \text{No Push}\)

Experimental Performance Results

XML Parsing (Single Large DPDA)
- ASPEN significantly outperforms CPU and GPU for large, complex trees

Subtree Mining (Many Small PDPA)
- ASPEN provides additional cache when not in use

Acknowledgements

This work is funded, in part, by the NSF (1763674, 1619098, CAREER-1652294 and CCF-1629450); Air Force (FA9550-17-2-0079); and CRISP, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.