ASPEN: A Scalable In-SRAM Architecture for Pushdown Automata

Kevin Angstadt† Arun Subramaniyan† Elaheh Sadredini‡ Reza Rahimi‡
Kevin Skadron‡ Westley Weimer† Reetuparna Das†

†University of Michigan
{angstadt, arunsub, weimerw, reetudas}@umich.edu

‡University of Virginia
{elaheh, rahimi, skadron}@virginia.edu

COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

UNIVERSITY of VIRGINIA

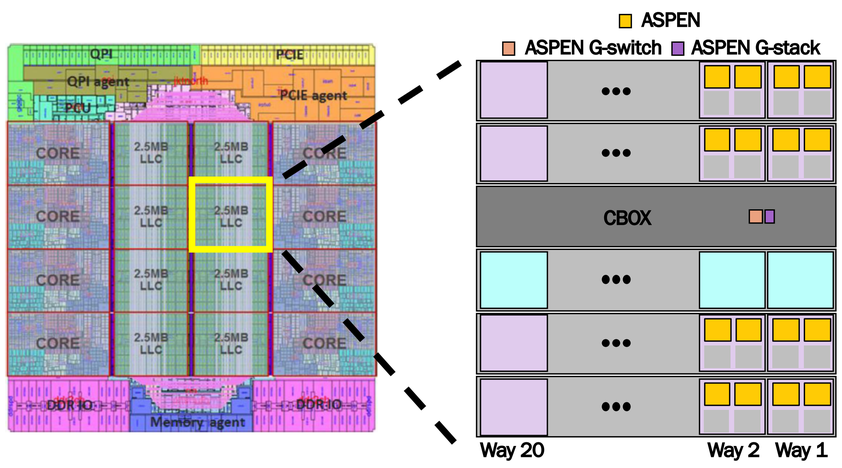
Problem

- 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 **A**ccelerated in-**S**RAM **P**ushdown **EN**gine
  - 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 DPDAs with a custom compiler
  - Parsers transformed to *homogeneous* deterministic pushdown automata for execution on ASPEN
  - Opimizations improve hardware resource usage and **reduce number of processing stalls**

