AN OVERVIEW OF MICRON'S **AUTOMATA PROCESSOR**

[1]**Ke Wang**, [1]*Kevin Angstadt*, [1]*Chunkun Bo*, [1]*Nathan Brunelle*,[1]*Elaheh Sadredini*,[2]*Tommy Tracy II*,[1]*Jack Wadden*, [2]*Mircea Stan*, [1]*Kevin Skadron*

Center for Automata Computing

[1]*Department of Computer Science*

[2]*Department of Electrical and Computer Engineering*

University of Virginia

Outline

- Motivation
- Architecture of the Automata Processor (AP)
  - Overview
  - AP board
  - Programming
- Applications on the AP
  - Bioinformatics
  - Data mining
  - Machine learning
  - Other applications
- Summary

Motivation

- 'Big data' age
- The end of Moore's law
- Limits of von Neumann architectures to support high degrees of parallelism

- Specialized hardware? – Accelerators

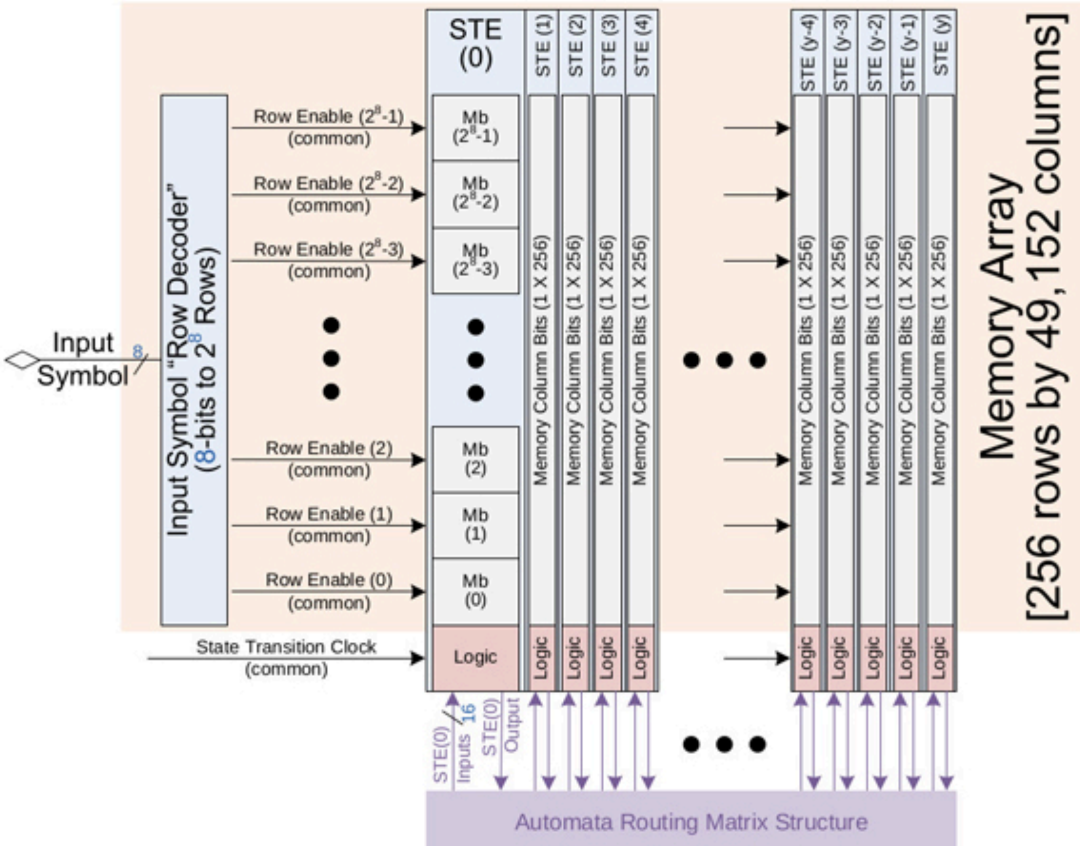
  Boost of performance and energy efficiency
- Pattern-based algorithm

  One of the most important building block in many data mining, machine learning, cybersecurity and bioinformatics

  Performance bottleneck in many scenarios

Architecture of the Automata Processor

❑ Overview



State transition element (STE)

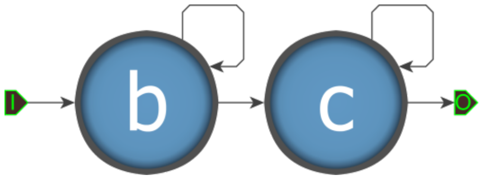
- AP is a hardware implementation of non-deterministic finite automata
- "MISD" with massive parallelism
- STE is based on DRAM column: 256 bits (per column) represent any set of 8-bit symbols
- Rich and reconfigurable on-chip routing resources
- All chips on a board work in parallel
- Operates on 133 MHz

UNIVERSITY of VIRGINIA

❑ Function Elements and Capacity

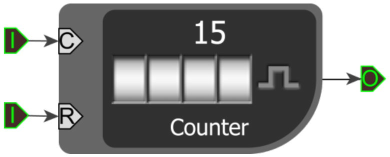
per chip

State Transition Element (STE)



