UNIVERSITY of VIRGINIA

CRISP
Center for Research on Intelligent
Storage and Processing in Memory

SRC®
Semiconductor
Research
Corporation

A Scalable and Efficient in-Memory Interconnect Architecture for Automata Processing

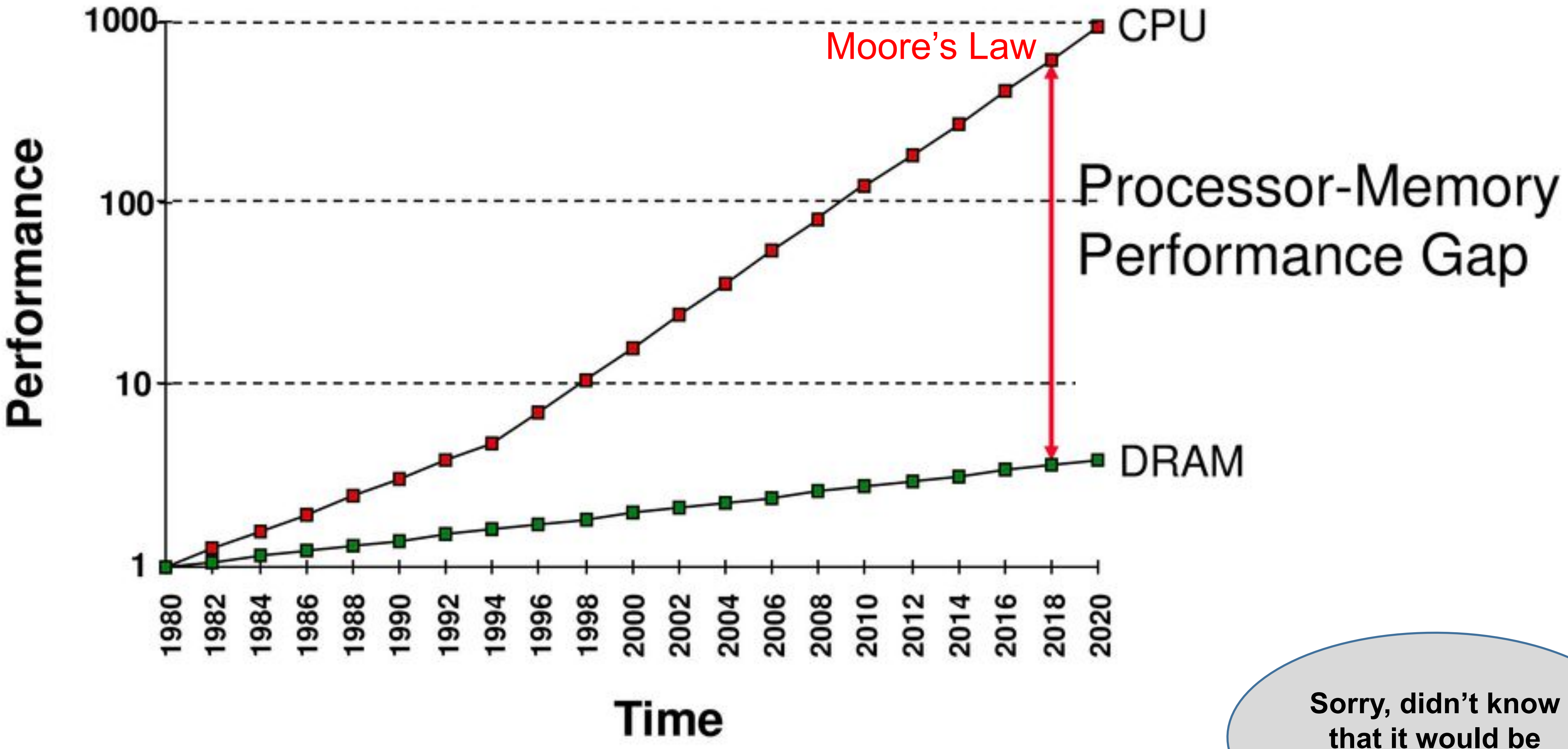
Elaheh Sadredini, Reza Rahimi, Vaibhav Verma, Mircea Stan, Kevin Skadron

University of Virginia

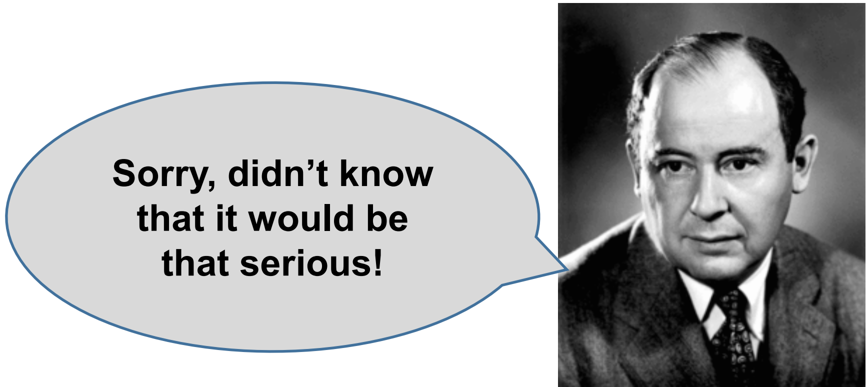
elaheh@virginia.edu

Task: 2780.001

Processor / Memory Performance Gap

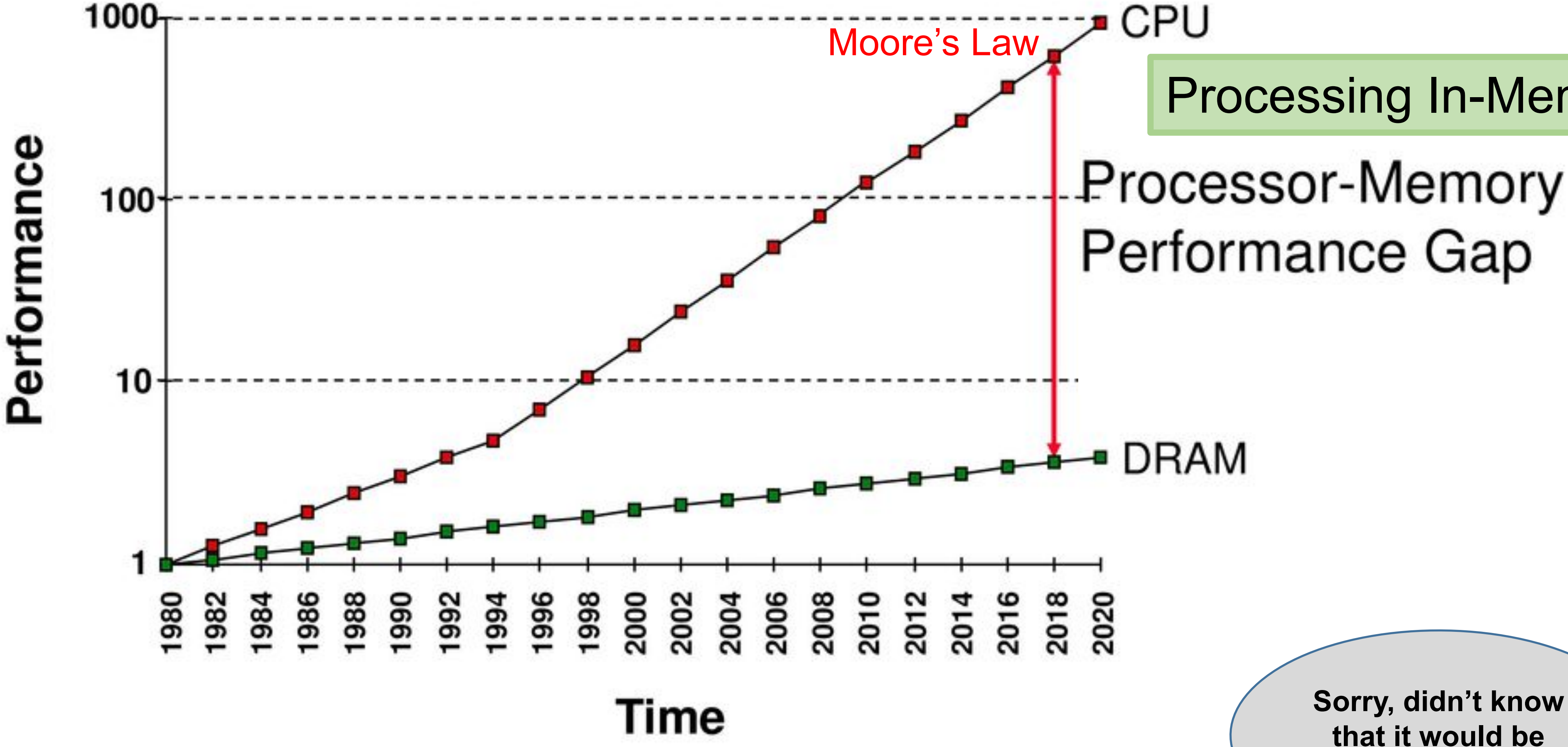


Source: David Patterson, UC Berkeley



Task: 2780.001

Processor / Memory Performance Gap



Processing In-Memory (PIM)

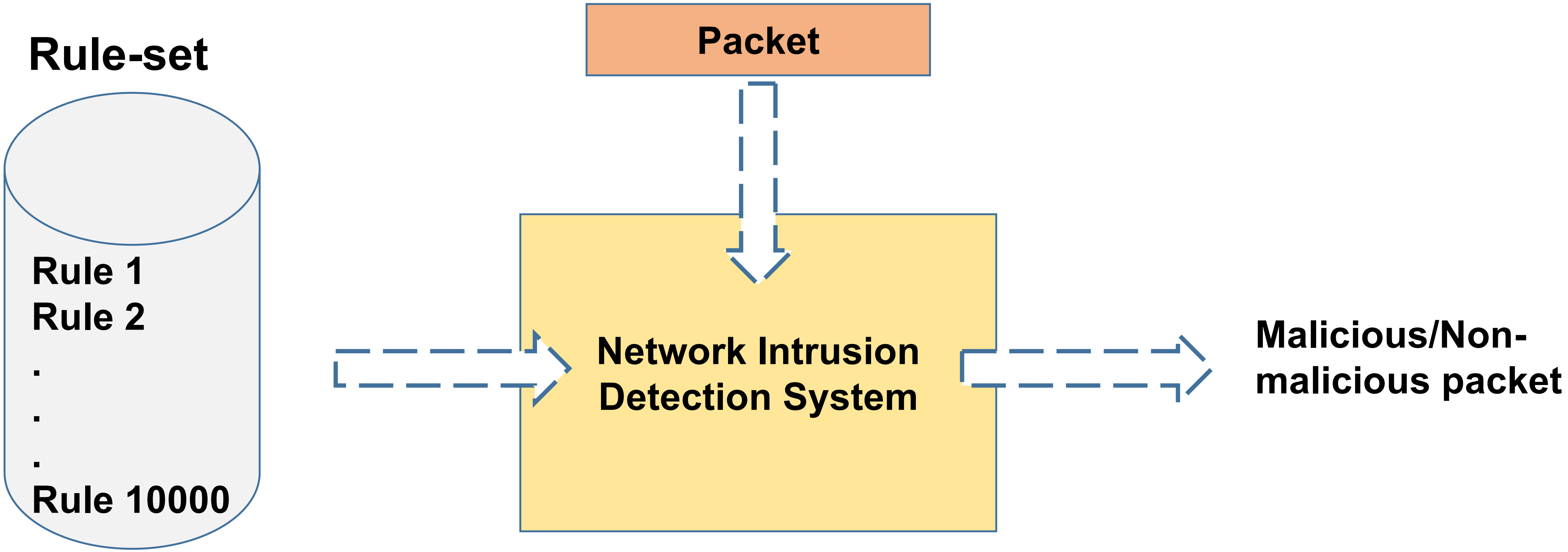
Sorry, didn't know that it would be that serious!

Task: 2780.001

Source: David Patterson, UC Berkeley

Scalable and High-Performance Techniques Are Needed for Pattern Processing

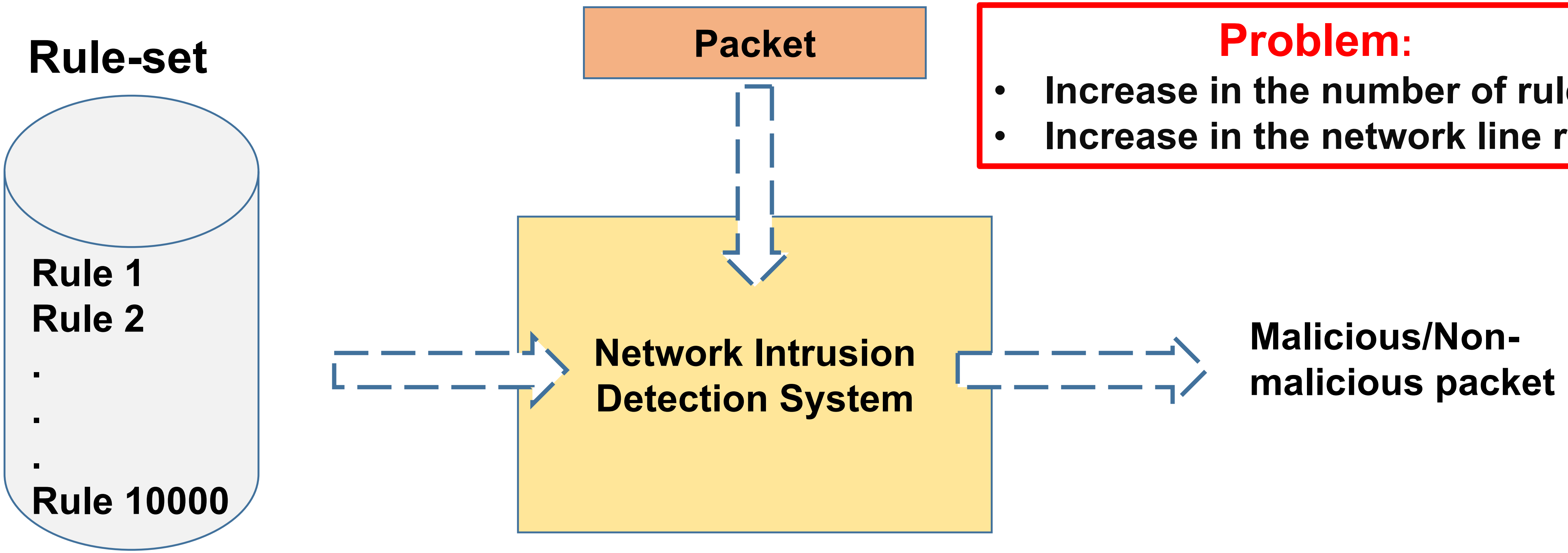
- Incoming packet is checked against every single rule of the database


Rule-set
Rule 1
Rule 2
.
.
.
Rule 10000
Packet
Network Intrusion Detection System
Malicious/Non-malicious packet

Task: 2780.001

Scalable and High-Performance Techniques Are Needed for Pattern Processing

- Incoming packet is checked against every single rule of the database


Rule-set
Rule 1
Rule 2
.
.
.
Rule 10000
Packet
Network Intrusion Detection System
Malicious/Non-malicious packet

Problem:

- Increase in the number of rules
- Increase in the network line rate

Task: 2780.001

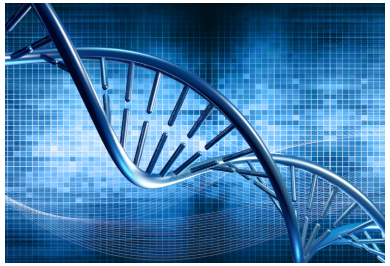
Pattern Recognition Importance

Network security

Pattern Recognition Importance

Network security

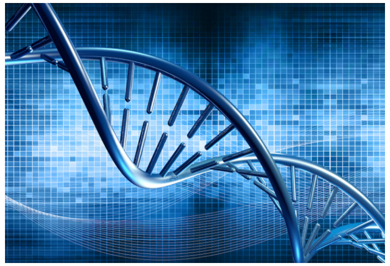
Bioinformatics



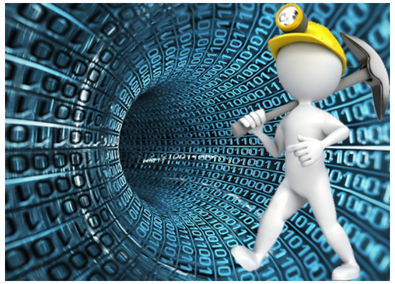
Pattern Recognition Importance

Network security

Bioinformatics



Data mining

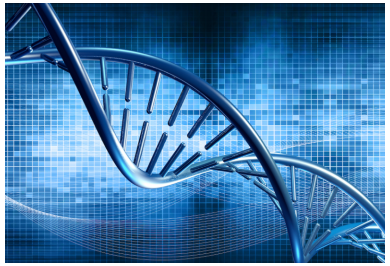


Task: 2780.001

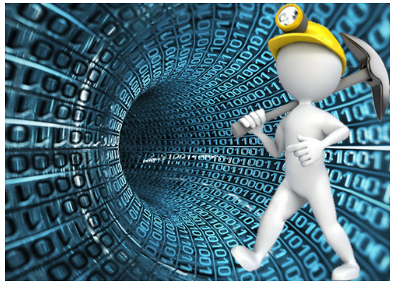
Pattern Recognition Importance

Network security

Bioinformatics



Data mining

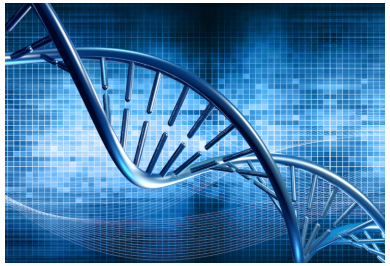


NLP

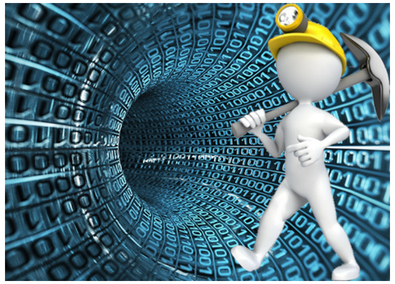
Pattern Recognition Importance

Network security

Bioinformatics



Data mining



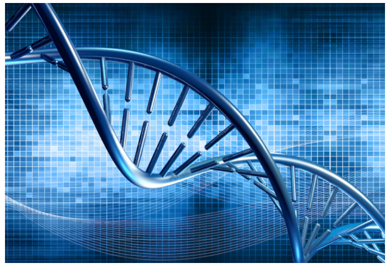
NLP

Patterns are often complex

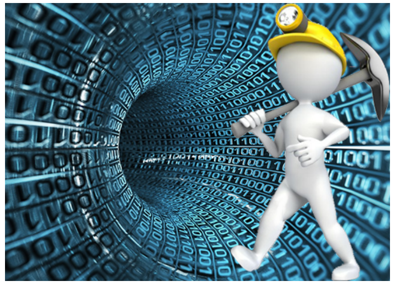
Pattern Recognition Importance

Network security

Bioinformatics



Data mining



NLP

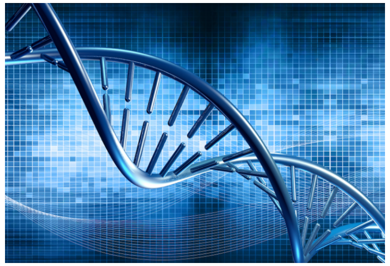
Patterns are often complex

Thousands of patterns need to be processed in parallel

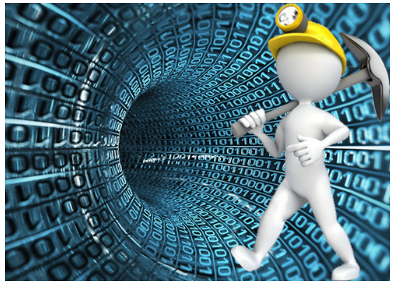
Pattern Recognition Importance

Network security

Bioinformatics



Data mining



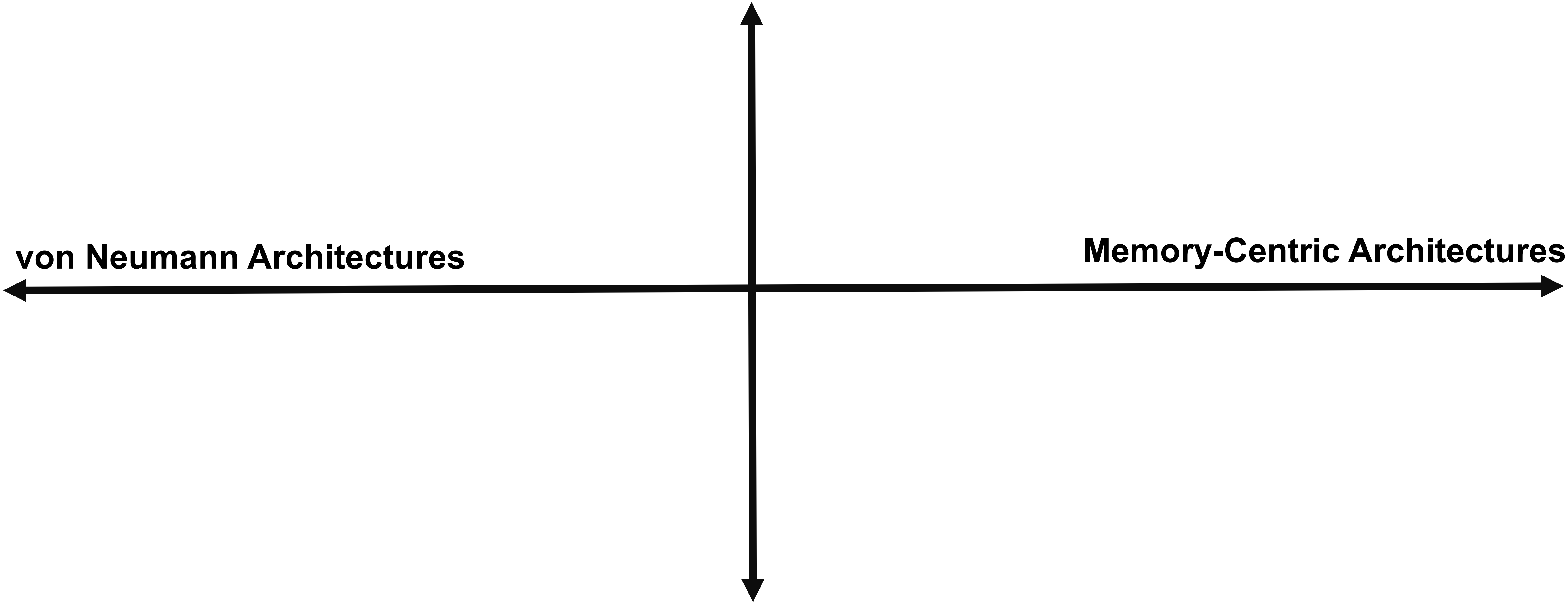
NLP

Patterns are often complex

Thousands of patterns need to be processed in parallel

Regular Expressions = **Finite Automata**

Existing Automata Processing Solution



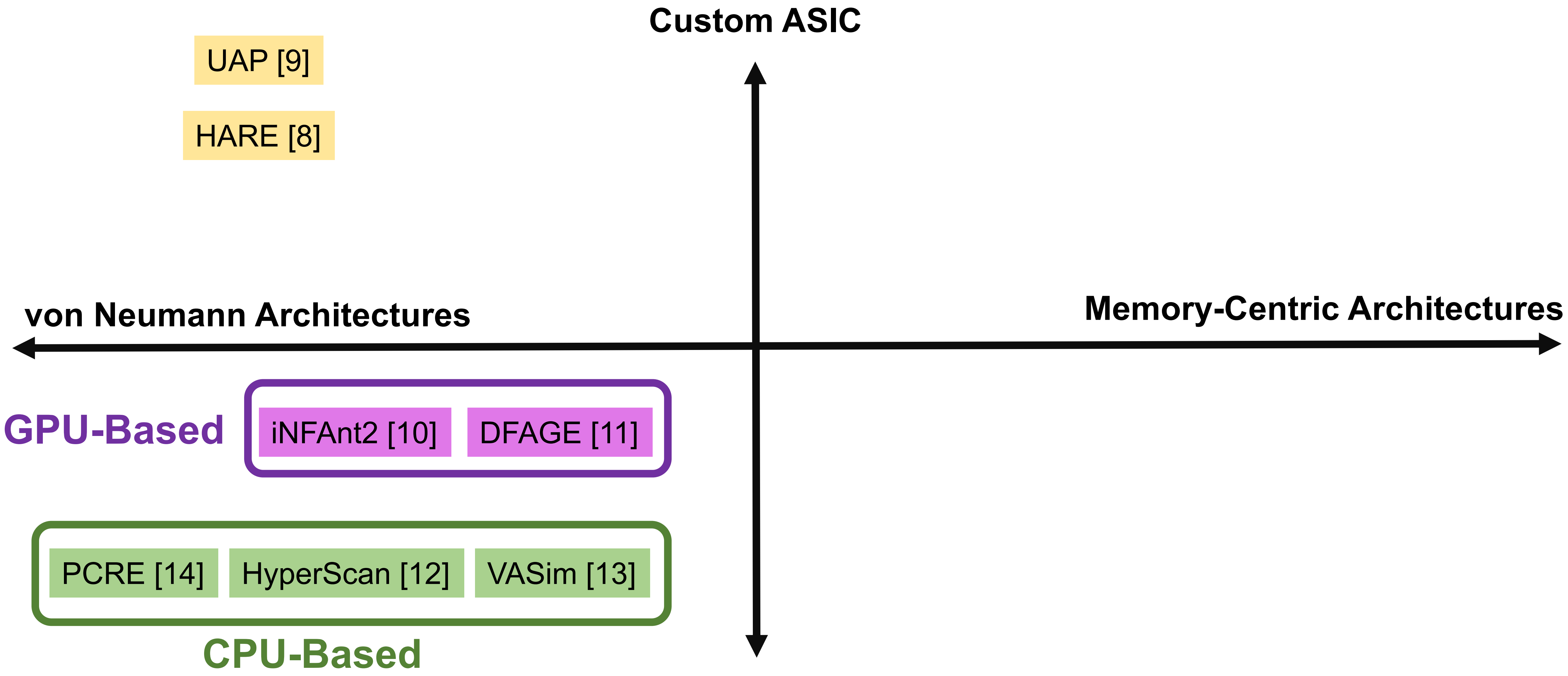
Custom ASIC

von Neumann Architectures

Memory-Centric Architectures

Reconfigurable SW/HW Engines

Existing Automata Processing Platforms

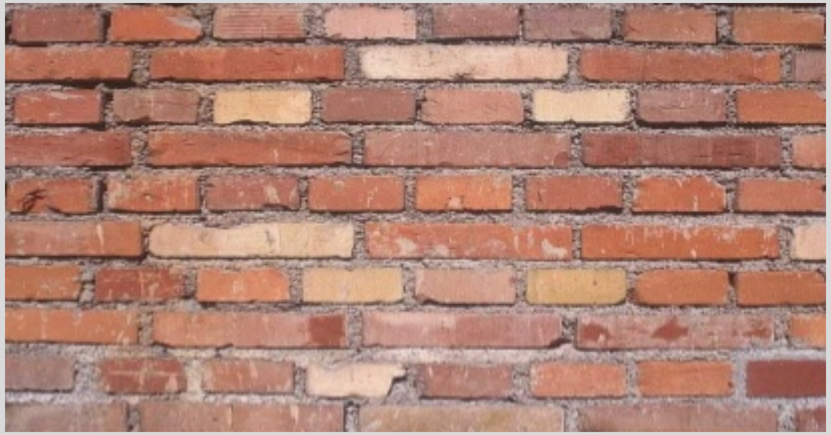


Existing Automata Processing Platforms

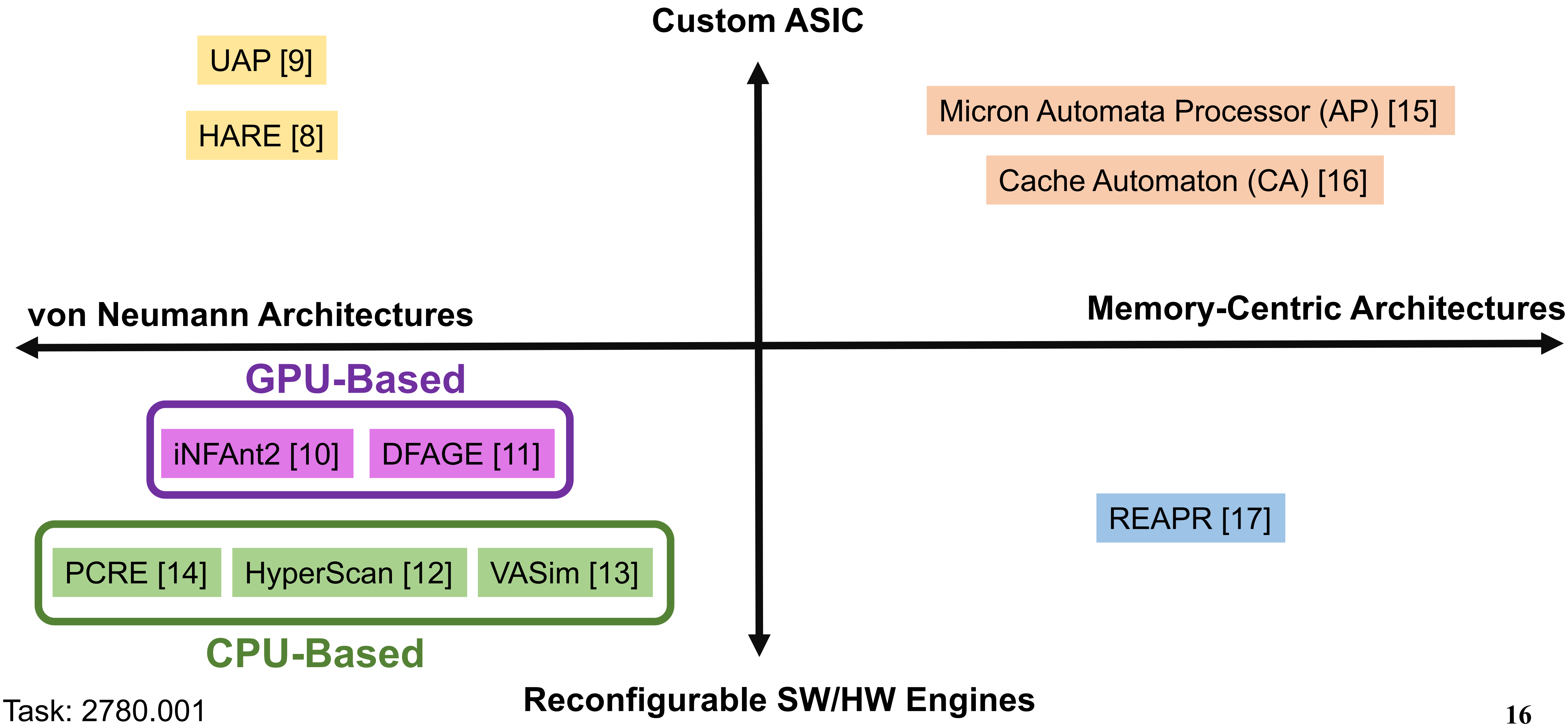
Custom ASIC

Problem: von Neumann processors easily become **memory bound**

- Unpredictable behavior
  Branch mispredictions
- Irregular access pattern
  Cache-miss
- Many parallel state transitions
  Saturate memory bandwidth



Existing Automata Processing Platforms



Custom ASIC

UAP [9]

HARE [8]

Micron Automata Processor (AP) [15]

Cache Automaton (CA) [16]

von Neumann Architectures

Memory-Centric Architectures

GPU-Based

iNFAnt2 [10]

DFAGE [11]

PCRE [14]

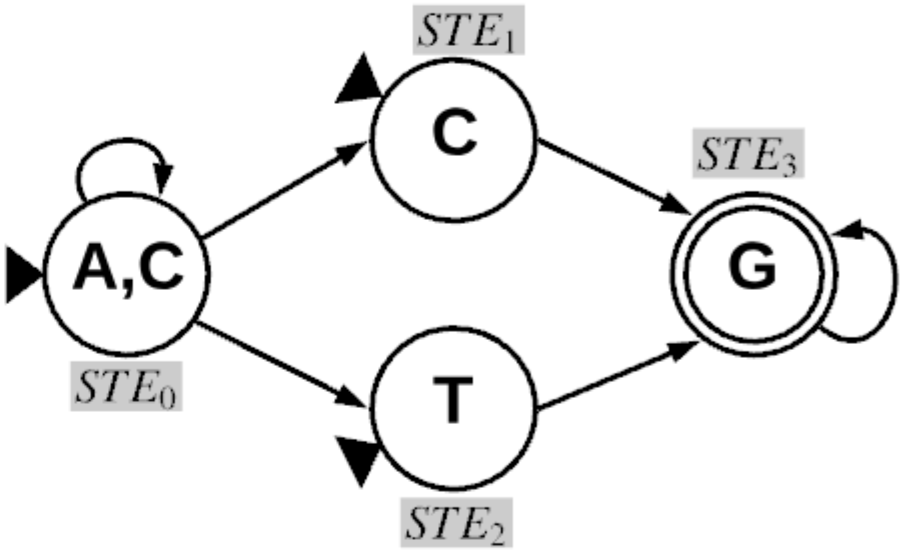
HyperScan [12]

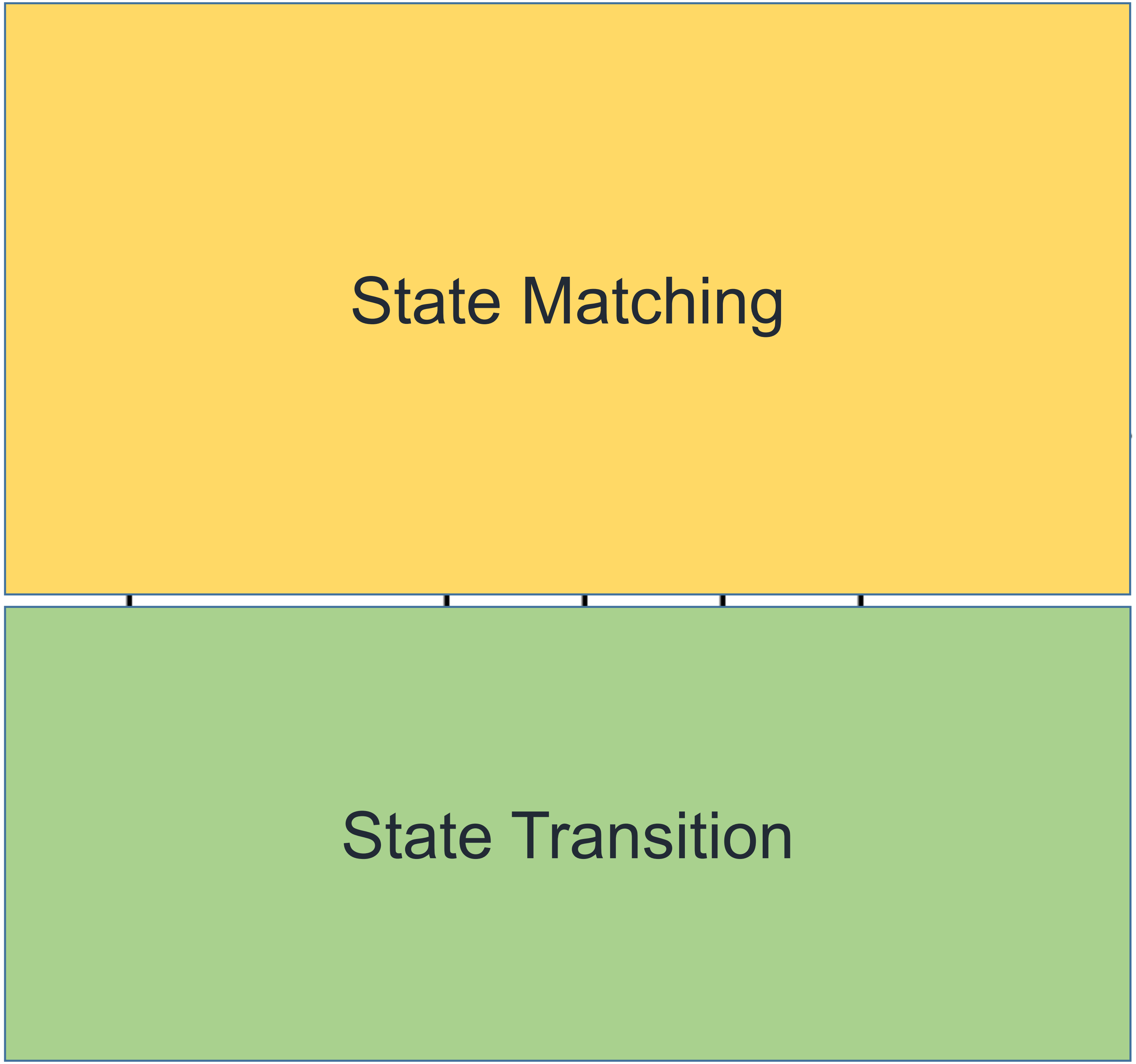
VASim [13]

CPU-Based

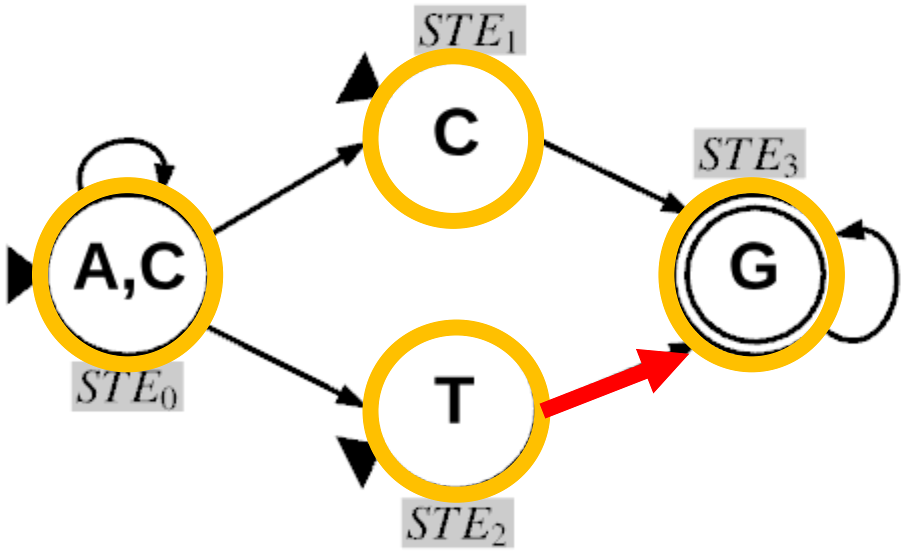
REAPR [17]

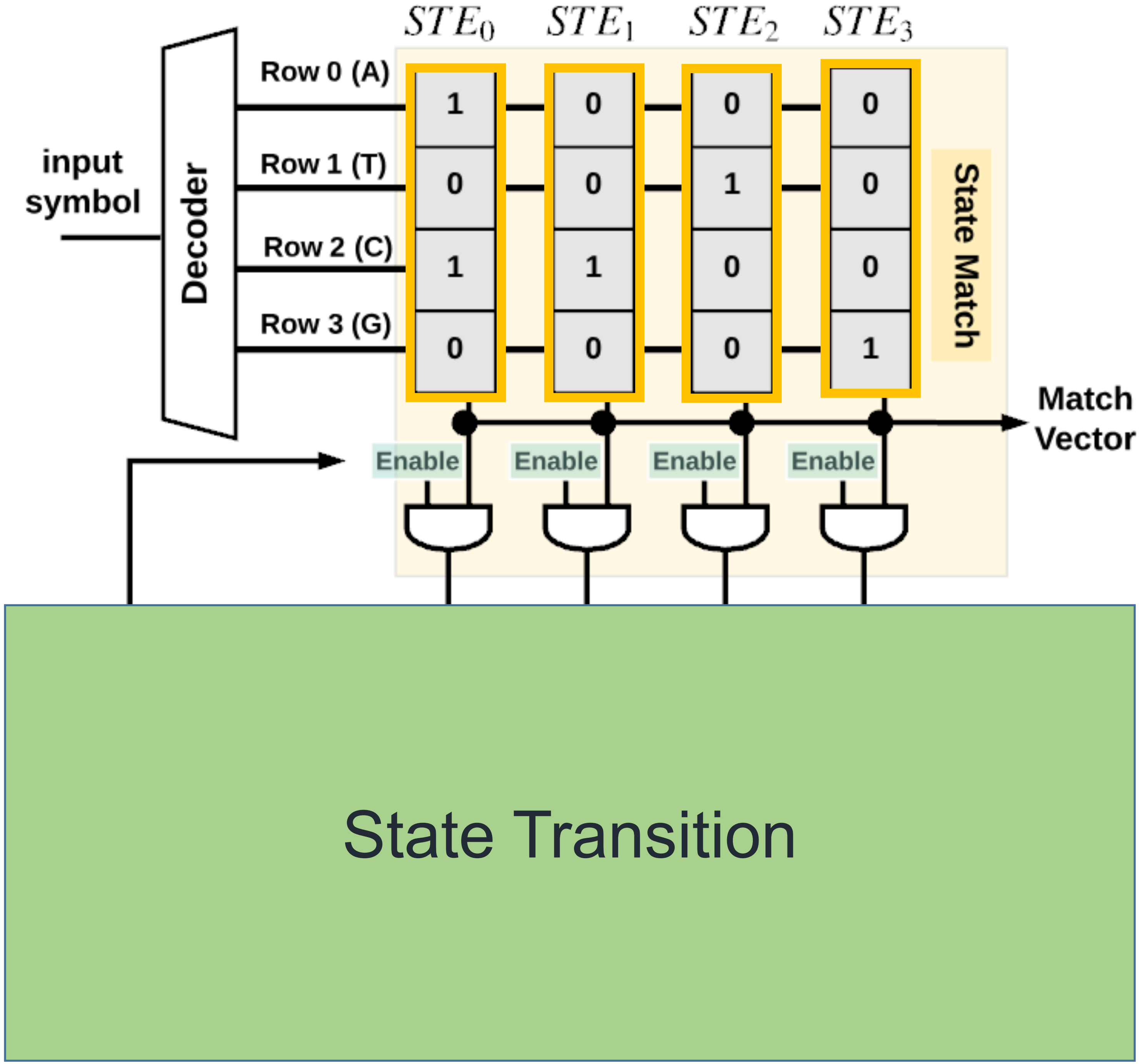
Reconfigurable SW/HW Engines




State Matching

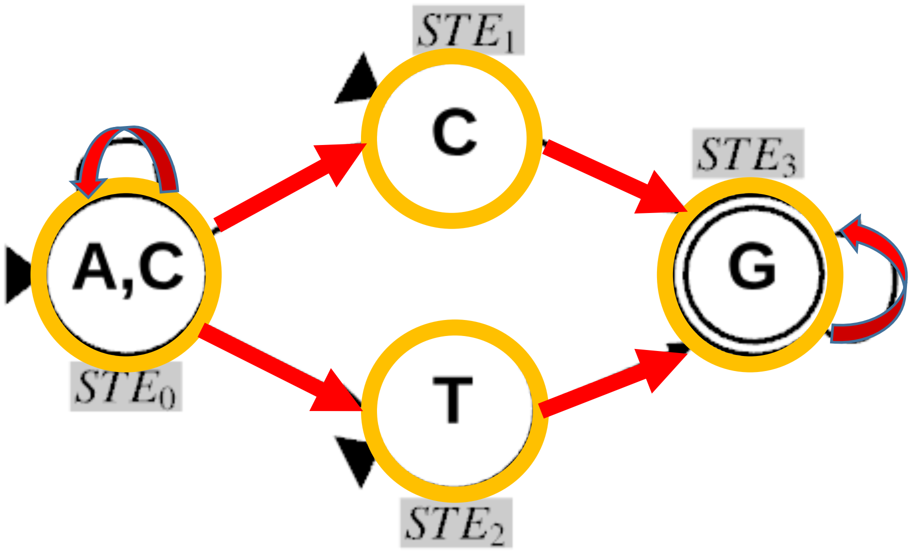
State Transition

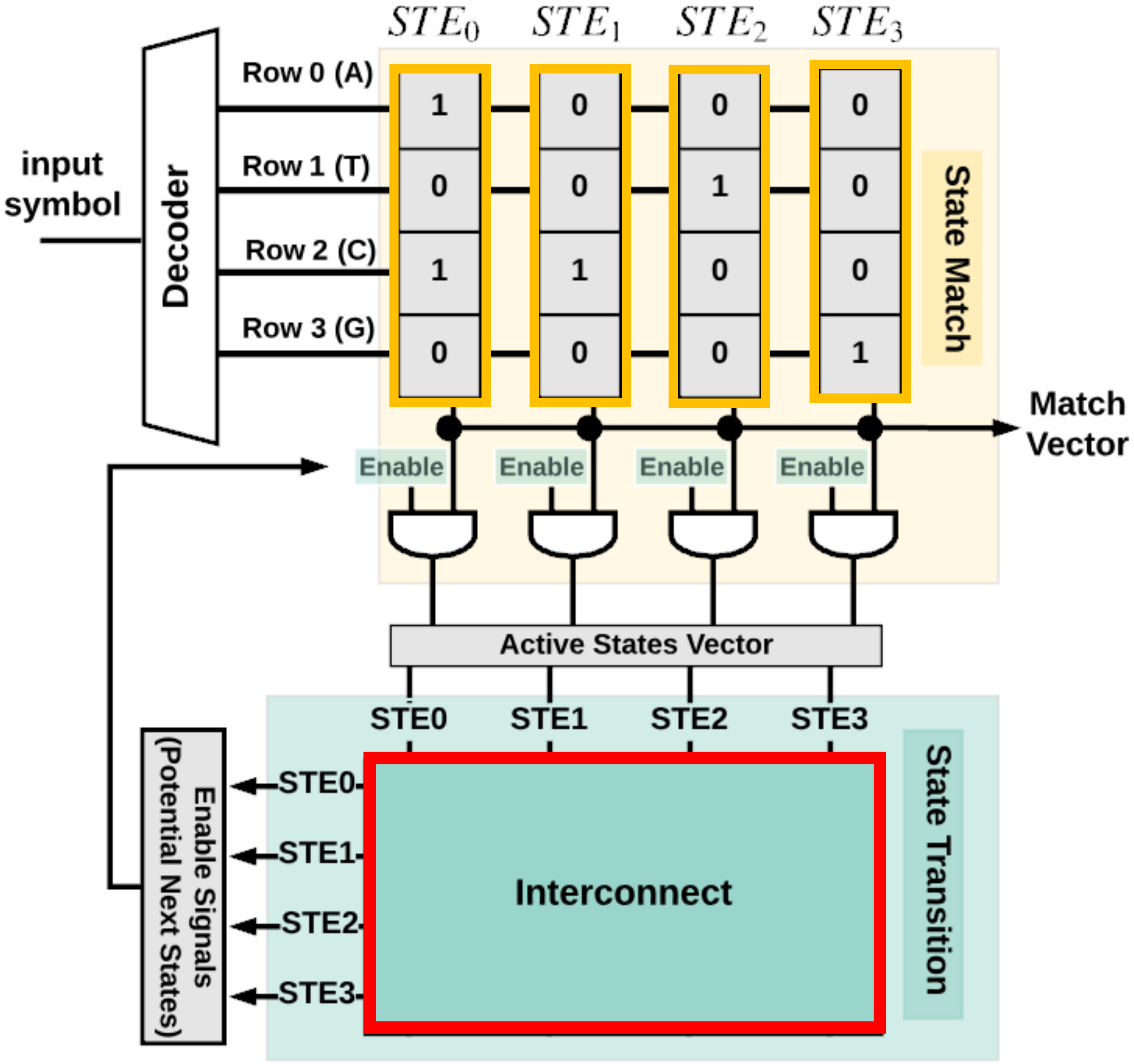




	STE_0	STE_1	STE_2	STE_3
Row 0 (A)	1	0	0	0
Row 1 (T)	0	0	1	0
Row 2 (C)	1	1	0	0
Row 3 (G)	0	0	0	1

State Transition



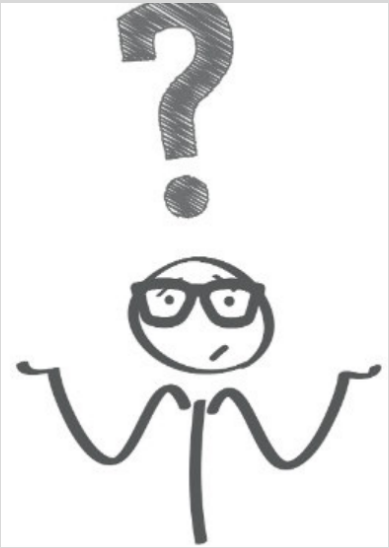


	STE_0	STE_1	STE_2	STE_3
Row 0 (A)	1	0	0	0
Row 1 (T)	0	0	1	0
Row 2 (C)	1	1	0	0
Row 3 (G)	0	0	0	1

Problems: interconnect inefficiency in the existing memory-centric architectures

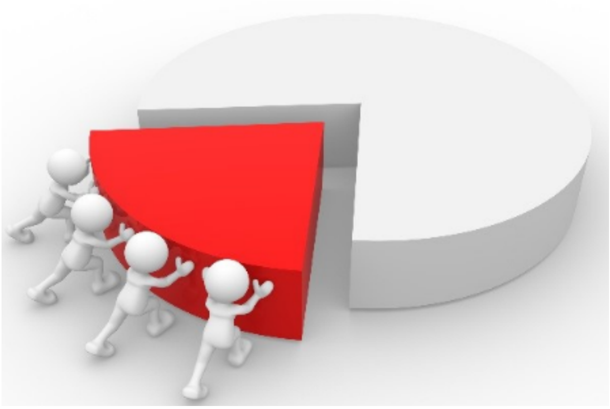
- Automata Processor [15]
  - Routing matrix congestion
  - **13%** state utilization for applications with complex routing!

- Cache Automaton [16]
  - Full-crossbar is excessive for interconnect
  - On average, only **0.53%** of switches are utilized!

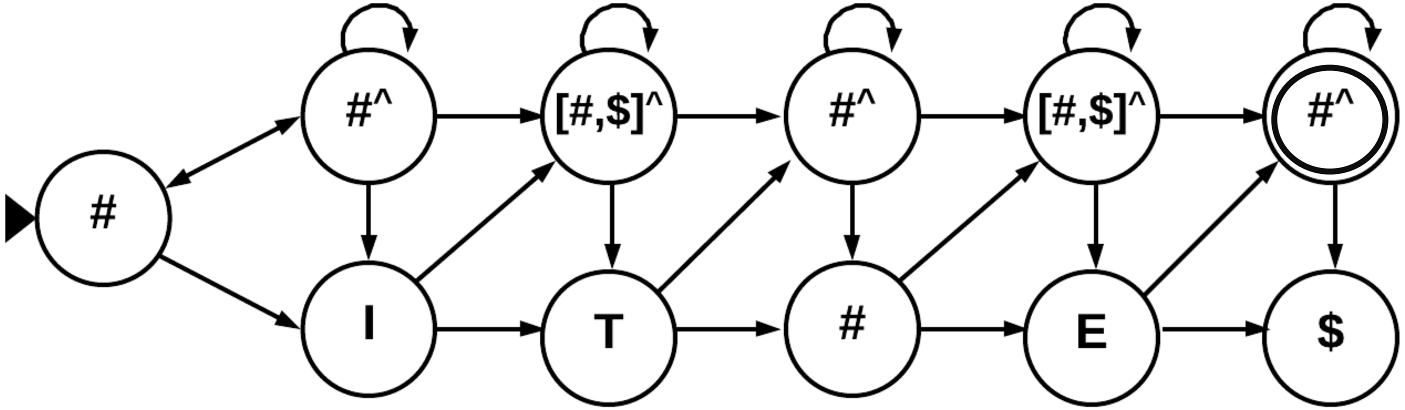


Main Contribution:

Designing a **low-overhead, yet flexible** routing architecture for automata processing and mapping it to a **right memory technology**



Full-Crossbar interconnect

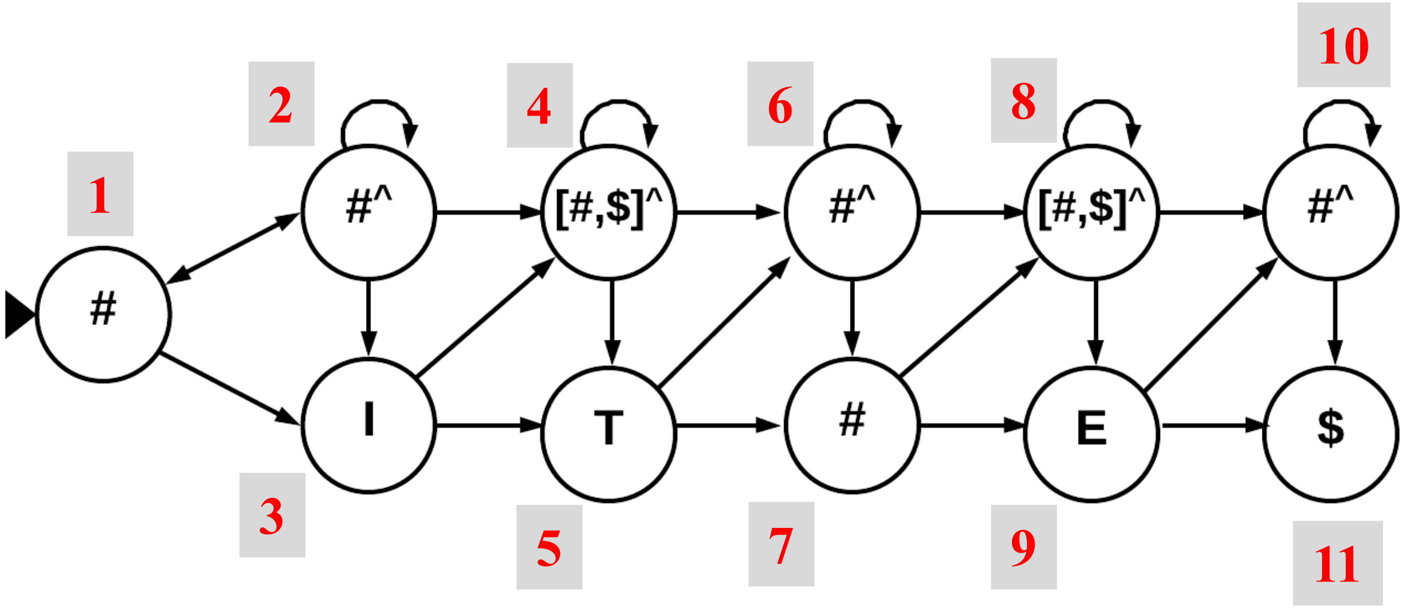


An Example Automaton

Task: 2780.001

Full-Crossbar interconnect

BFS Labeling



An Example Automaton

Full-Crossbar interconnect

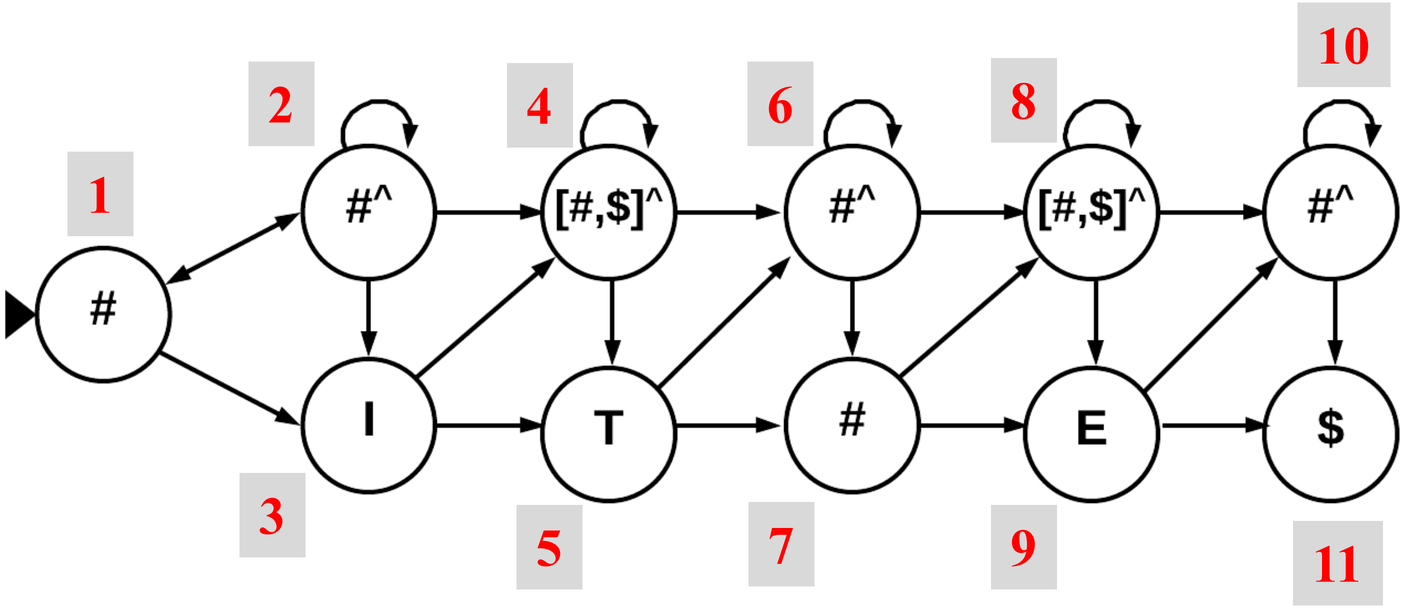
Connectivity Matrix

Source State

	1	2	3	4	5	6	7	8	9	10	11
11											
10											
9											
8											
7											
6											
5											
4											
3											
2											
1											

Destination State

BFS Labeling



An Example Automaton

Task: 2780.001

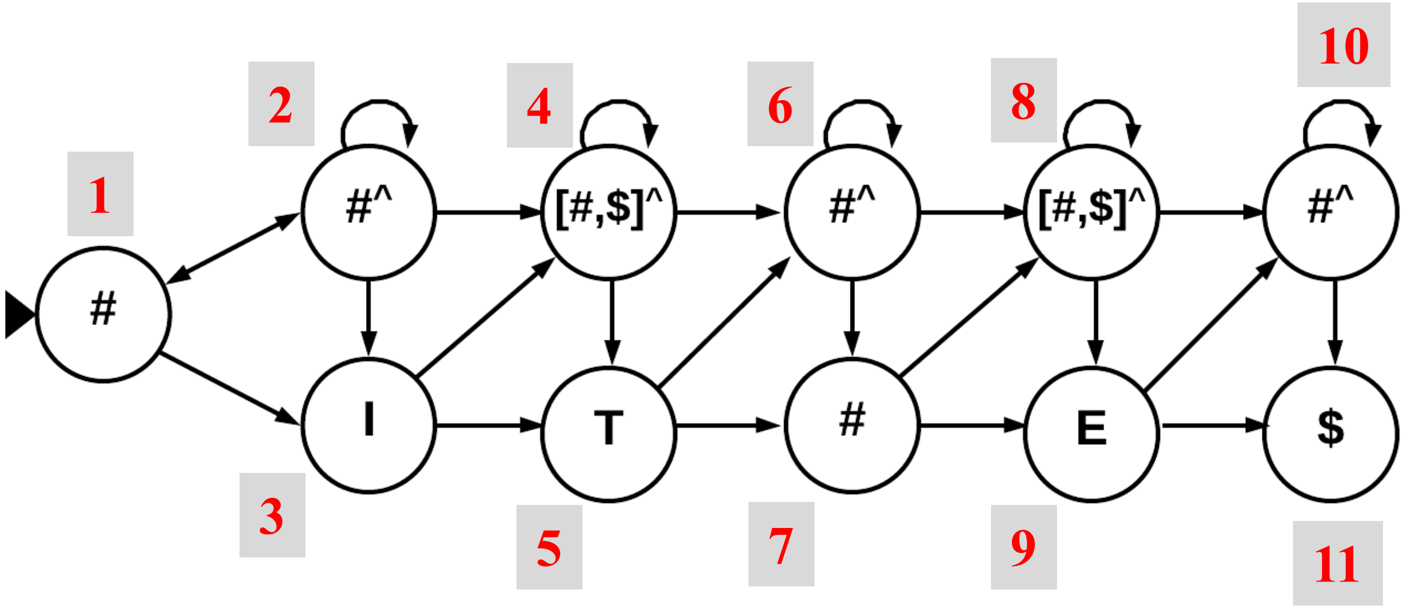
Solution: Minimizing Full-Crossbar

Connectivity Matrix

Source State											
11											
10										■	■
9										■	■
8								■	■	■	
7								■	■		
6						■	■	■			
5						■	■				
4				■	■	■					
3				■	■						
2	■	■		■							
1		■	■								
	1	2	3	4	5	6	7	8	9	10	11

Destination State

BFS Labeling

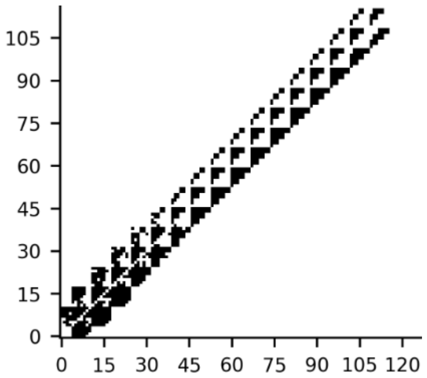


An Example Automaton

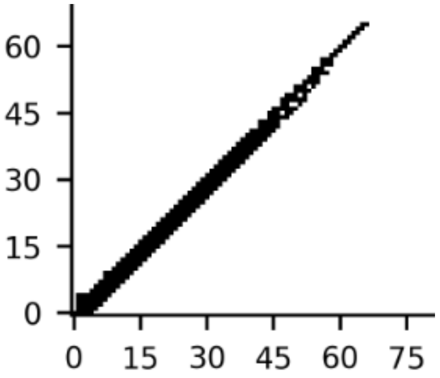
Task: 2780.001

Observation: Union Heatmap of Routing Switches with BFS Labeling

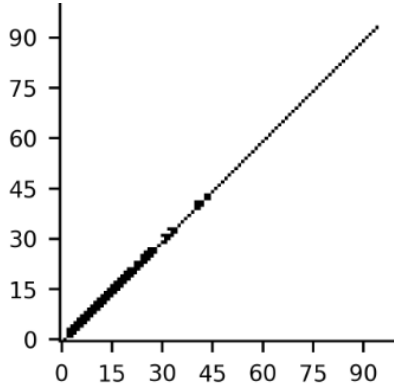
- 17 out of 19 benchmark applications show **diagonal property**



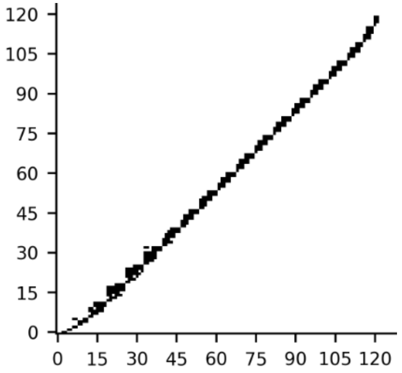
Levenshtein



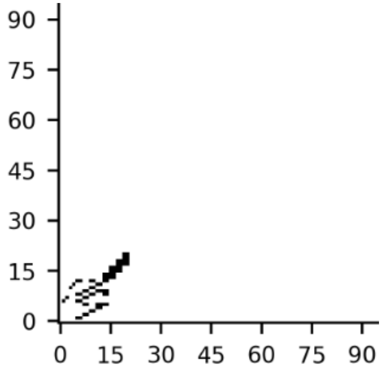
Brill



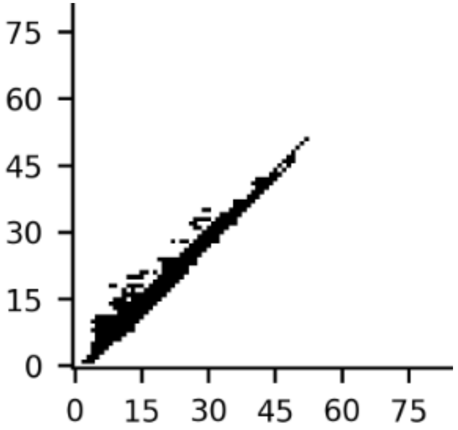
Dotstar



Hamming

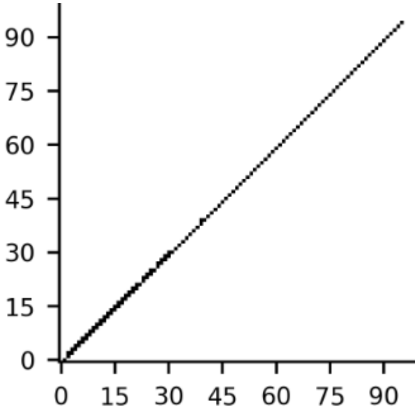


SPM

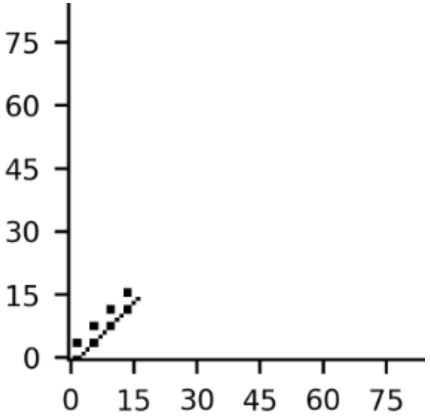


Task:

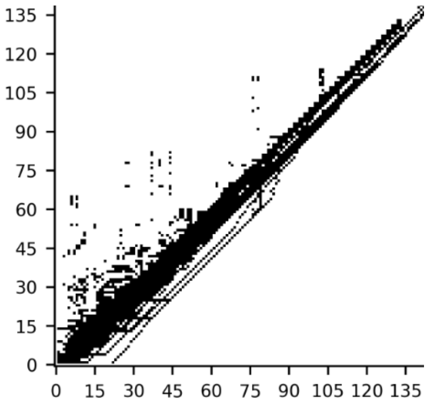
PowerEN



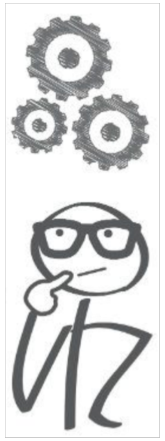
Ranges



Fermi

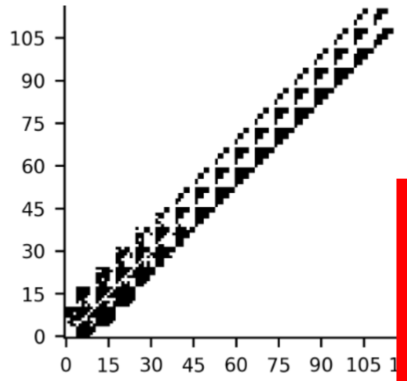


Snort

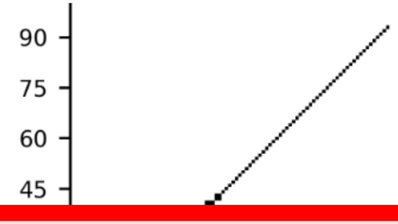


Observation: Union Heatmap of Routing Switches with BFS Labeling

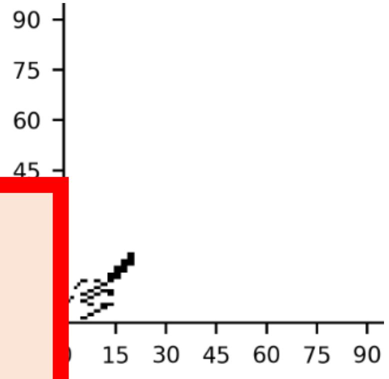
- 17 out of 19 benchmark applications show **diagonal property**



Levenshtein

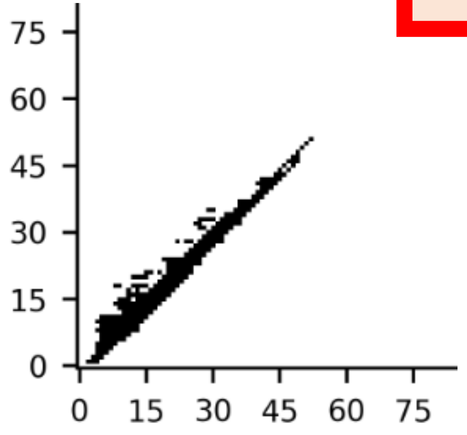


120
105
90
75
60

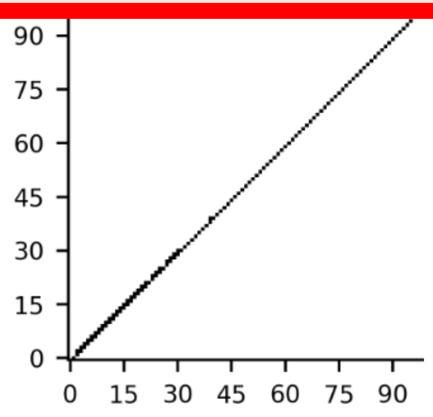


SPM

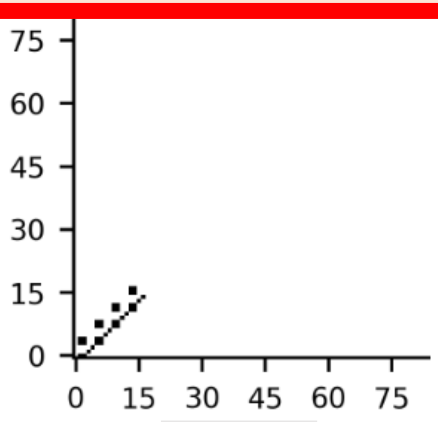
Only less than 1% of the switch cells are utilized!



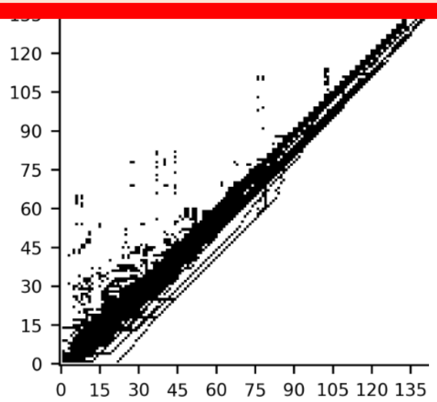
Task: **PowerEN**



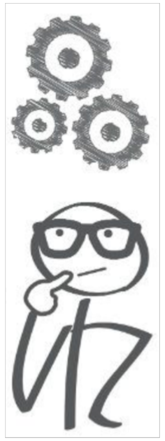
Ranges



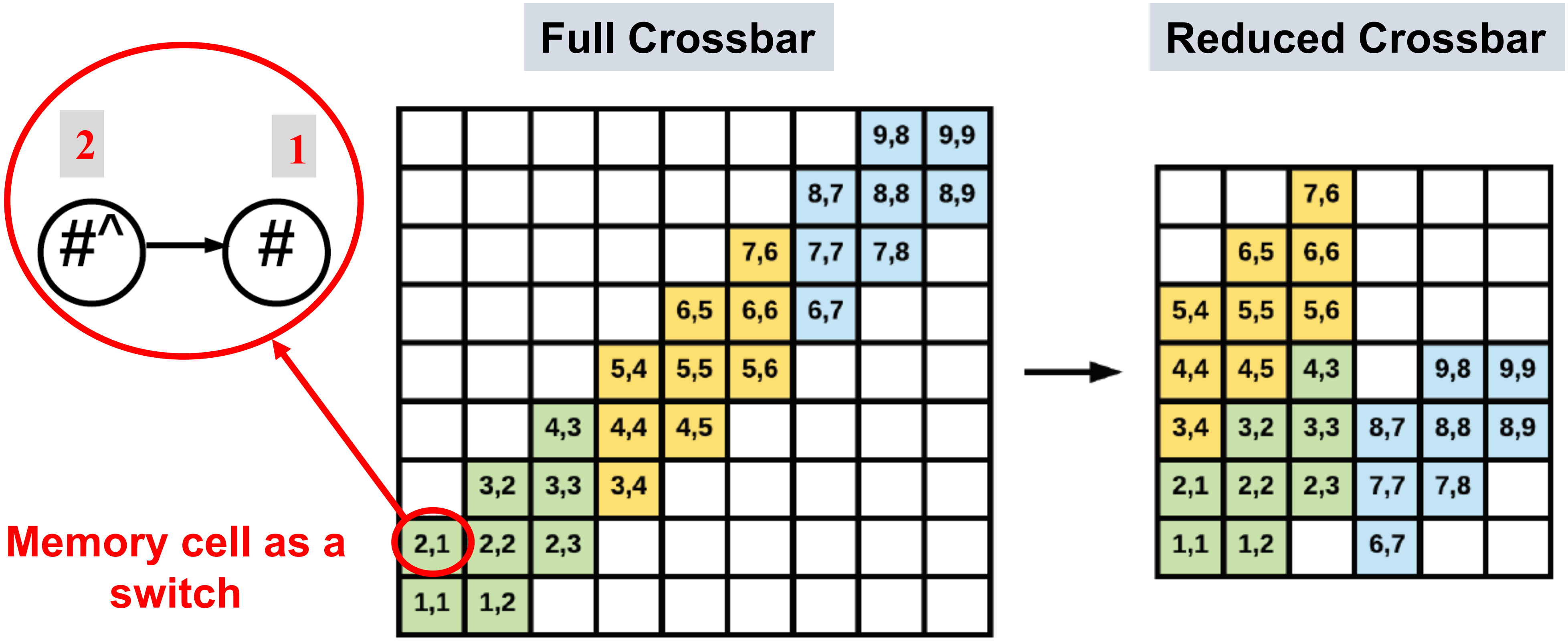
Fermi



Snort



Solution: Reduced Crossbar Interconnect



Memory cell as a switch

Full Crossbar

							9,8	9,9
						8,7	8,8	8,9
					7,6	7,7	7,8	
				6,5	6,6	6,7		
			5,4	5,5	5,6			
		4,3	4,4	4,5				
	3,2	3,3	3,4					
2,1	2,2	2,3						
1,1	1,2							

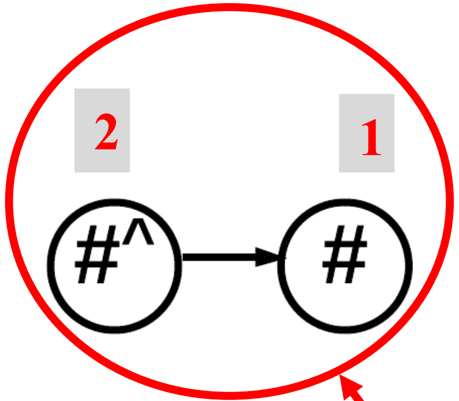
9×9

Reduced Crossbar

		7,6			
	6,5	6,6			
5,4	5,5	5,6			
4,4	4,5	4,3		9,8	9,9
3,4	3,2	3,3	8,7	8,8	8,9
2,1	2,2	2,3	7,7	7,8	
1,1	1,2		6,7		

6×6

Solution: Reduced Crossbar Interconnect



Memory cell as a switch

Full Crossbar

							9,8	9,9
						8,7	8,8	8,9
					7,6	7,7	7,8	
				6,5	6,6	6,7		
			5,4	5,5	5,6			
		4,3	4,4	4,5				
	3,2	3,3	3,4					
2,1	2,2	2,3						
1,1	1,2							

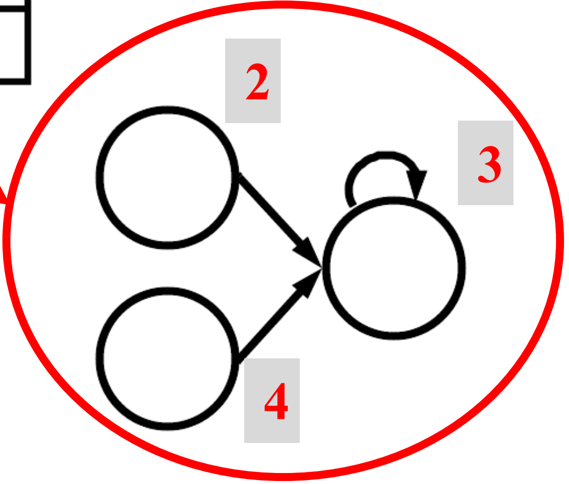
9 × 9

Reduced Crossbar

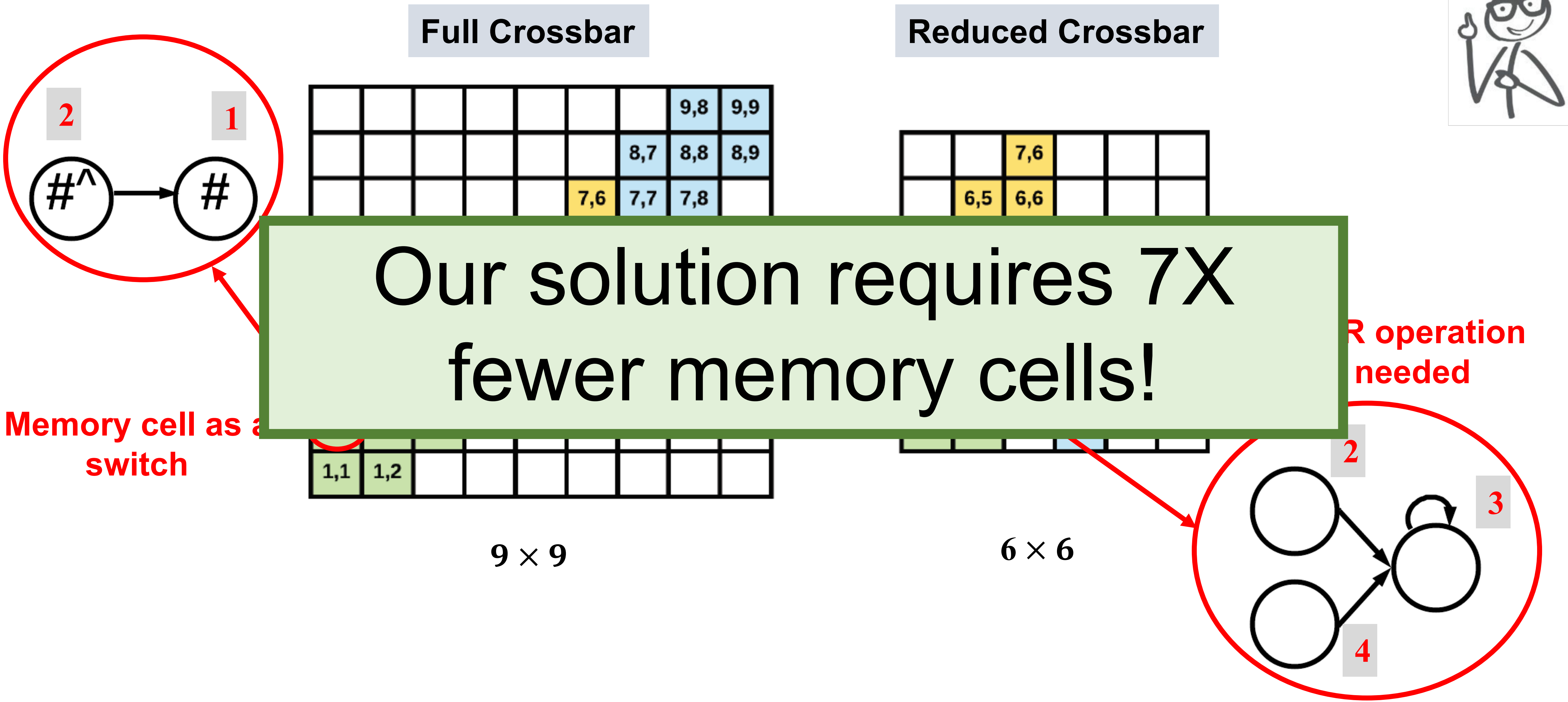
		7,6			
	6,5	6,6			
5,4	5,5	5,6			
4,4	4,5	4,3		9,8	9,9
3,4	3,2	3,3	8,7	8,8	8,9
2,1	2,2	2,3	7,7	7,8	
1,1	1,2		6,7		

6 × 6

An OR operation is needed



Solution: Reduced Crossbar Interconnect



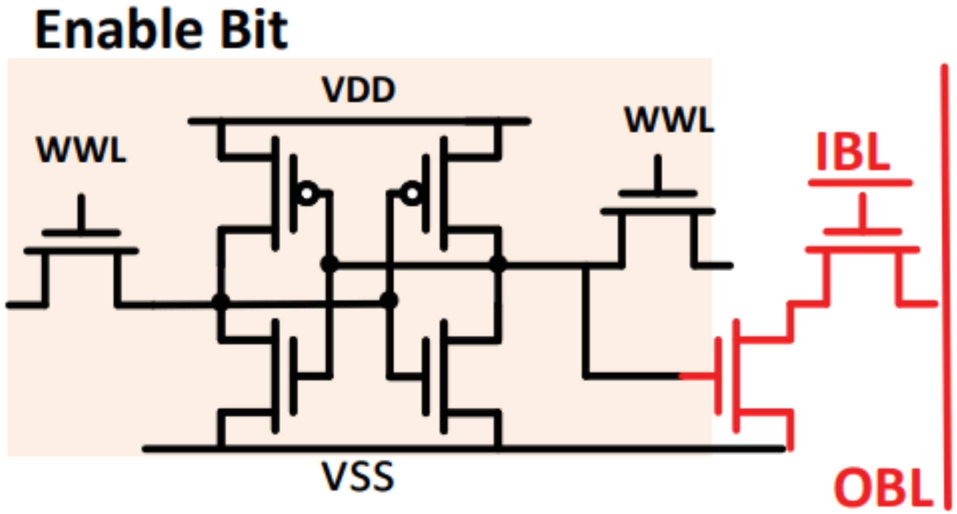
Our solution requires 7X fewer memory cells!

Memory cell as a switch

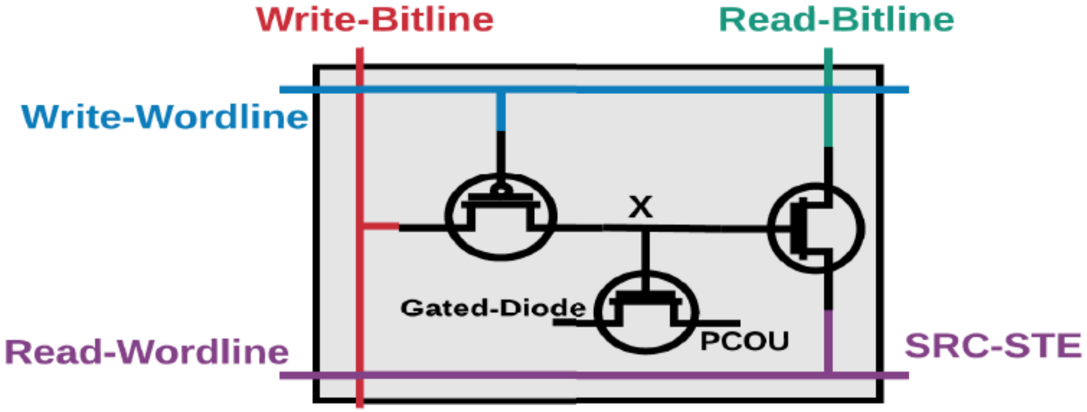
R operation needed

Mapping to Memory Technology

- Non-destructive read is necessary to implement OR functionality
- 2T1D cell has lower area overhead than 8T cell



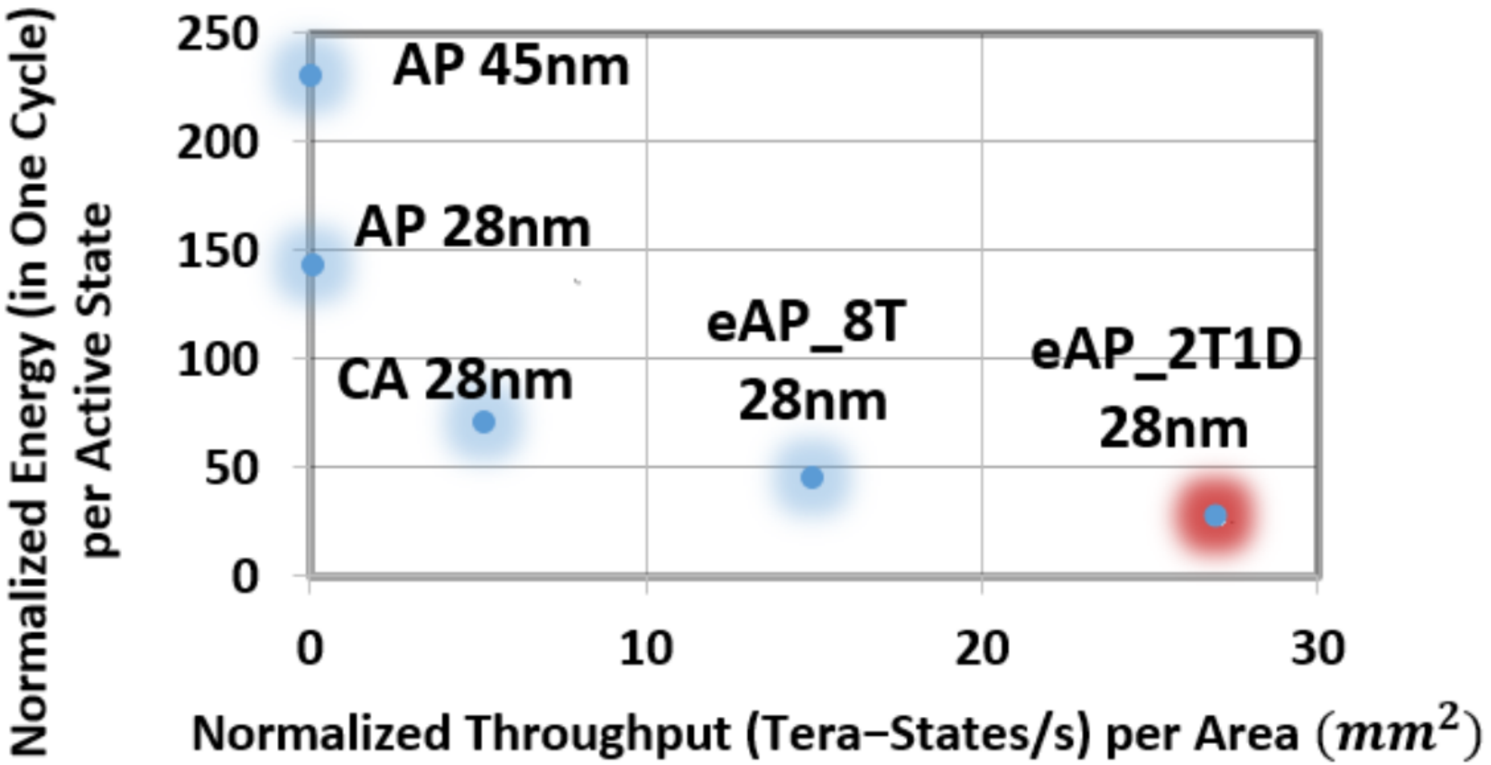
Cache Automaton use
8T SRAM cell



We propose to use 2T1D cell
(a type of gain cell)

Summary of Performance Evaluation

- Incorporate both architectural contribution and technology contribution
- eAP_2T1D has **1.7X**, **3.3X** and **210X** better throughput per unit area than **eAP_8T**, **CA,** and the **AP**



Thanks for Listening!

Questions?
Please stop by my poster

This work was supported in part by Semiconductor Research Corporation (SRC).