Advantages of LLC

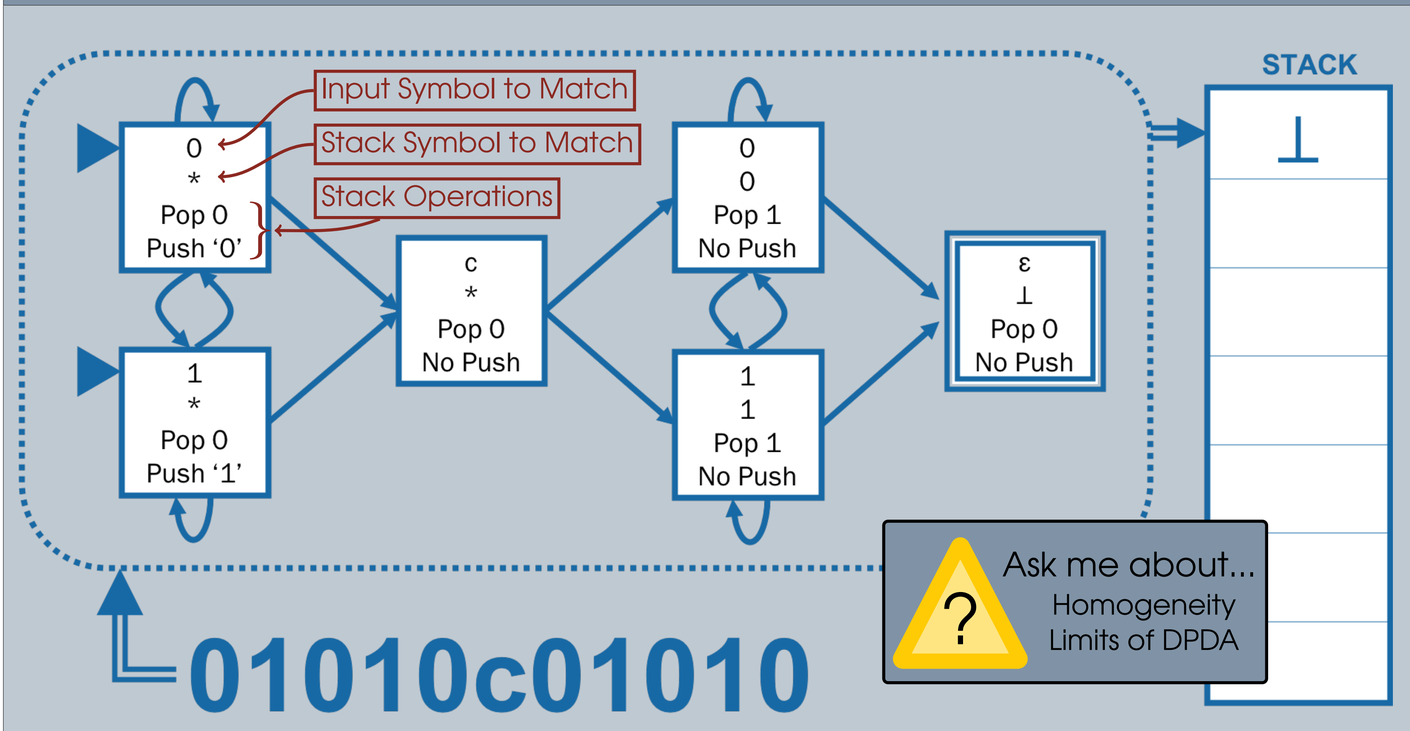


- Supports tight coupling with CPU for processing that is part of a workflow
- LLC supports SRAM lookup operations needed in ASPEN datapath
- High clock frequency supports high data throughputs
- ASPEN provides additional cache when not in use

Compiler Optimizations

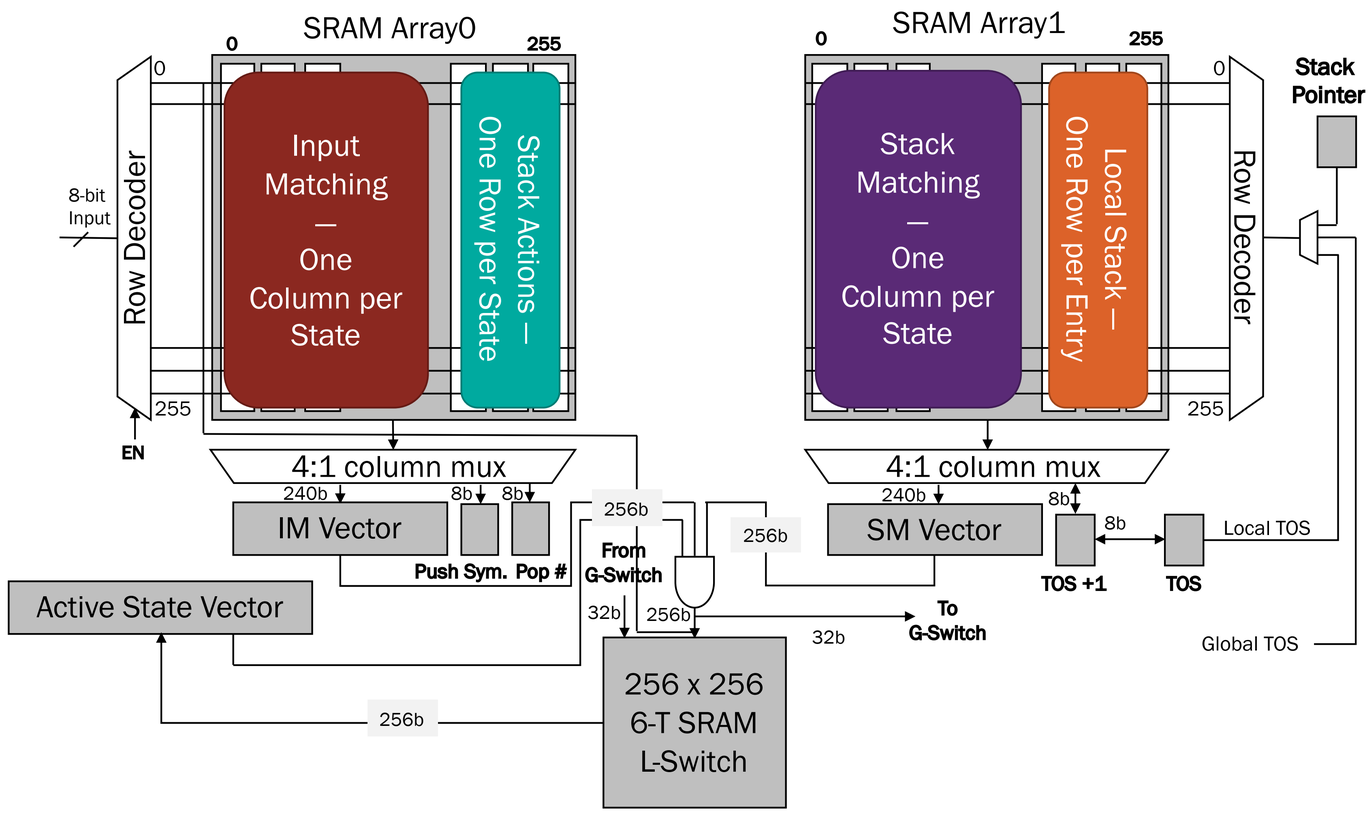
Result: Reduces epsilon states by 65% across four benchmarks