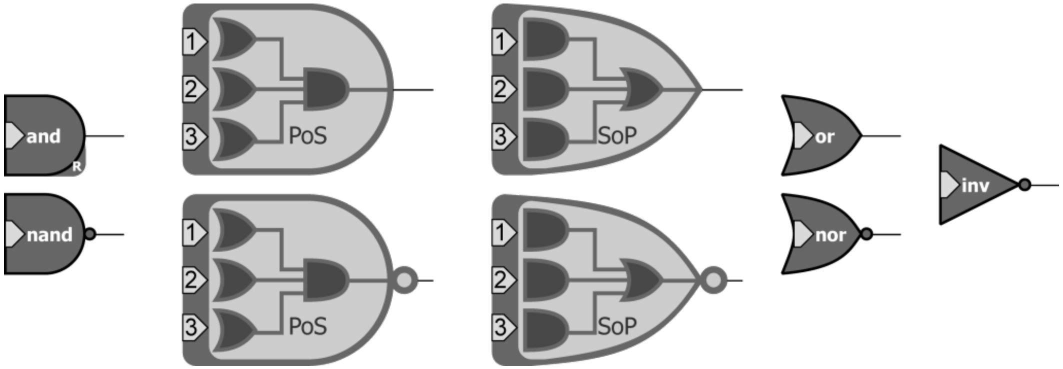
49,152

Counter Element



768

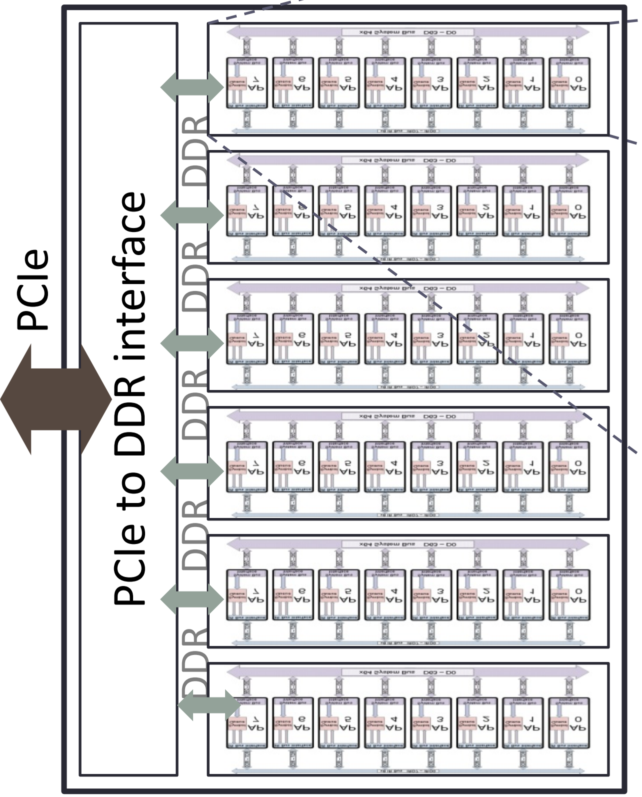
Boolean Logic Element

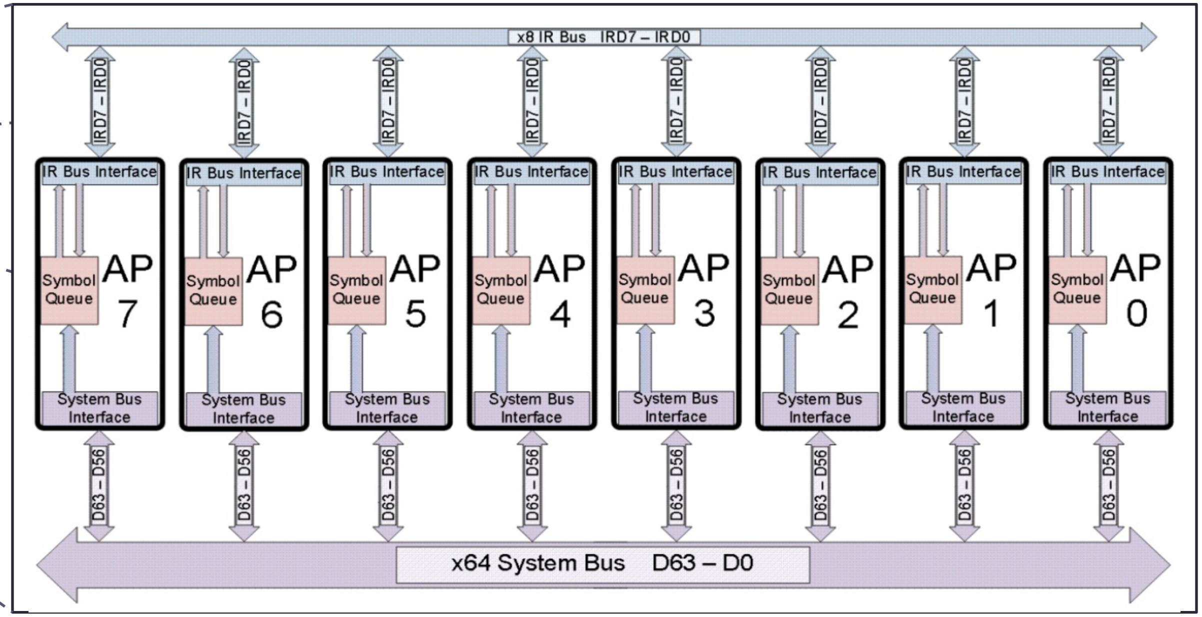


2,304

32 chips/board -> 1.5 million concurrent operations

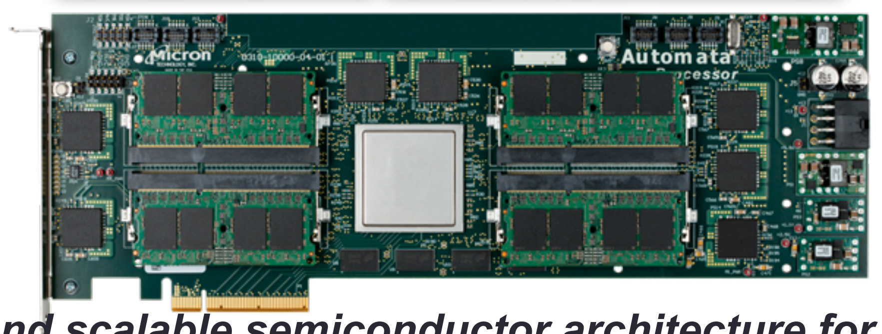
❑ AP board





Rank of Automata Processor chips

An AP rank (8 chips)



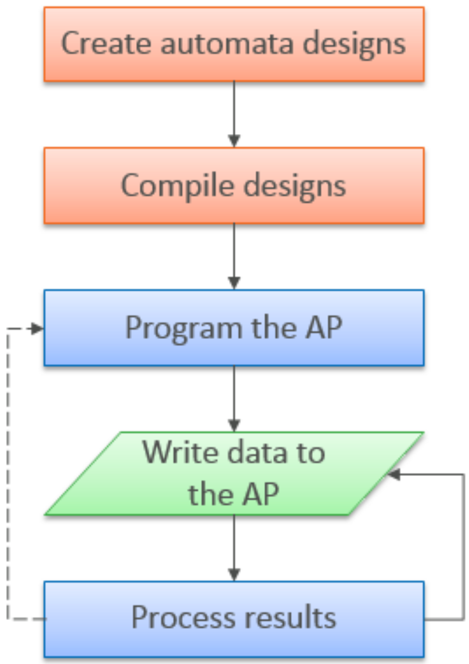
An AP board with 32 chips

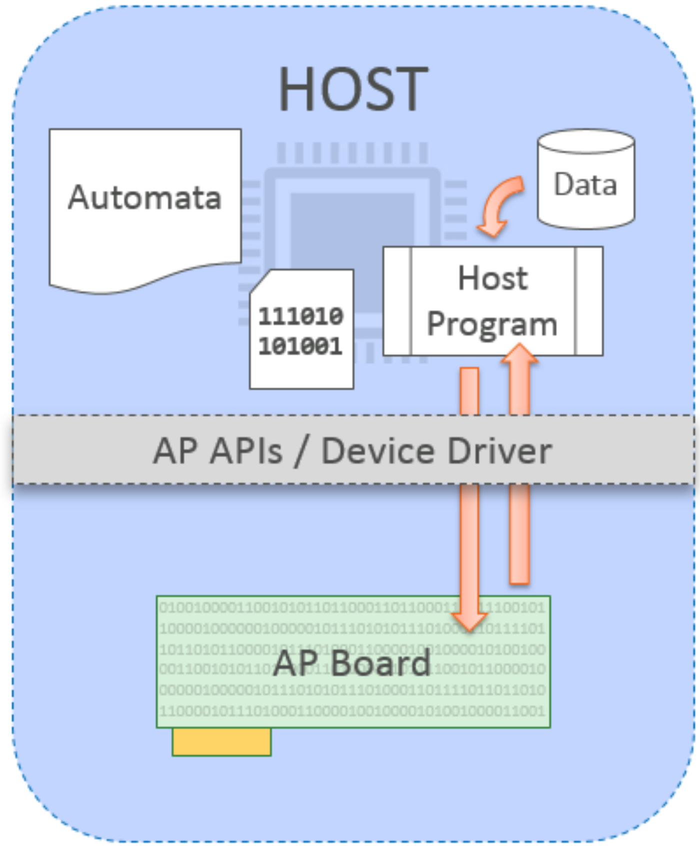
Automata Processor board

PCI-E
4GB on-board memory
On-board FPGA

P. Dlugosch et al. , An efficient and scalable semiconductor architecture for parallel automata processing, IEEE TPDS , vol. 25, no. 12, 2014.

