A Dynamically Reconfigurable Automata Processor Overlay

Rasha Karakchi, Lothrop O. Richards, **Jason D. Bakos**

Department of Computer Science and Engineering

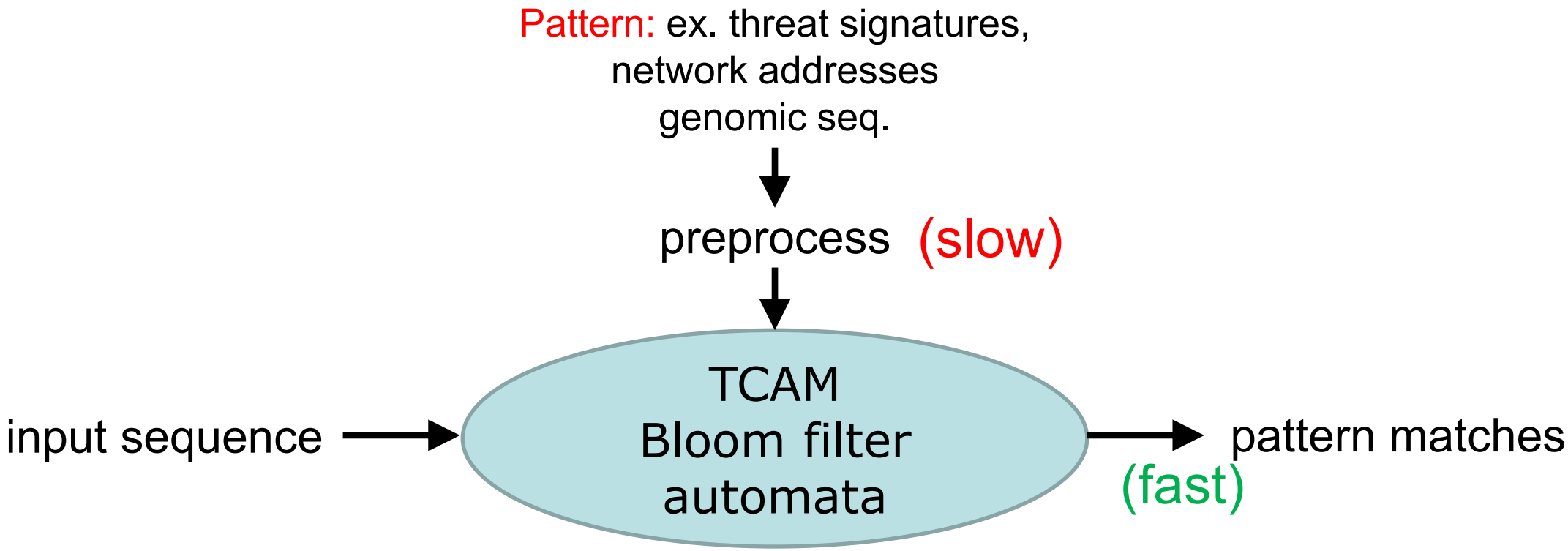
Heterogeneous and Reconfigurable Computing Group

UNIVERSITY OF SOUTH CAROLINA

This material is based upon work supported by the National Science Foundation under Grant No. 1421059.

Pattern Matching

- Pattern matching historically a good fit for FPGAs

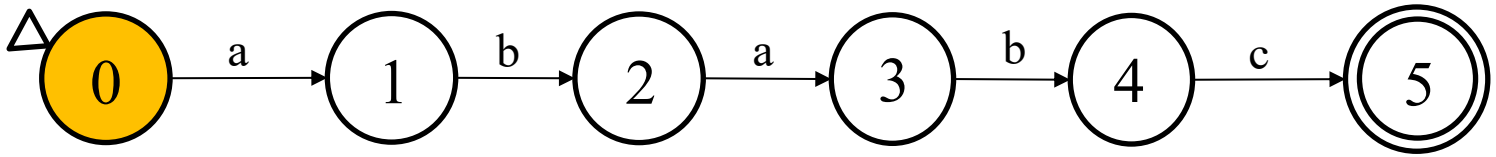


Nondeterministic Finite Automata (NFA)

- Directed graph
  - Vertices => states for partial or complete sequence matches
  - Edges => input value
- Multiple states may be active at one time
- Parallelism exploited from the parallel evaluation of all next-state tables

Automata Processing

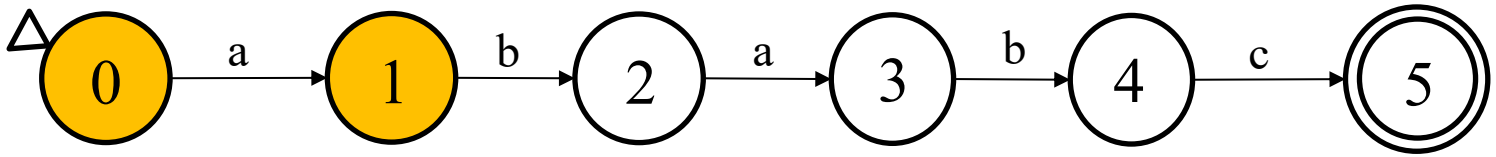
- Recognize pattern: “ababc”



Input	Active States
	0

Automata Processing

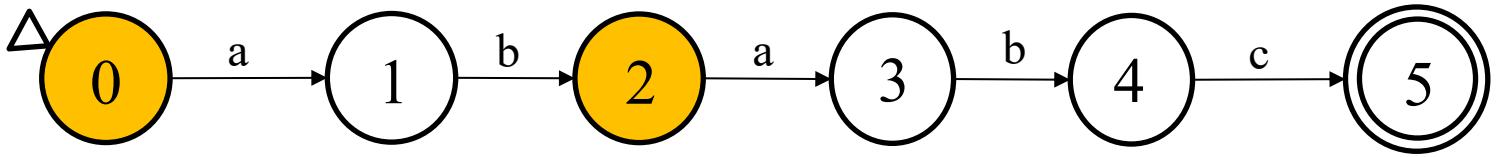
- Recognize pattern: “ababc”
- Input: “a”