Background: Deterministic Pushdown Automata

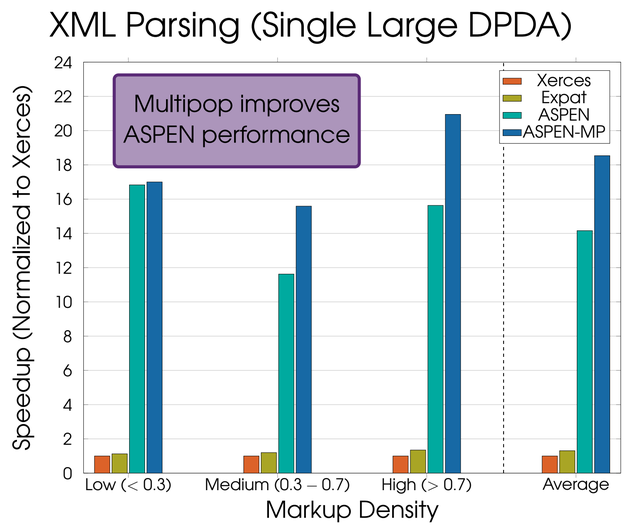


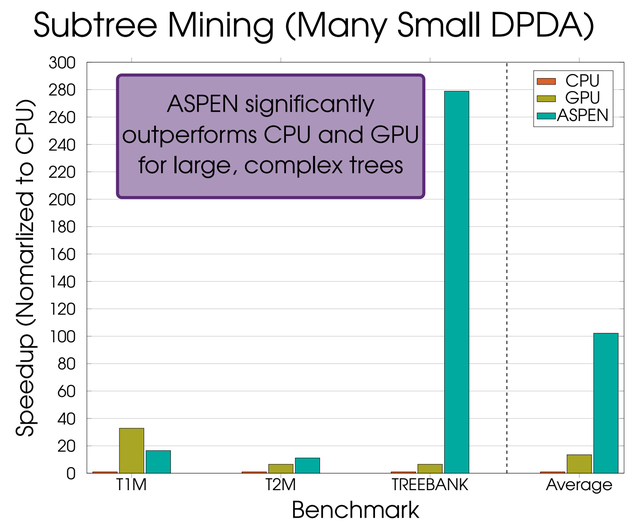
- Deterministic Pushdown Automata (DPDA) support a commonly-used subset of Context Free Grammars often used for defining languages
- DPDA consist of finite state control and a global stack memory
  - Transitions between states consider both the next input character and the top of the stack
  - Stack may be manipulated by pushing and popping symbols
  - Reports (matches) are generated when an accepting state becomes active

ASPEN Datapath: Two SRAM Arrays Support 240 States



Experimental Performance Results





Acknowledgements

This work is funded, in part, by the NSF (1763674, 1619098, CAREER-1652294 and CCF-1629450); Air Force (FA8750-17-2-0079); and CRISP, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.