❑ Programming



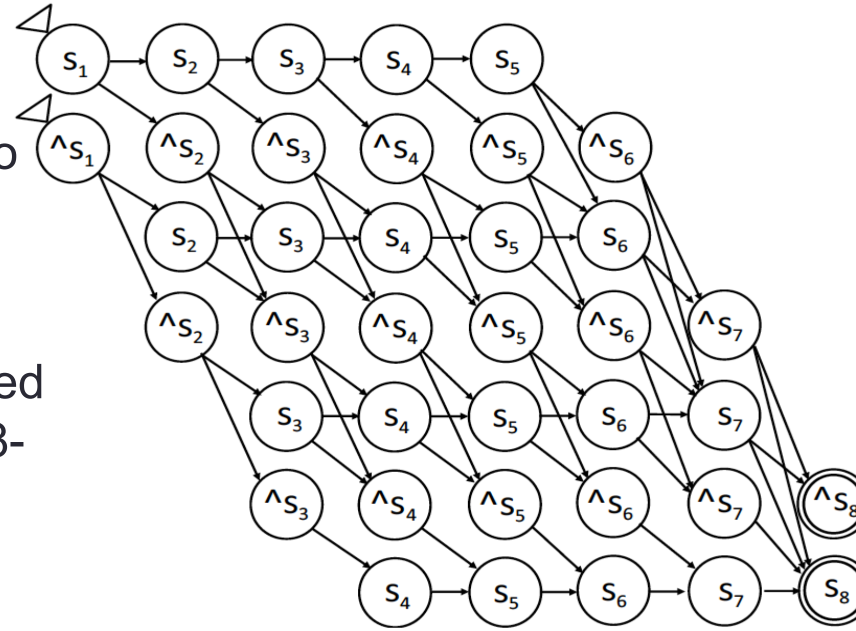


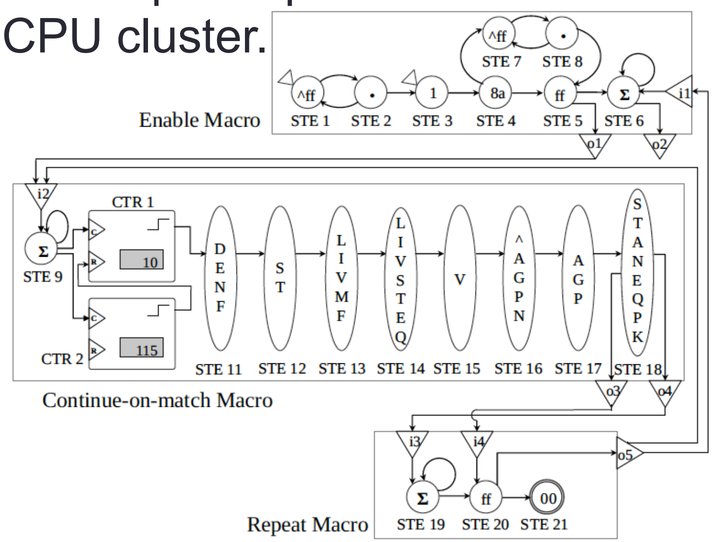
- Automata Network Markup Language (ANML) is an XML language for describing the composition of automata networks
- AP SDK: automata simulator, hardware emulator and API bundling with other languages: C, Python and Java
- Fast reconfiguration: 50ms for whole board, 45ms for symbol replacement
- Angstadt et al. presented RAPID (ASPLOS 2016) , a high-level programming language to improve programming productivity

Applications

❑ Bioinformatics

The DNA (l,d) motif search problem is known to be NP-hard, and the largest solved instance reported to date is (26,11). Roy and Aluru [1] proposed a novel algorithm using streaming execution over a large set of NFAs and achieved over 200X speedups on an AP board over a 48-core CPU cluster.



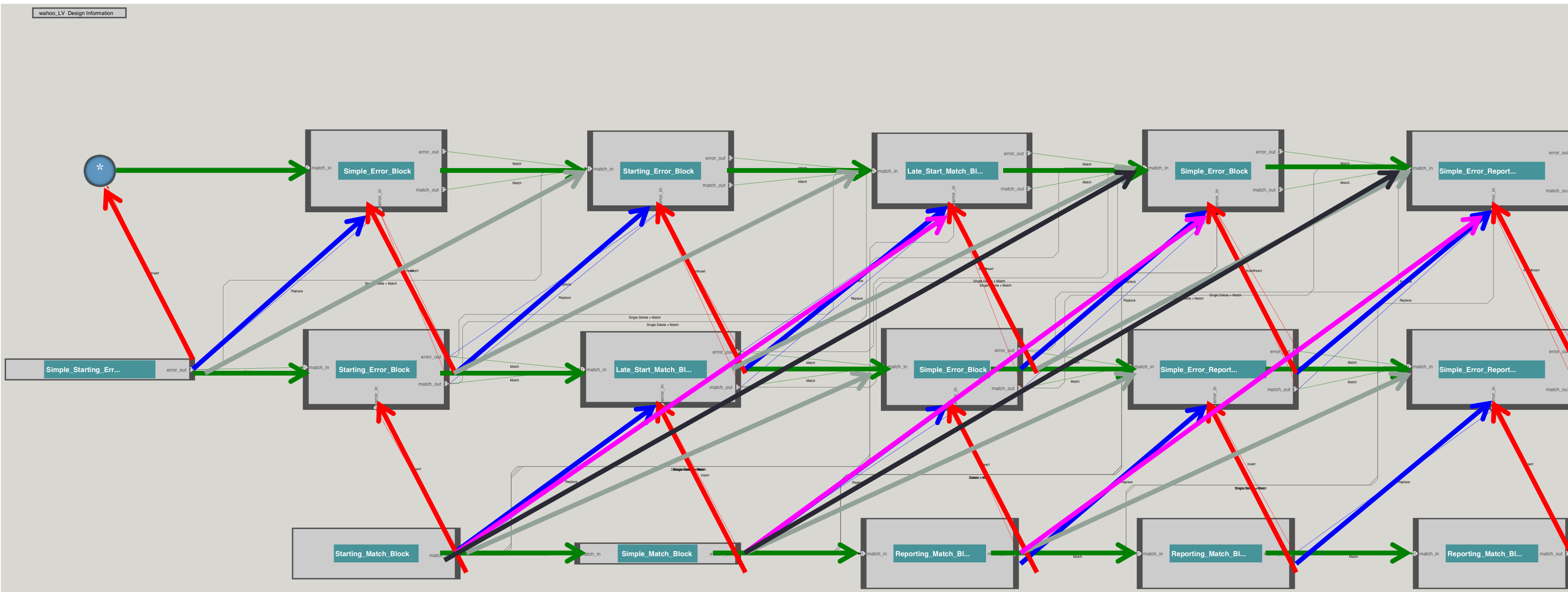


They also demonstrated PROTOMATA [2], an AP-accelerated protein motif algorithm, and achieved up to a half million times speed-up over single-threaded CPU with one AP board.

[1] I. Roy and S. Aluru, Finding motifs in biological sequences using the micron automata processor. In Proc. of IPDPS'14
[2] I. Roy et al., High Performance Pattern Matching using the Automata Processor. In Proc. of IPDPS'16, 2016.

UNIVERSITY of VIRGINIA

AP Levenshtein Automaton Network



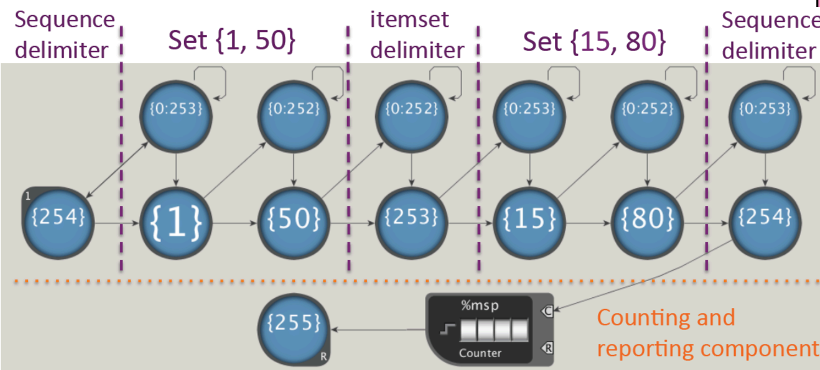
Delete + Delete + Match

The experiments show that run time remains linear with the input while the space requirement of the automaton becomes linear in the product of the configured pattern length and edit distance.