Input	Active States
	0
a	0,1

Automata Processing

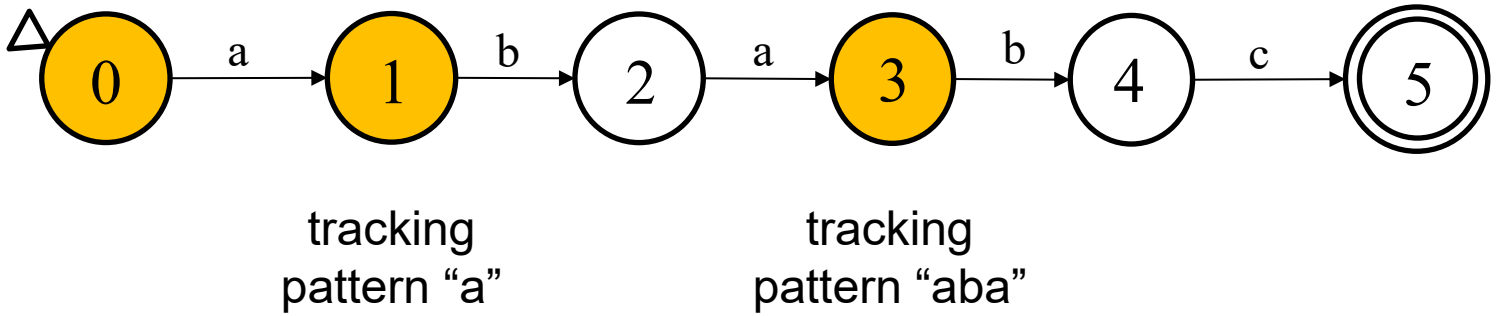
- Recognize pattern: “ababc”
- Input: “ab”



Input	Active States
	0
a	0,1
b	0,2

Automata Processing

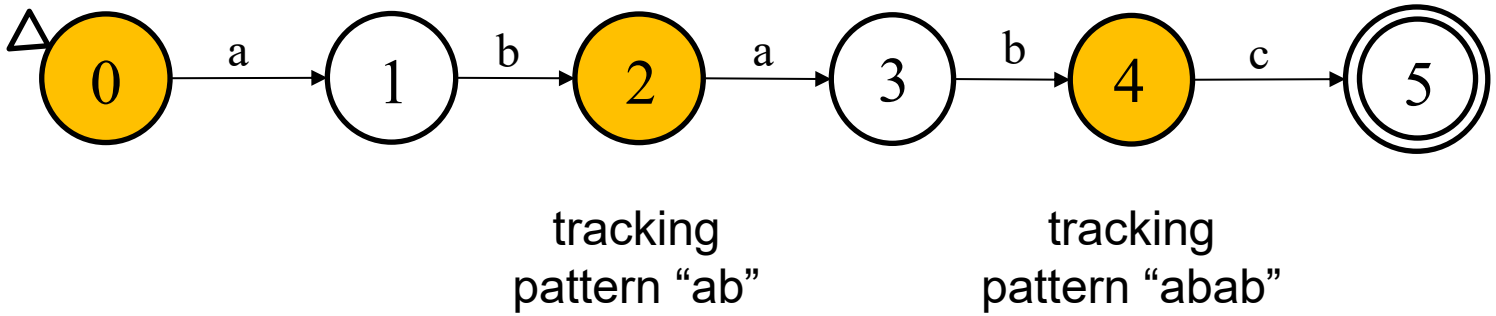
- Recognize pattern: “ababc”
- Input: “aba”



Input	Active States
	0
a	0,1
b	0,2
a	0,1,3

Automata Processing

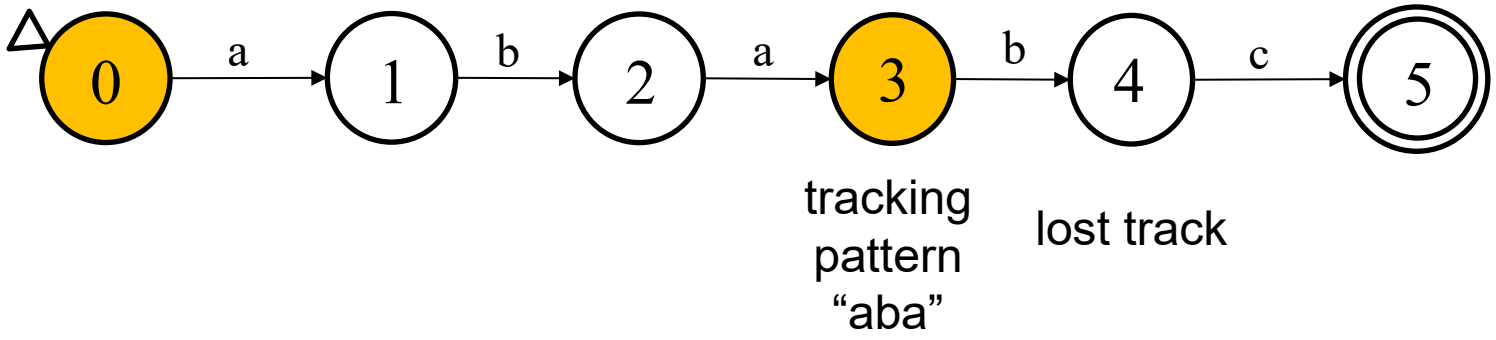
- Recognize pattern: “ababc”
- Input: “abab”



Input	Active States
	0
a	0,1
b	0,2
a	0,1,3
b	0,2,4

Automata Processing

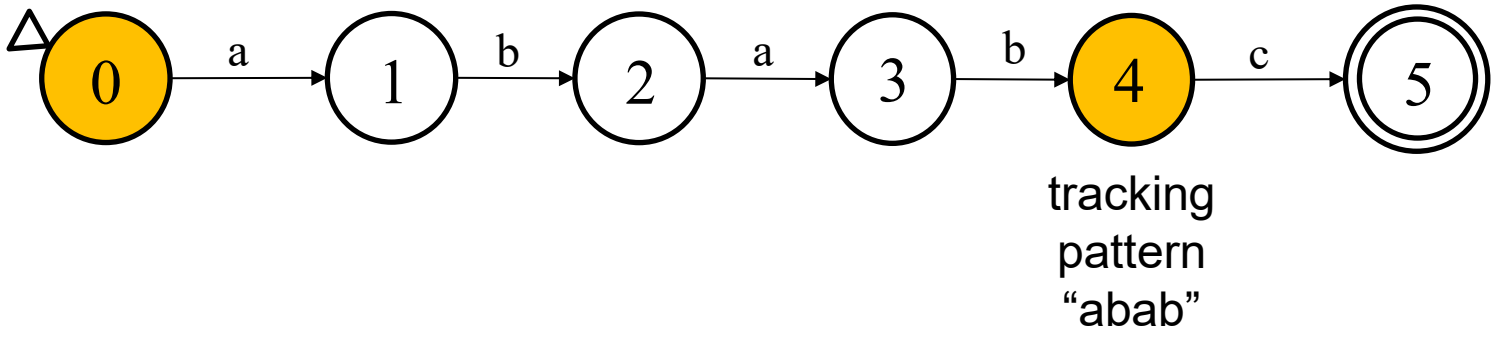
- Recognize pattern: “ababc”
- Input: “ababa”



Input	Active States
	0
a	0,1
b	0,2
a	0,1,3
b	0,2,4
a	0,3

Automata Processing

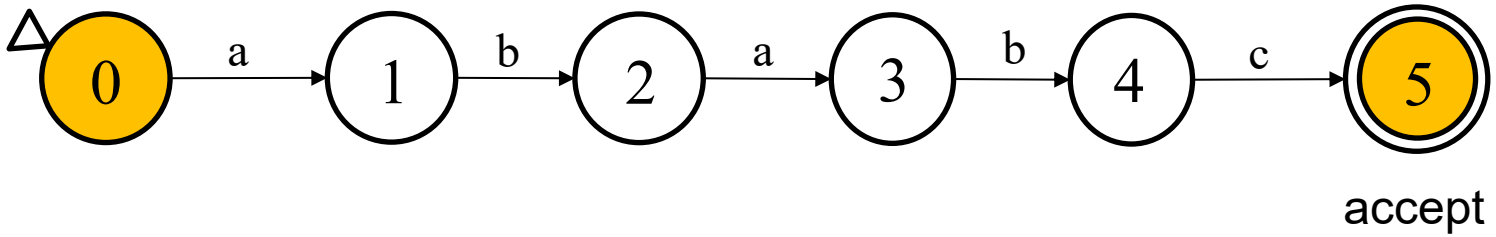
- Recognize pattern: “ababc”
- Input: “abababb”



Input	Active States
	0
a	0,1
b	0,2
a	0,1,3
b	0,2,4
a	0,3
b	0,4

Automata Processing

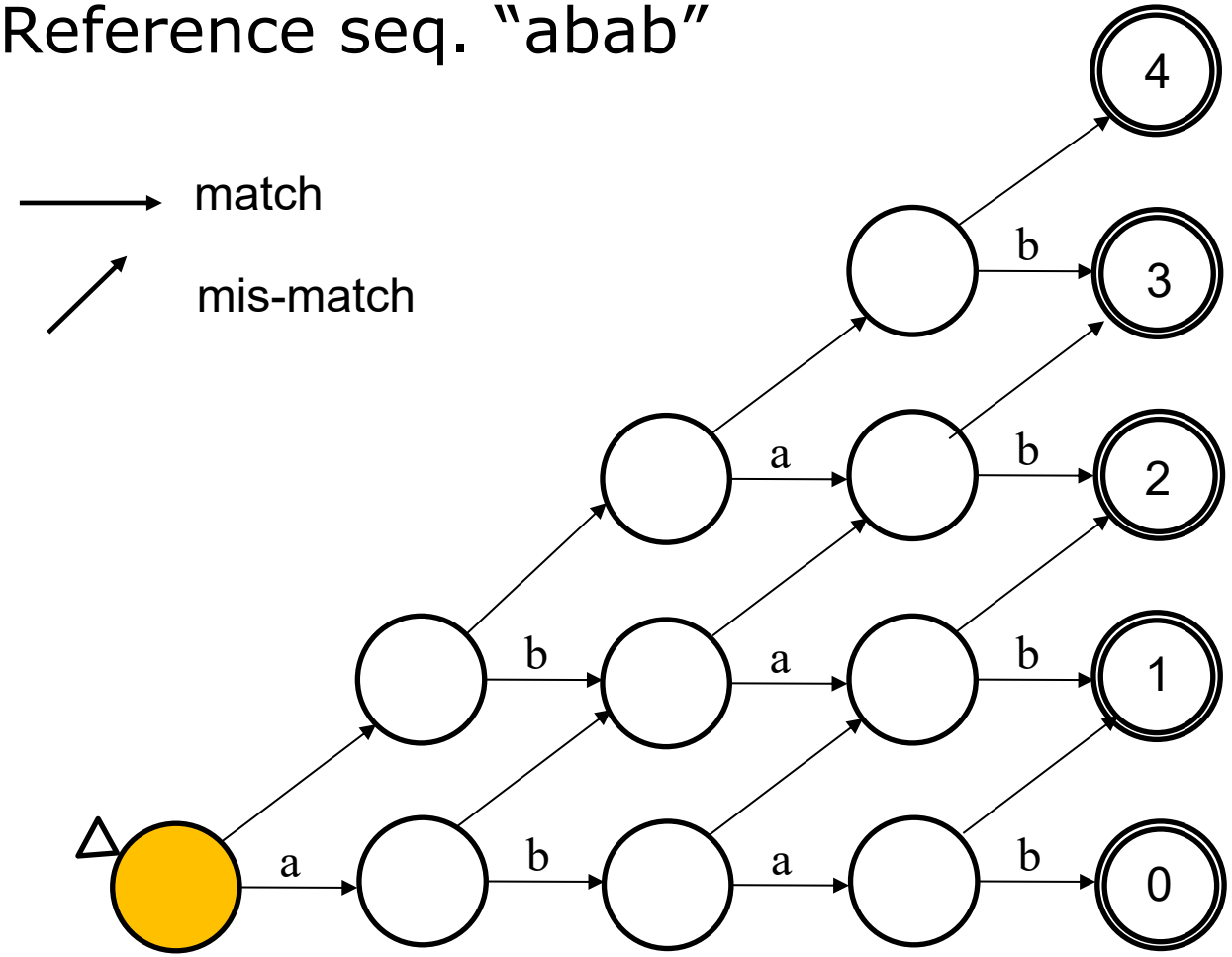
- Recognize pattern: “ababc”
- Input: “abababc”



Input	Active States
	0
a	0,1
b	0,2
a	0,1,3
b	0,2,4
a	0,3
b	0,4
c	0,5 (accept)

Hamming Distance

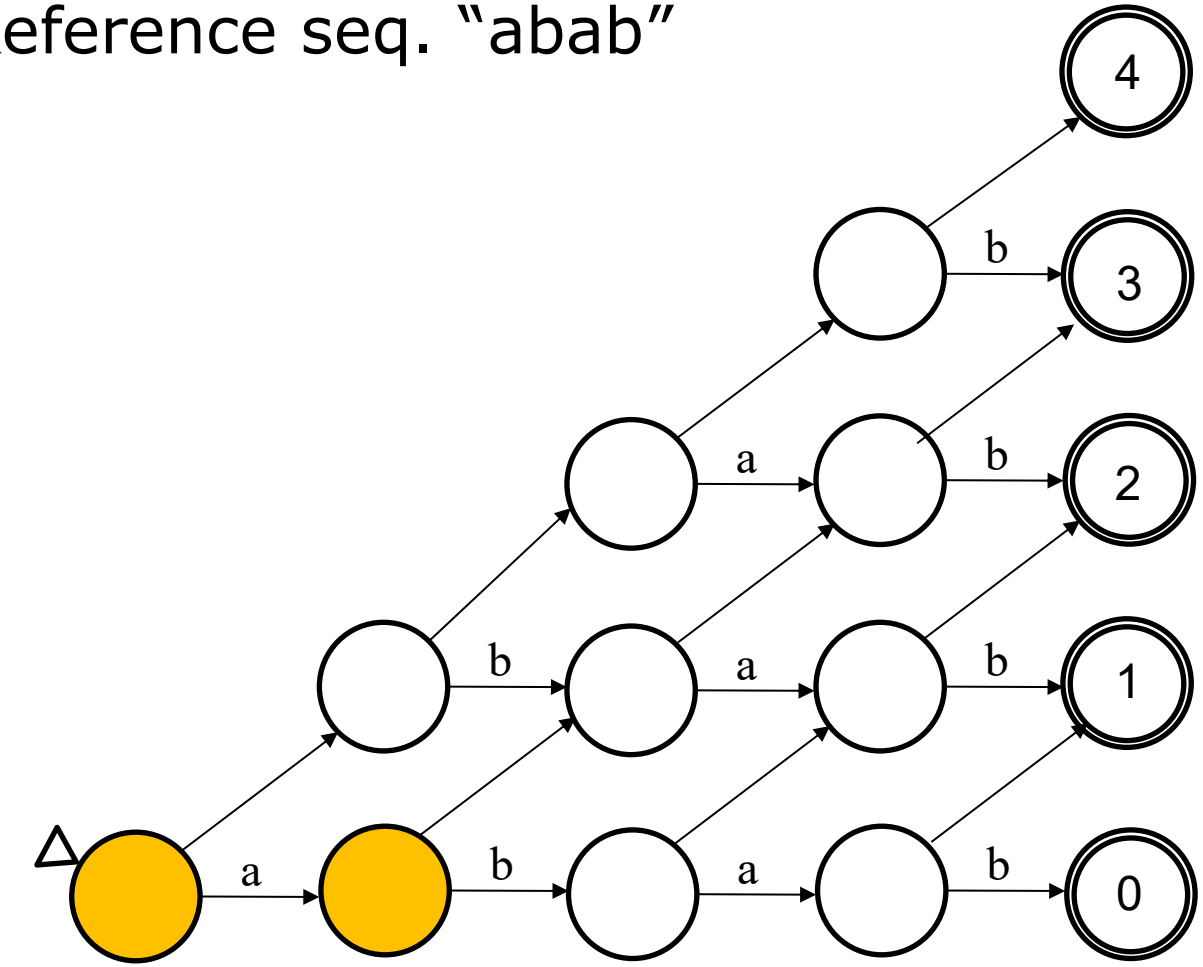
- Reference seq. “abab”



- Input seq. 1 “”

Hamming Distance

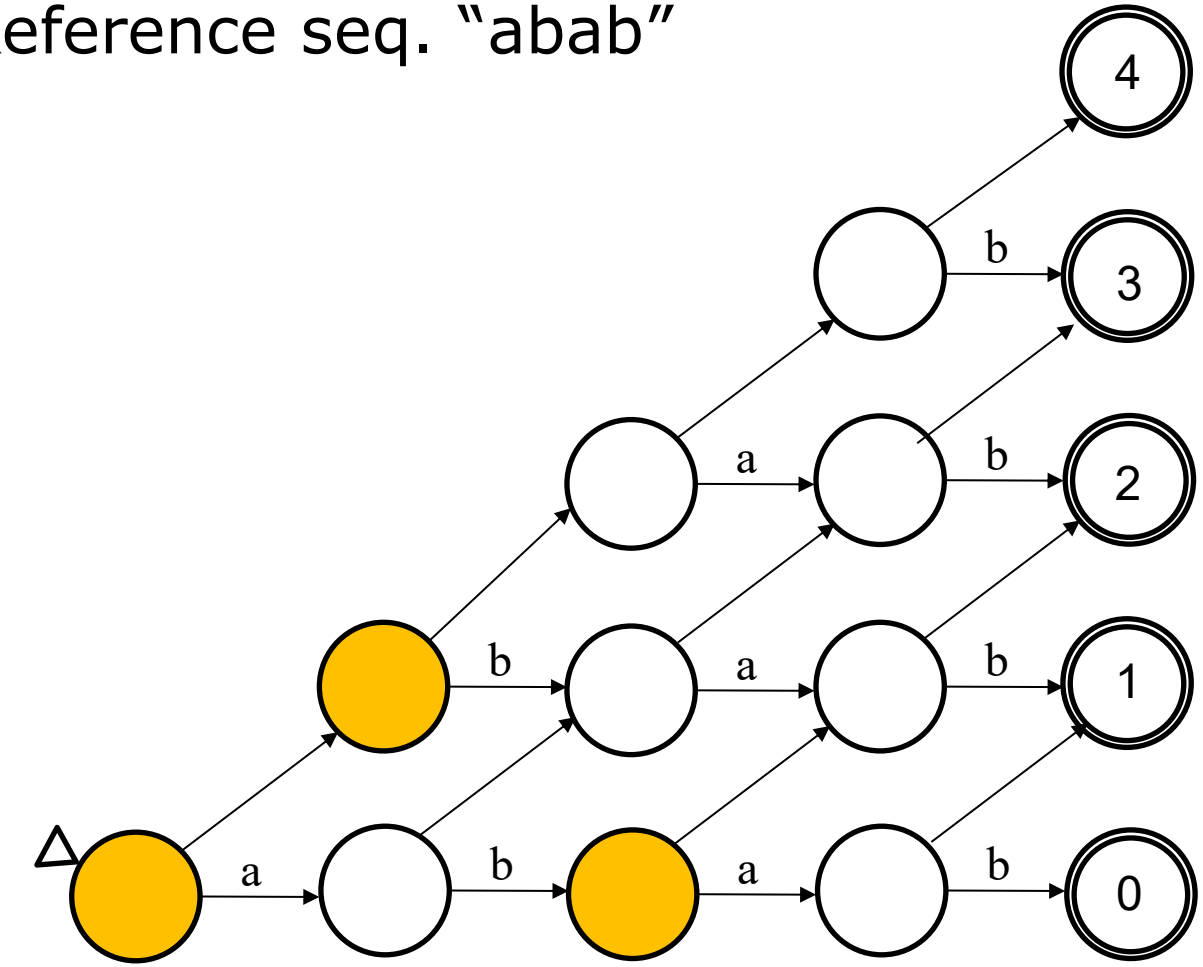
- Reference seq. “abab”



- Input seq. 1 “a”

Hamming Distance

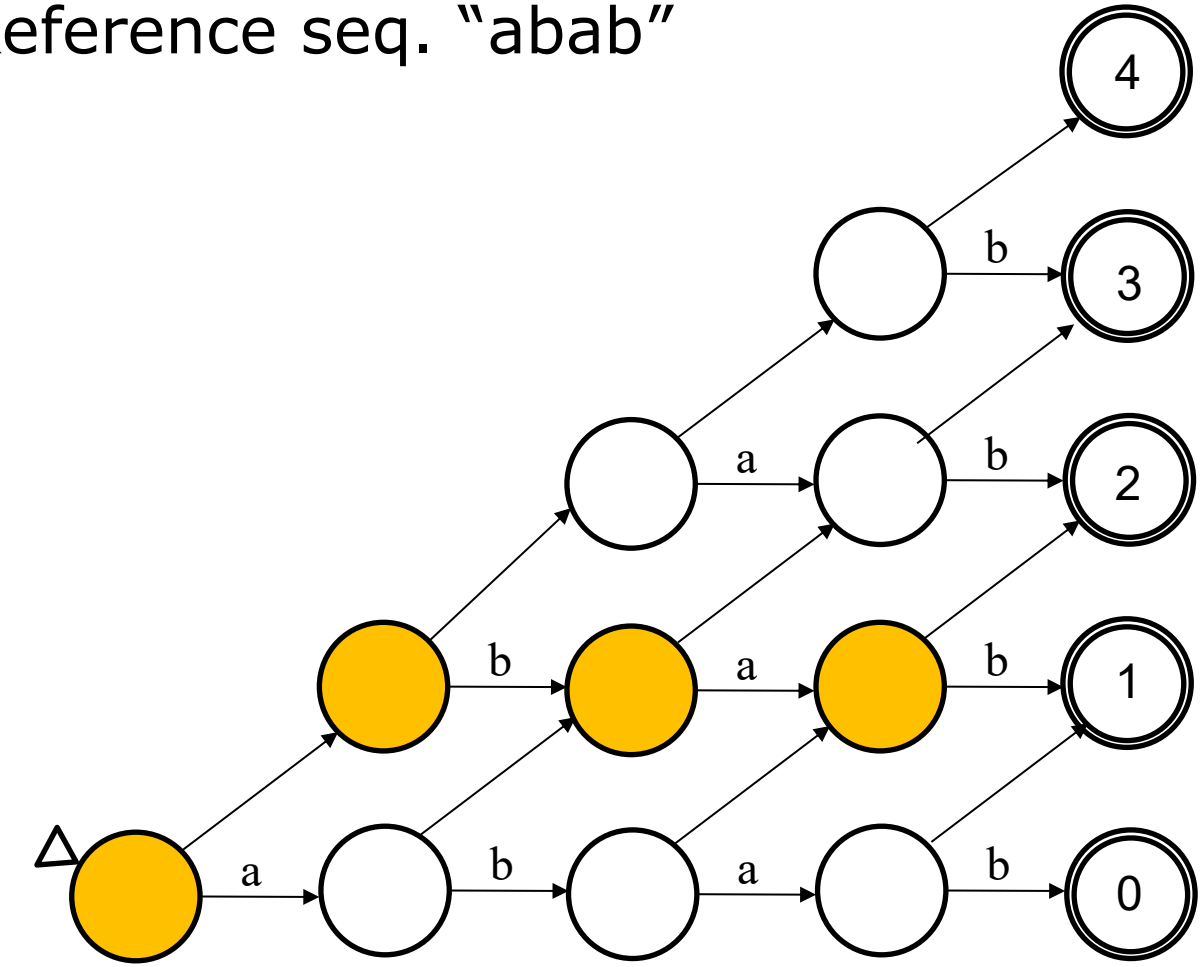
- Reference seq. “abab”



- Input seq. 1 “ab”
- Input seq. 2 “b”

Hamming Distance

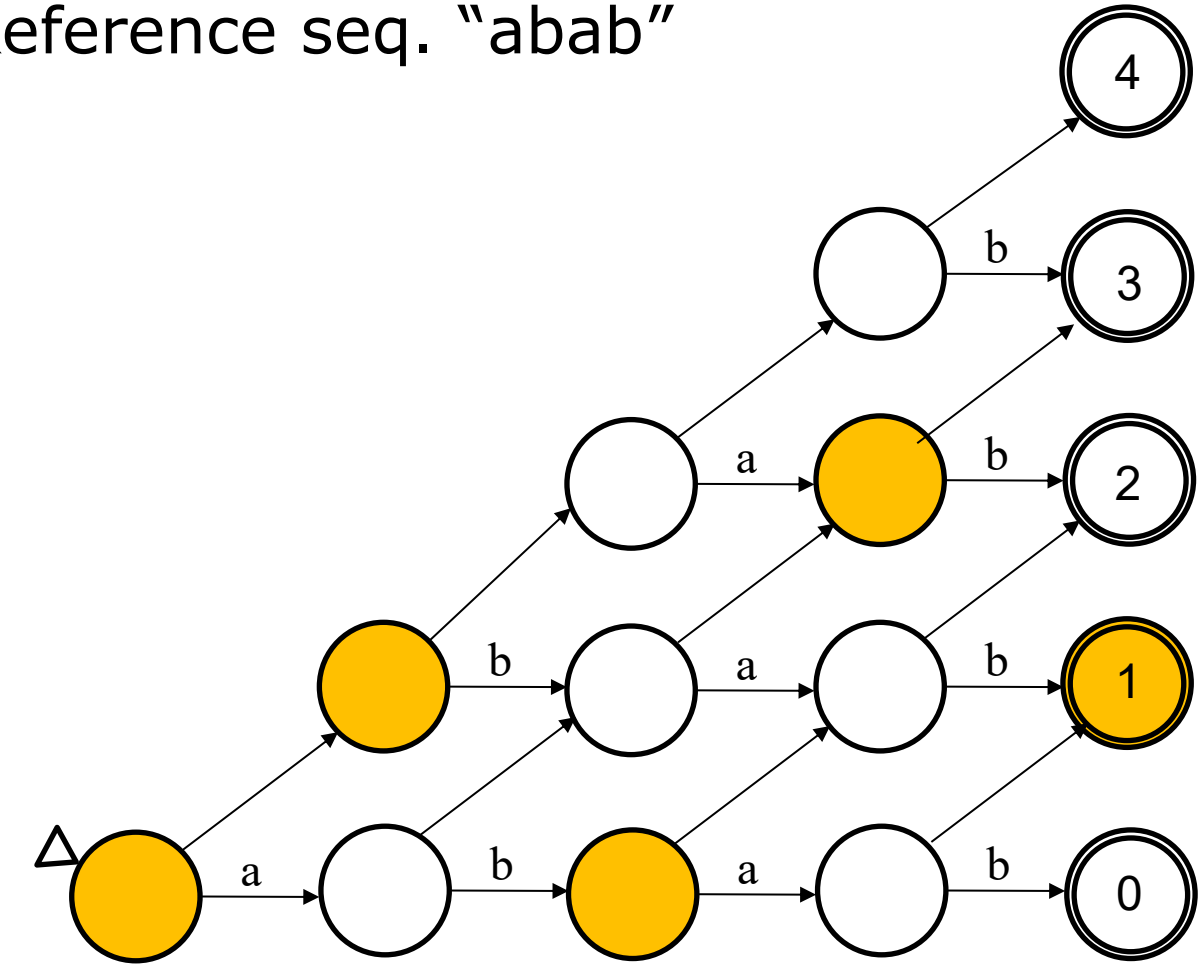
- Reference seq. “abab”



- Input seq. 1 “abb”
- Input seq. 2 “bb”
- Input seq. 3 “b”

Hamming Distance

- Reference seq. “abab”



- Input seq. 1 “abbb”
- Input seq. 2 “bbb”
- Input seq. 3 “bb”

Other Applications of NFAs

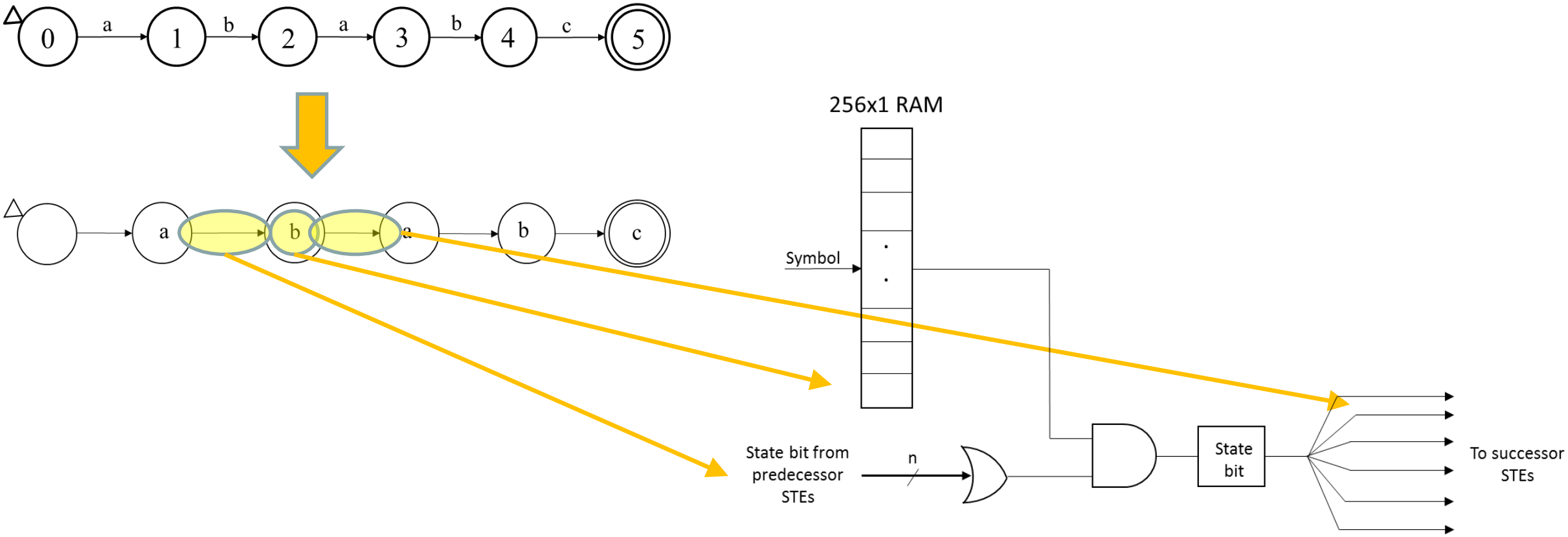
- Snort NID
- Motif finding
- Association rule mining
- Sequence distance
- Brill tagging

- Our approach: reusable NFA overlay
  - Similar to Micron Automata Processor

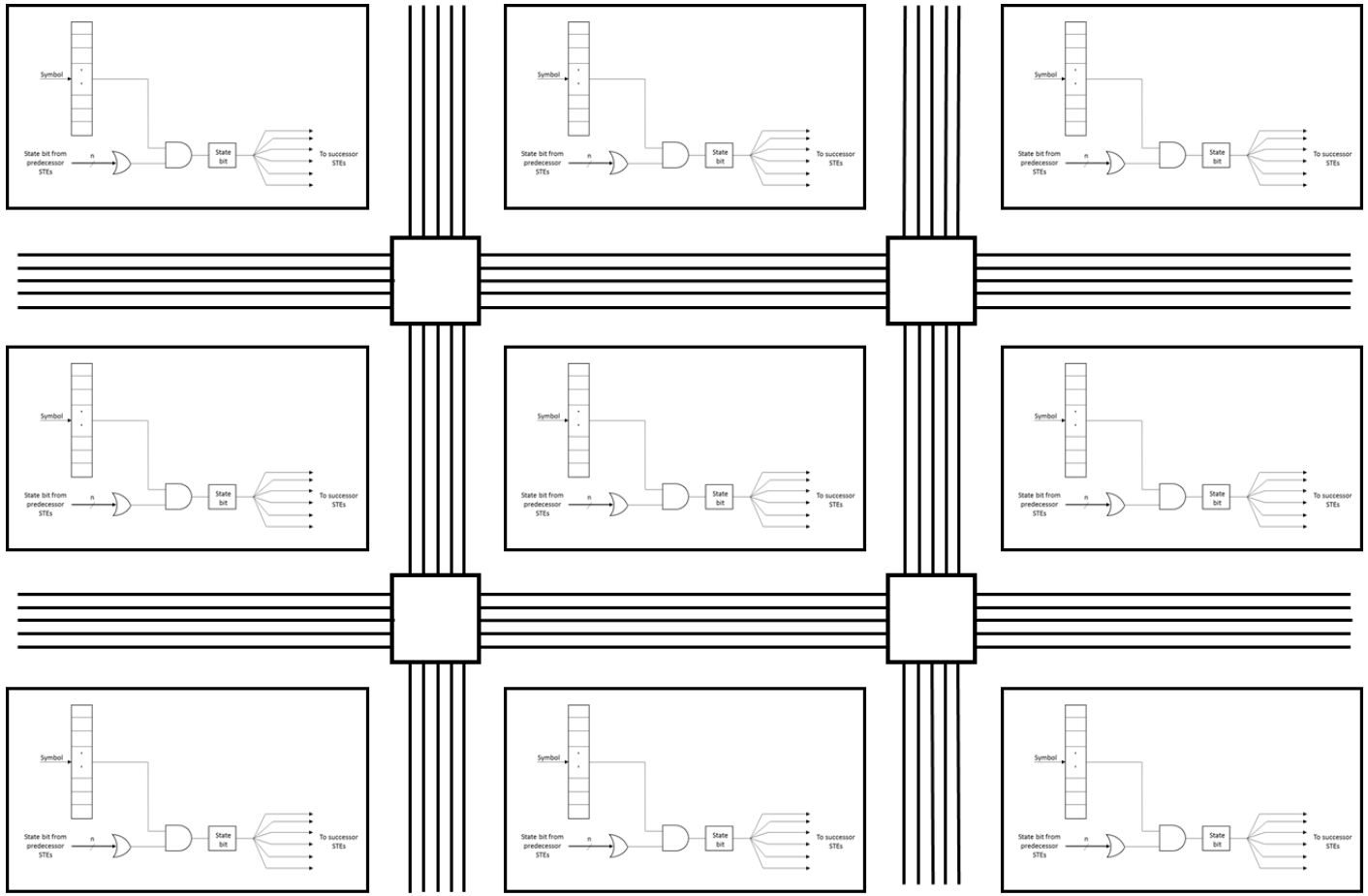
Micron Automata Processor (2013)

- Built on a DRAM substrate
- Basic element "State Transition Elements" (48K/chip)
- FPGA-like switched programmable interconnect

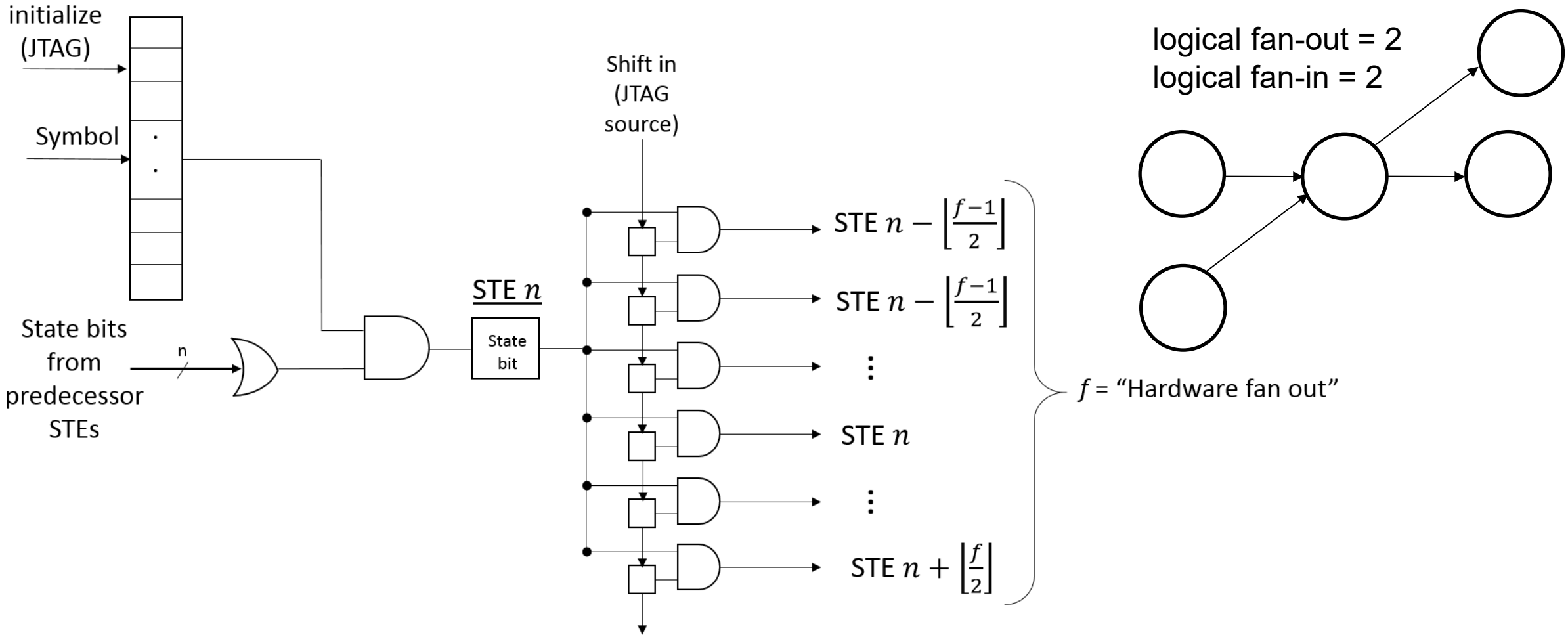
State Transition Element (STE)



Interconnection Network: Micron AP



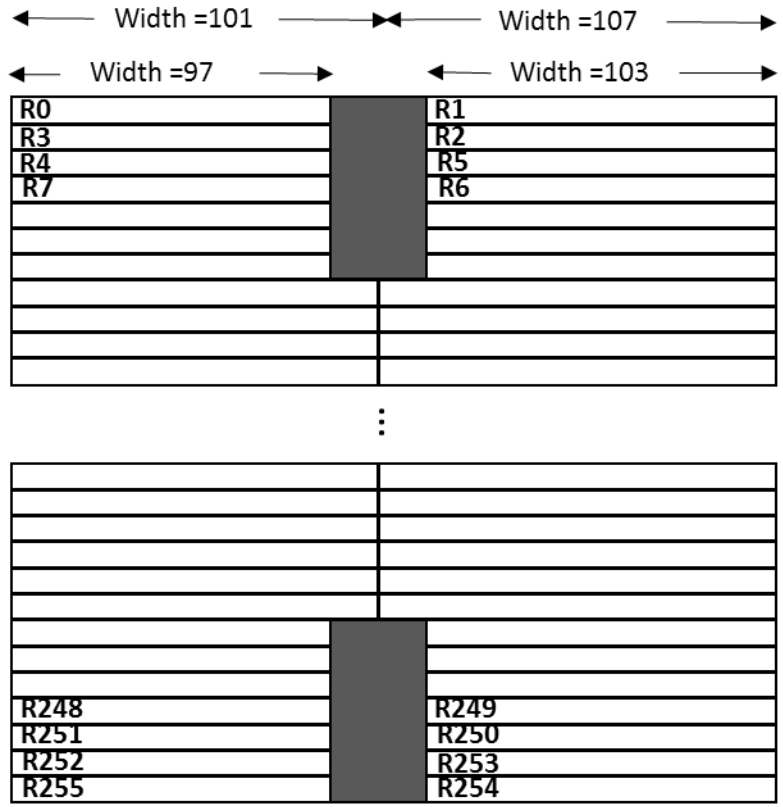
Overlay Interconnection Network



Location Constraints

- