RAPID: Accelerating Pattern Search Applications with Reconfigurable Hardware

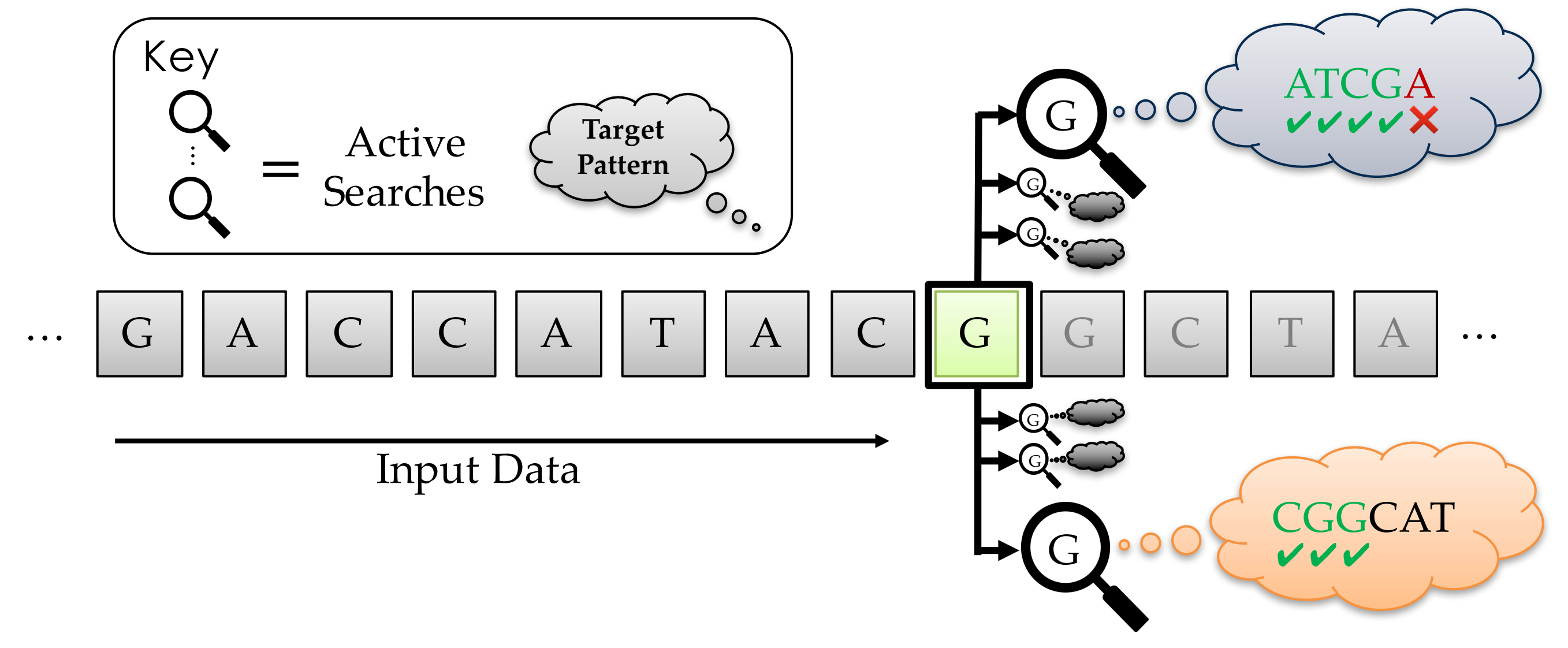
Kevin Angstadt   Jack Wadden   Xiaoping Huang[†]   Mohamed El-Hadedy[‡]   Westley Weimer   Kevin Skadron

University of Virginia, [†]Northwestern Polytechnical University, [‡]University of Illinois at Urbana-Champaign
{angstadt, wadden, weimer, skadron}@virginia.edu, huangxp@nwpu.edu.cn, hadedy@illinois.edu

UNIVERSITY of VIRGINIA
Department of Computer Science
Charlottesville, VA 22904

Problem

- Many big data applications reduce to highly parallel pattern search problems against a single stream of data
- Hardware accelerators, such as the Automata Processor and FPGAs, support these searches, but programming can be challenging
- **Goal:** A concise, maintainable programming model that maximizes the number of parallel searches for higher throughput



Our Approach: RAPID

- We developed RAPID, a high-level programming language for pattern-search algorithms
  - C-style language with added domain-specific parallel control structures
  - Suitable data structures for pattern search problems
  - Synchronized access to input data stream across active computation
- RAPID programs are compiled to finite automata via a recursive algorithm
- Finite automata are then synthesized for reconfigurable hardware (e.g. AP, FPGA)

Manipulating Automata: VASim

- Open-source, highly-flexible platform for automata engine and architecture research
- Common repository for automata optimizations, transformations, and static and dynamic analyses
- Multi-threaded, high-performance CPU simulation core
- Emits synthesizable Verilog for executing automata on FPGAs
  - Update activations every clock cycle (activations stored in registers)
  - Activate state if: (1) the state accepts the input symbol and (2) a state with an incident edge is active

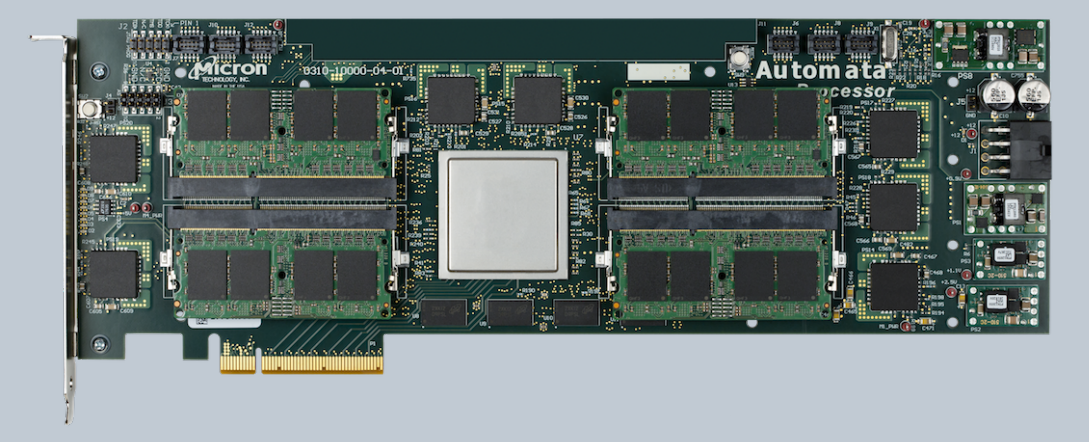
Acknowledgments

We acknowledge the partial support of the NSF (CCF-0954024, CCF-1116289, CDI-1124931); Air Force (FA8750-15-2-0075); Virginia Commonwealth Fellowship; Jefferson Scholars Foundation; the ARCS Foundation; and C-FAR, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA. We would like to thank Micron Technology for their support with programming the AP.

Background: Reconfigurable Hardware

Micron's Automata Processor

- Memory-derived hardware implementation of non-deterministic finite automata
  - States stored in memory arrray
  - Connections made with hierarchical, reconfigurable routing matrix
- Accelerates identification of patterns in data stream using massive parallelism



Field Programmable Gate Arrays

- Logic-based reconfigurable fabric of LUTs and Memory
- Allow custom implementation of applications for high-speed processing
- We use: Xilinx Kintex UltraScale XCKU060

Parallel Control Structures

Static Thread Spawning

Example: Search for 'and' and 'or' in parallel

```
either {
  'a' == input();
  'n' == input();
  'd' == input();
} orelse {
  'o' == input();
  'r' == input();
}
```

Dynamic Thread Spawning

Example: Search for all strings within a Hamming distance of 5 from any string in the `dna` array

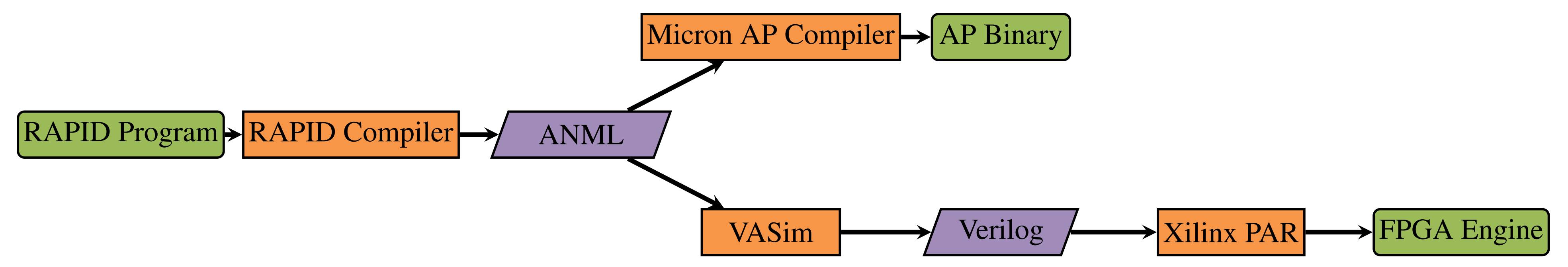
```
some(String s : dna) {
  hamming_distance(s,5);
}
```

Sliding Window Searches

Example: Find the word 'rapid' anywhere in the input stream

```
whenever(  ALL_INPUT == input() ) {
  foreach(char c : "rapid")
    c == input();
  report;
}
```

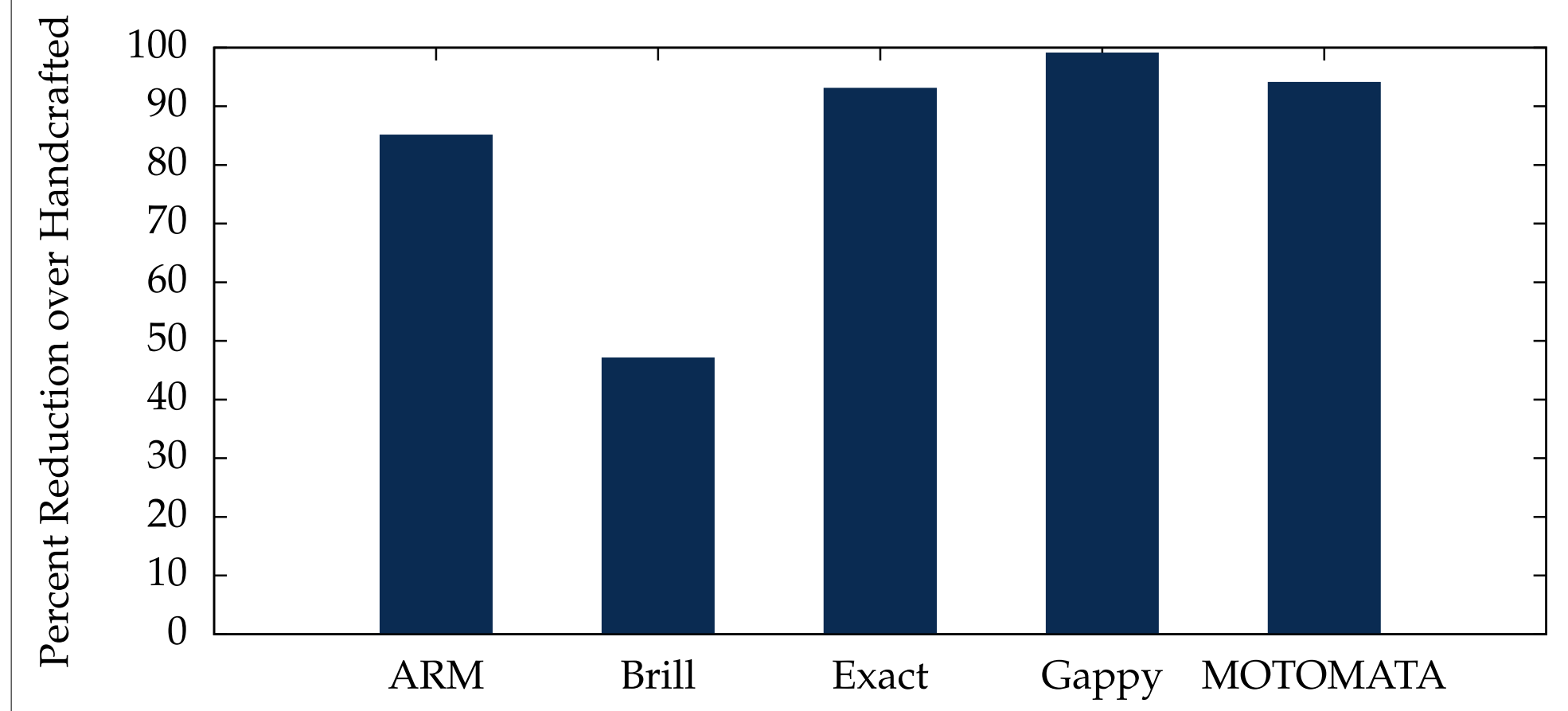
Tool Pipeline



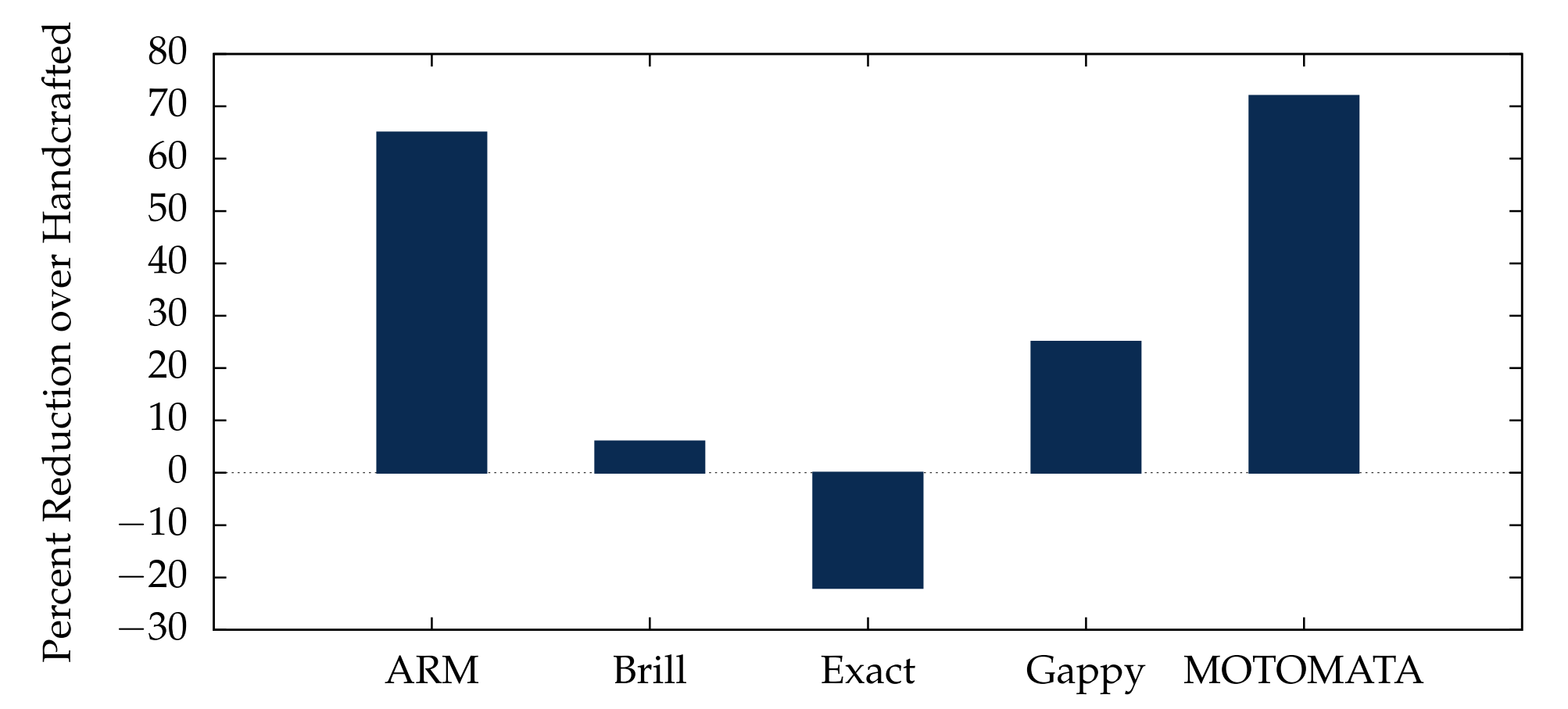
Experimental Results

Benchmarks: Association Rule Mining (*ARM*), *Brill* POS Tagging, *Exact* DNA Alignment, *Gappy* DNA Alignment, Planted Motif Search (*MOTOMATA*)

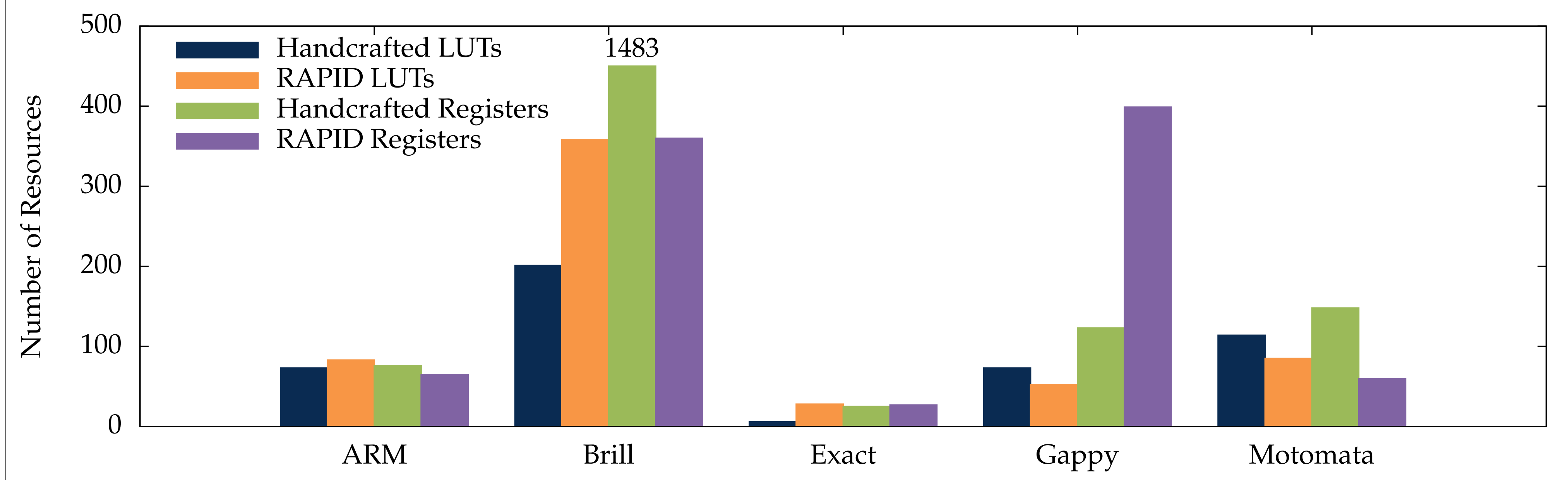
Lines of Code



State Usage* (Automata Processor)



LUT and Register Usage* (FPGA)



***Note:** Reducing resource usage (states, LUTs, and registers) allows for more active searches to be executed in parallel on reconfigurable hardware. This improves throughput and overall performance.

Conclusions

- RAPID reduces program text by 92%–98%
- RAPID programs are just as fast as their hand-crafted counterparts
- RAPID programs require no modifications to execute efficiently on the AP and FPGAs

Technology Transfer

Industry Collaboration

- Contacts with both Micron and Xilinx
- Center for Automata Processing at UVA brings together researchers and industry experts
- New features in AP runtime driver code as a result of this research

Publications and Presentations

- Associated work published and presented at ASPLOS 2016 and Supercomputing Frontiers 2016
- Weekly/semiweekly teleconferences with Micron and Xilinx to present research