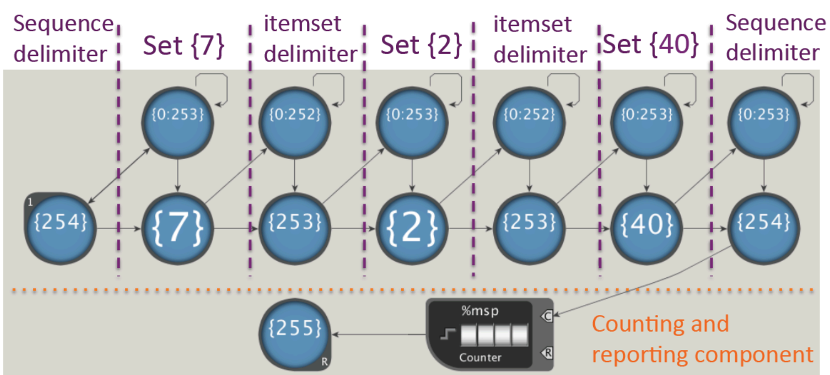
T. Tracy, M. Stan, N. Brunelle, J. Wadden, K. Wang, K. Skadron, G. Robins. In Proc. of ASBD, 2015

❑ Data mining

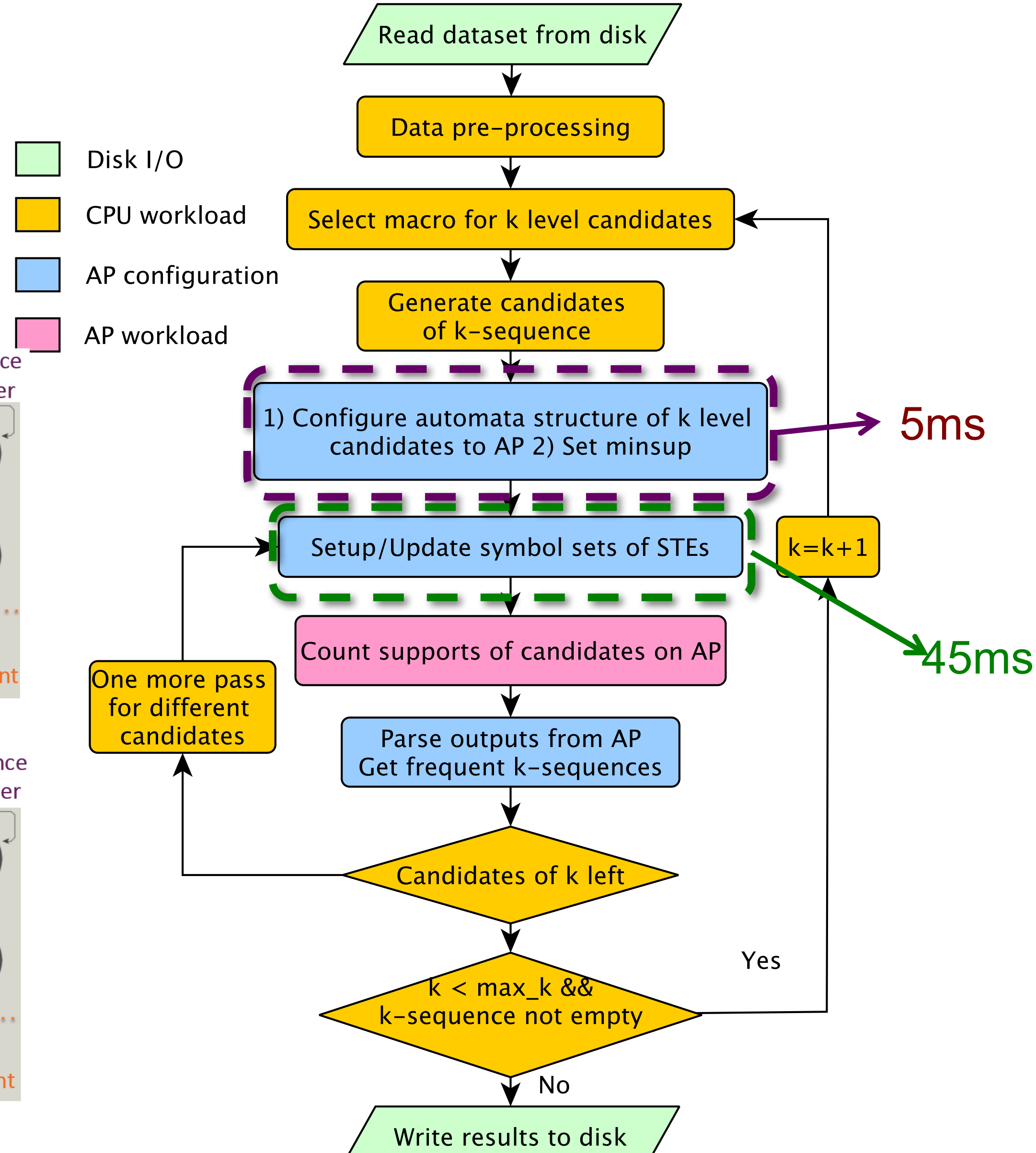
Frequent itemset mining and sequential pattern mining



(a) Automaton for matching sequence $< \{1, 50\}, \{15, 80\} >$

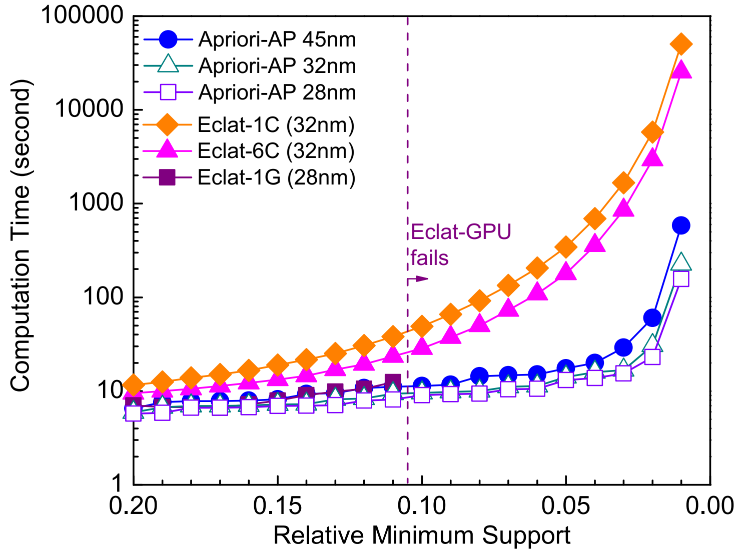


(b) Automaton for matching sequence $< \{7\}, \{2\}, \{40\} >$



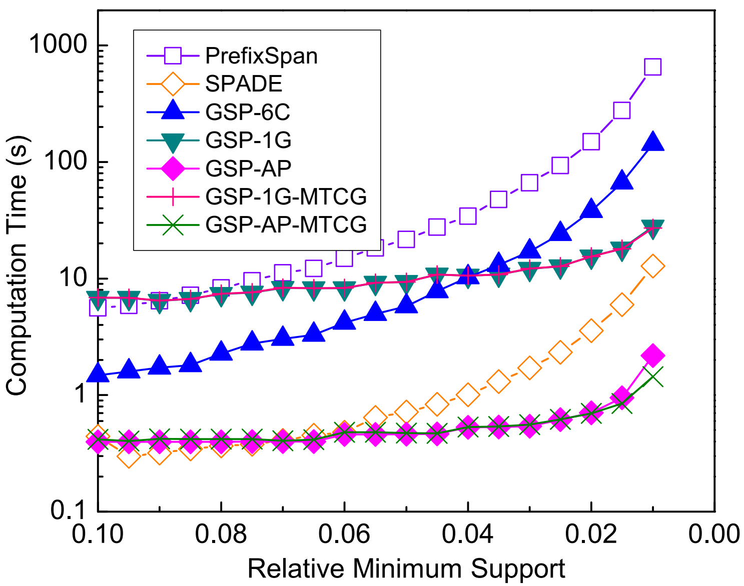
[1] K. Wang et al. Association rule mining with the micron automata processor. IPDPS '15

[2] K. Wang, Elaheh Sadredini and Kevin Skadron. Sequential Pattern Mining with the Micron Automata Processor. Computing Frontiers 2016

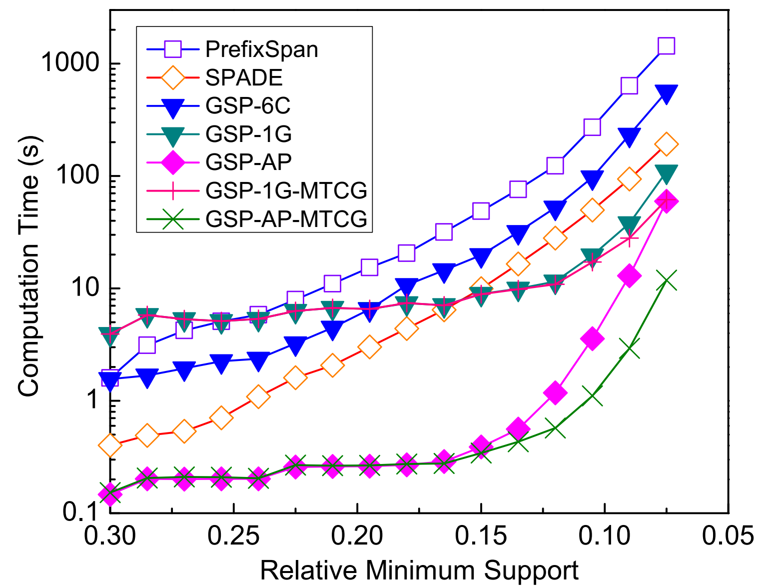


ENWiki

Up to 129X and 49X speedups are achieved by the AP-accelerated FSM on seven synthetic and real-world datasets, when compared with the Apriori single core CPU implementation and Eclat, a more efficient FSM algorithm, compared to a 6-core multicore CPU



Bible

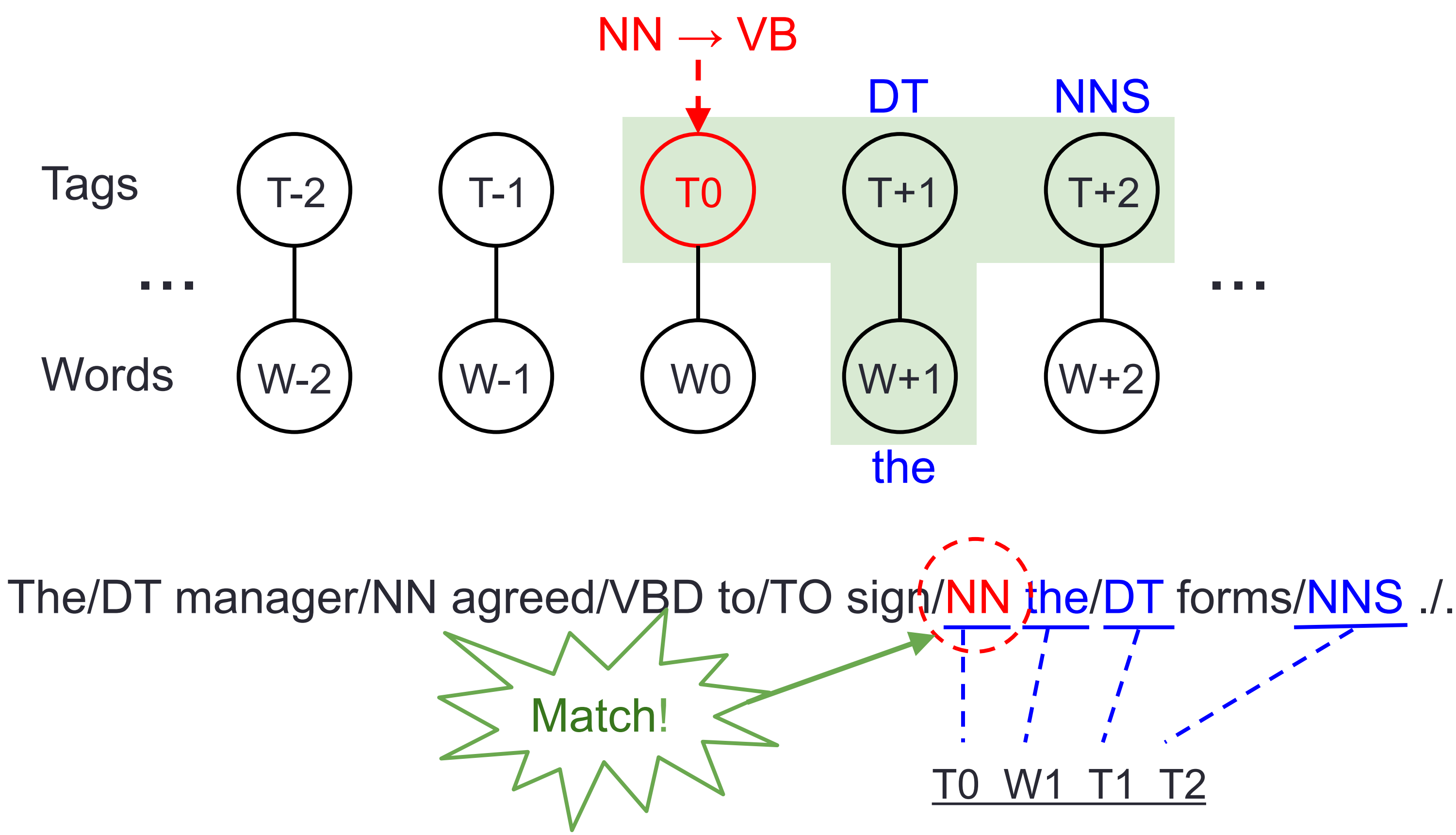


FIFA

Sequential pattern mining: the GSP-AP-MTCG get 452X speedup over PrefixSpan (in Bible) and up to 49X speedup over SPADE (in FIFA).

Machine learning

Brill Tagging – a rule based part-of-speech tagging



[1] K. Zhou et al. Regular expression acceleration on the micronautomata processor: Brill tagging as a case study. Big Data 2015

Converting transformation rules to regex

NN->VB if Pos:DT@[1] & Pos:NNS@[2] & Word:the@[1]

\s+[^\s+]\/NN\s+the\/DT\s+[^\s+]\/NNS\s+[^\s+]\s

Word/Tag Position: 0 1 2 3 (padding)

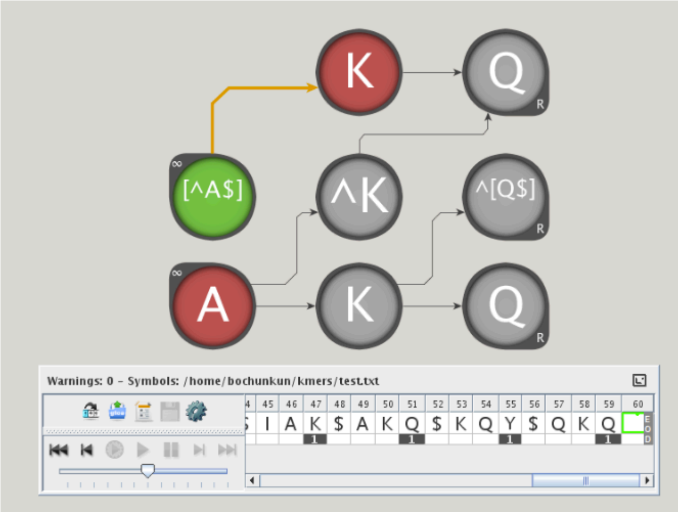
2.2 MB file size (2,198,493 characters); Time in microseconds (µs)			
	200 Simple (2737 STEs)	**200 Original** (2934 STEs)	**200 Complex** (5843 STEs)
Xeon Phi	8,159,165µs **(20 Threads)**	8,367,107µs **(100 Threads)**	8,882,996µs **(200 Threads)**
Intel i7 (12 Threads)	513,631µs	563,720µs	1,053,676µs
AP	16,489 5µs		
AP Speed-up over Intel i7	31.15X	34.19X	63.90X

K. Zhou et al. Regular expression acceleration on the micronautomata processor: Brill tagging as a case study. Big Data 2015

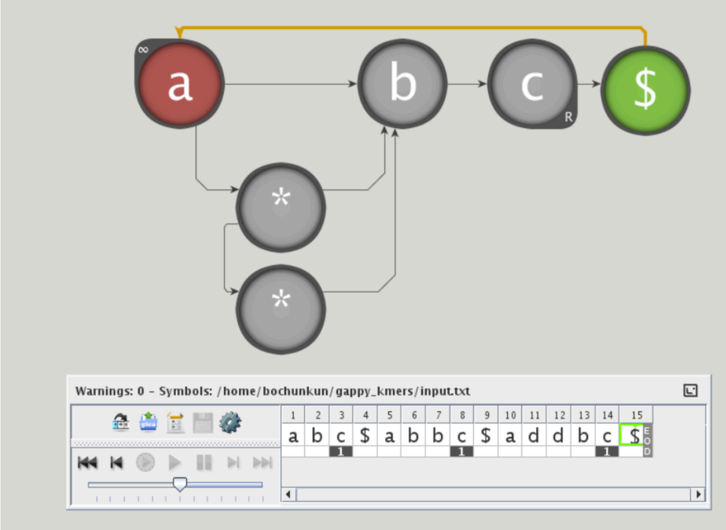
String Kernel

- String Kernel (SK) is a widely used kernel in machine learning and text mining
- Fast processing is required, especially for the testing phase
- Feature vector mapping is the current performance bottleneck, which involves a lot of pattern matching
- Exact-match kernel, mismatch kernel, gappy kernel, spatial kernel

- Mismatch kernel
  - K = 3
  - m = 0, 1



- Gappy kernel
  - K = 3
  - g <= 2



- Spatial Kernel
  - t = 2, k = 1, d < 5

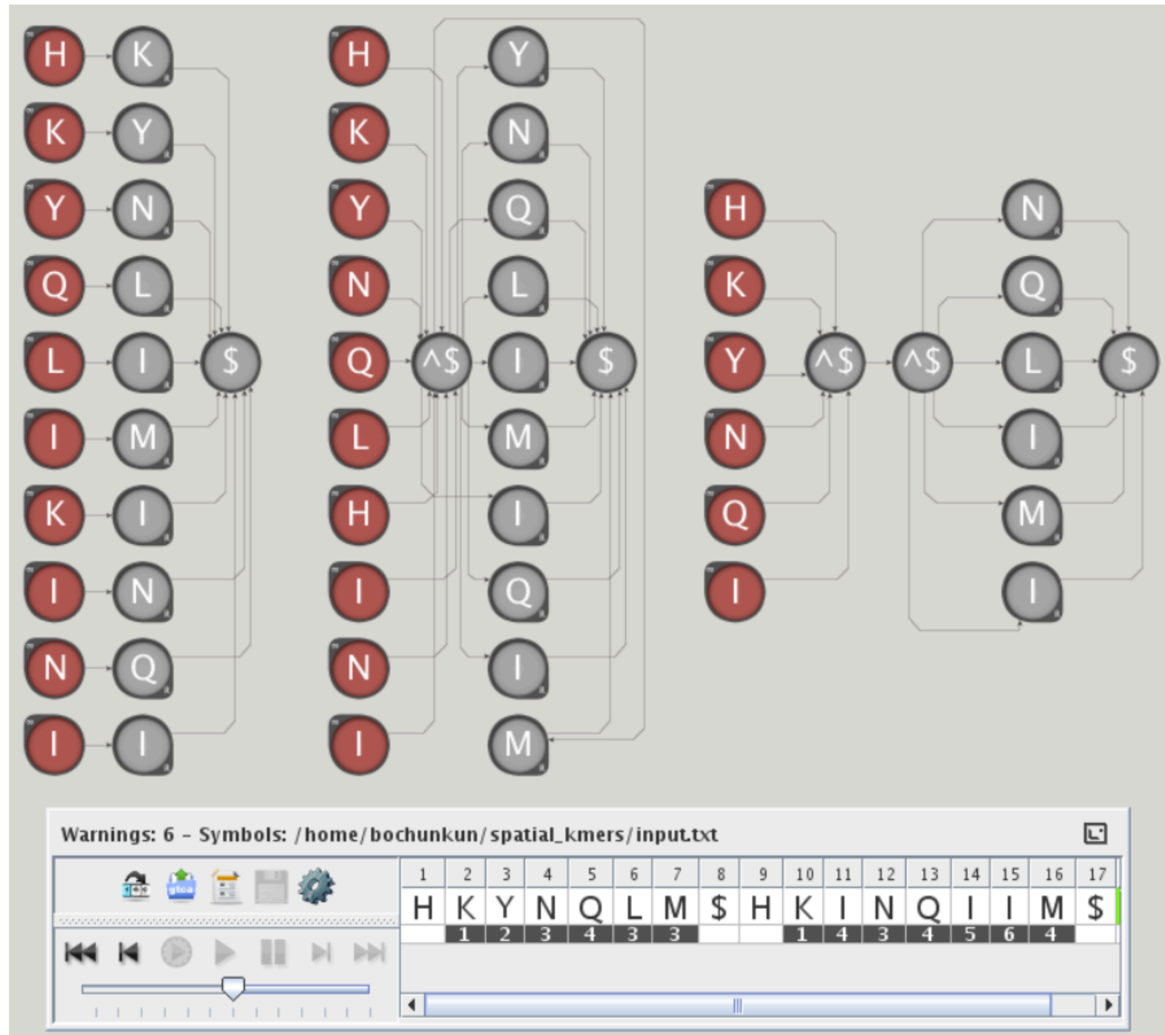
    Input1 = HKYNQLM

    Input2 = HKINQIIM

HK	H_Y	H__N	H___Q	H____L
KY	K_N	K__Q	K___L	K____I
YN	Y_Q	Y__L	Y___I	Y____M
NQ	N_L	N__I	N___M	
QL	Q_I	Q__M		
LI	L_M			
IM				

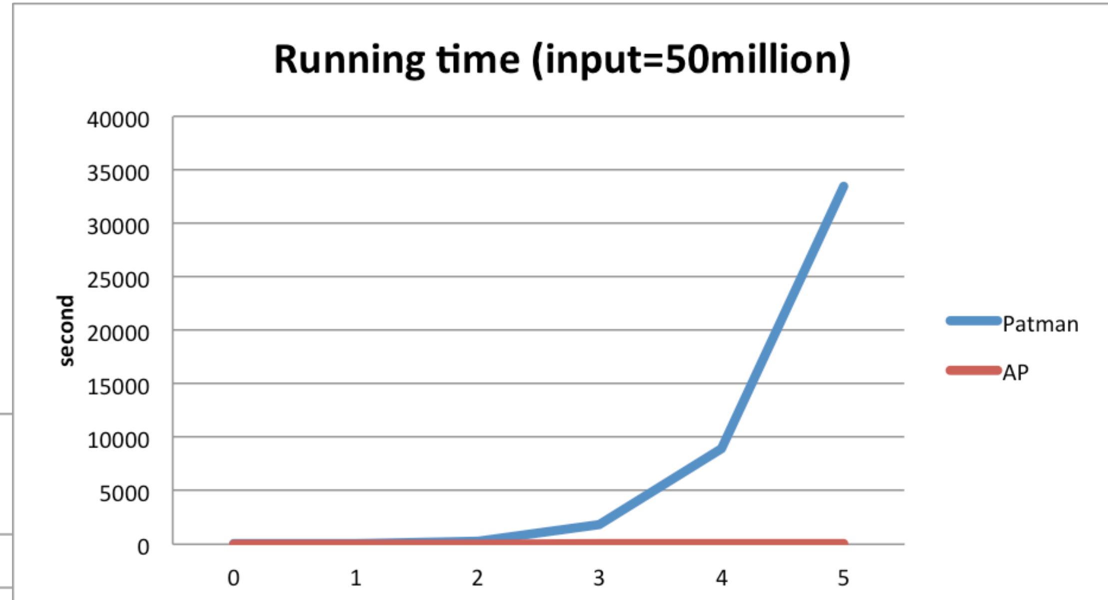
HK	H_I	H__N	H___Q	H____I
KI	K_N	K__Q	K___I	K____I
IN	I_Q	I__I	I___I	I____M
NQ	N_I	N__I	N___M	
QI	Q_I	Q__M		
II	I_M			
IM				

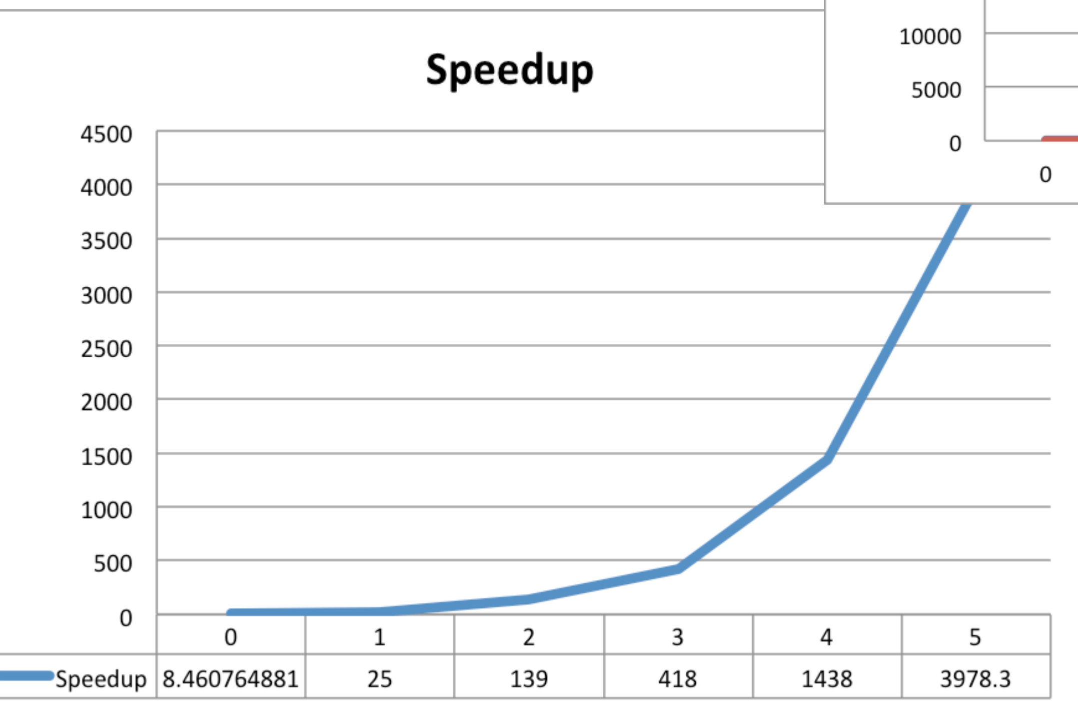
d = 0 d = 1 d = 2



String Kernel - performance

- PatMaN increases exponentially
- AP increases linearly



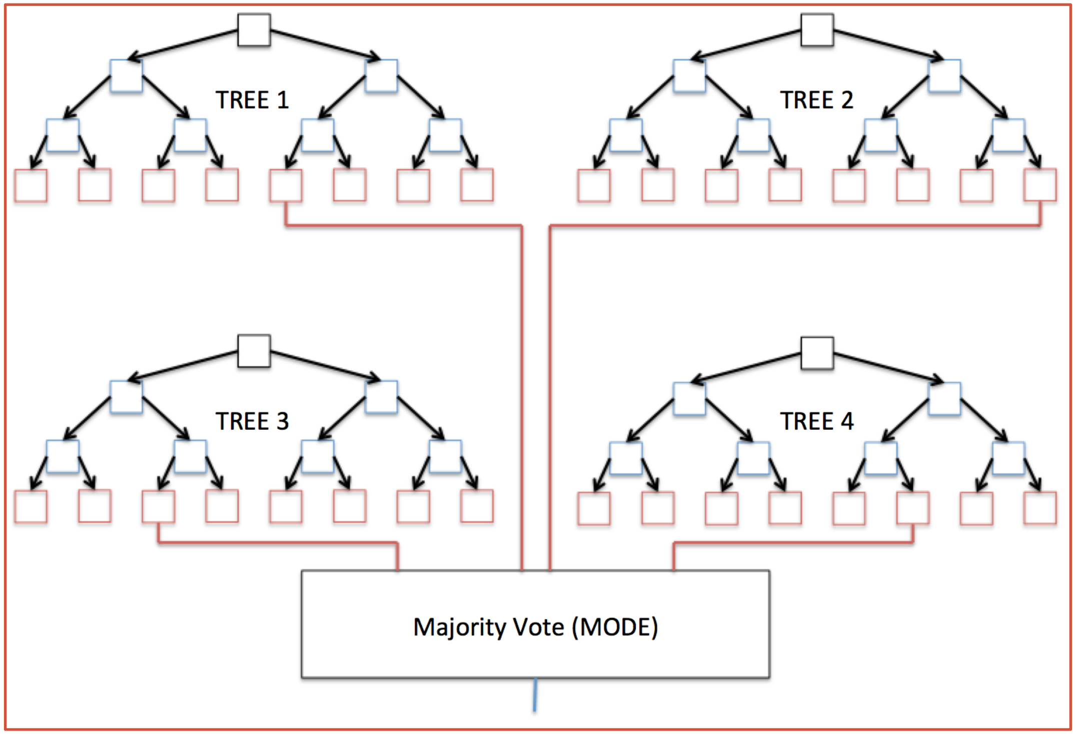


- Speedups increases exponentially
  8.5x ~ 3980x

C. Bo, K. Wang, Y. Qi, and K. Skadron. String kernel testing acceleration using the Micron Automata Processor.Workshop on Computer Architecture for Machine Learning, 2015

Random forest

- A general-purpose, supervised machine learning algorithm.
- Composed of an ensemble of decision trees.
  - The **majority vote** of the constituent weak classifiers serves as the resulting classification.
- Used in many big-data applications including:
  - Image processing
  - Natural Language Processing
  - Bioinformatics



Our approach:

- Restructure each Decision Tree into chains
  - Each chain represents a path through each tree in the forest.
  - Do this for ALL trees in the forest.
- Use the AP to match all chains in parallel

Random Forest – performance results

Table 2: Key data points of MNIST Results

Trees	Leaves	Accuracy	AP Throughput (k Pred/Sec)	CPU Throughput (k Pred/Sec)	AP Speed Up
5	50	82.2%	13200	337	39
10	50	86.1%	5980	242	25
20	50	87.8%	4170	150	28
40	50	88.7%	3350	86.5	39
80	50	89.2%	2940	46.4	63
160	50	89.6%	1350	25.0	54
10	500	93.3%	2480	205	12
20	500	94.3%	1160	125	9
40	750	95.2%	420	68.0	6
80	1250	96.0%	111	34.3	3
20	4000	96.1%	129	98.9	1.3
40	4750	96.7%	55.0	51.5	1.1
80	5000	96.9%	25.0	26.6	0.9
160	5000	97.1%	12.2	13.5	0.9

Table 1: Key data points of Twitter Results

Trees	Leaves	Accuracy	AP Throughput (k Pred/Sec)	CPU Throughput (k Pred/Sec)	AP Speed Up
5	40	66.9%	14400	154	93
10	40	67.5%	8130	129	63
20	40	67.7%	5360	93.4	57
40	40	68.0%	3750	58.5	64
5	600	70.4%	2010	118	17
10	600	71.4%	1530	86.4	18
20	700	71.7%	385	51.5	7
40	700	71.9%	194	32.4	6

- **MNIST**: The AP achieved a max 63x speedup over CPU
- **Twitter**: The AP achieved a max 93x speedup over CPU

T. Tracy II, Y. Fu, I. Roy, E. Jonas, P. Glendenning. Toward machine learning on the Automata Processor. ISC-HPC 2016

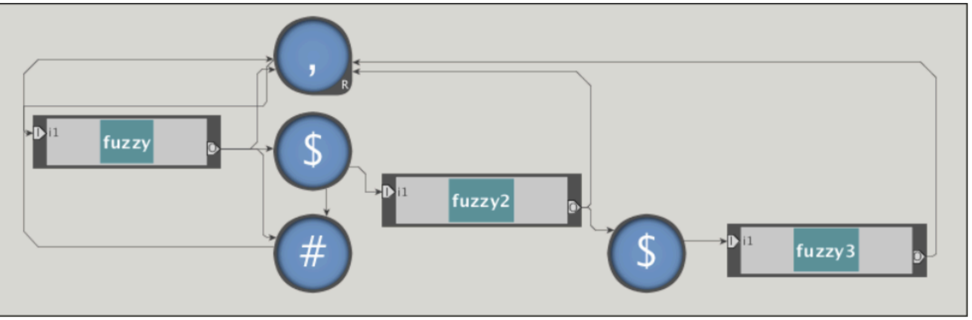
Other applications

Entity resolution

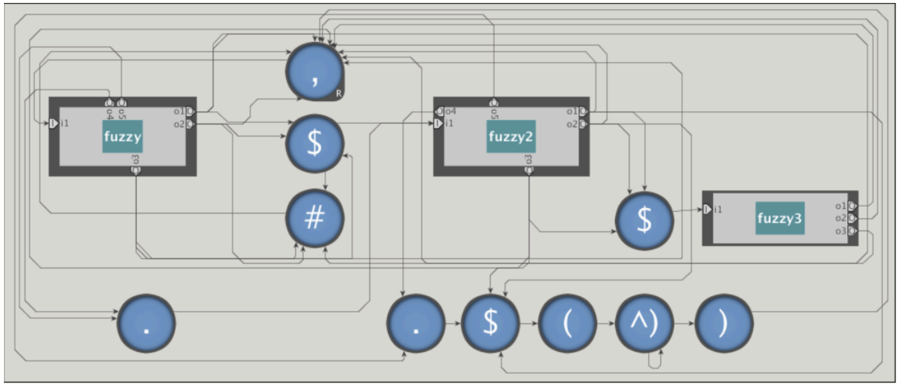
- ***Entity Resolution:*** process of determining whether two references to real-world objects are referring to the same object. It fixes the problems:
  - miss correct results
  - waste storage space
  - slow down query speed

Challenges
- time-consuming
- difficult to identify similar items
- lack of general methods

Utilize the massive parallelism and fuzzy matching ability of the AP to speed up ER

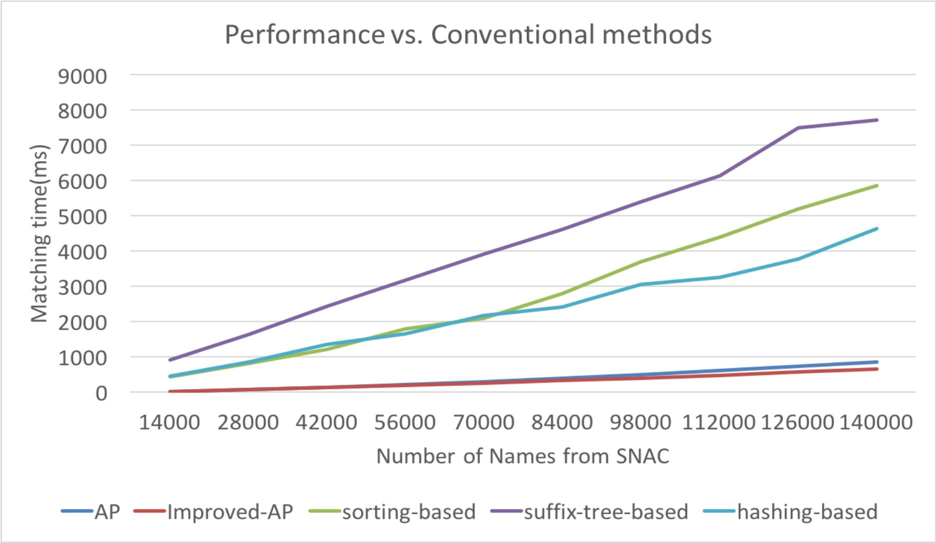


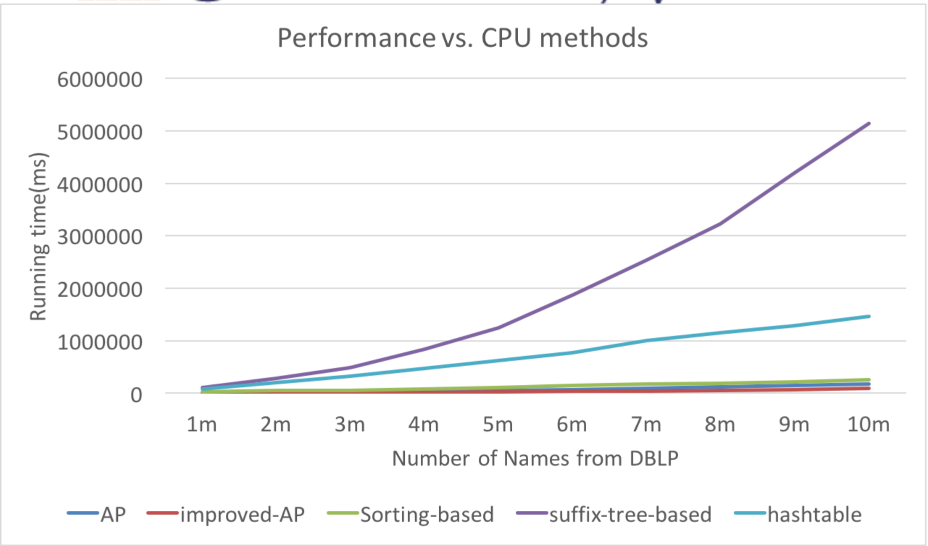
family name



first name

C. Bo, K. Wang, J. Fox, and K. Skadron. Entity Resolution Acceleration using Automata Processor. In Proc. ASBD,2015





Up to 17X speedup

Method	Comp Rate	Correct Pairs #	Percent age	GMD
Lucene	65.3%	262	80.6%	54
Sorting	71.4%	233	71.7%	63
Hashing	73.2%	213	65.6%	72
Suffix-tree	73.2%	213	65.6%	72
AP	57.2%	292	89.8%	31
Manual	47.4%	325	100%	0

Accuracy for SNAC

Method	Correct Pairs #	Perce ntage	GMD
Sorting	502	74.4%	183
Hashing	484	71.7%	212
Suffix-tree	484	71.7%	212
AP	615	91.4%	62
Manual	675	100%	0

Accuracy for DBLP

Summary

- The AP leverages bit-level parallelism to provide native support for efficient NFA execution, enabling dramatic speedups for a variety of algorithms
- We showed promising applications in the fields of bioinformatics, data mining and machine learning
- The University of Virginia established Center for Automata Processing (http://cap.virginia.edu) to build a vibrant ecosystem of researchers, developers, and adopters for the exciting new Automata Processor

Thank you!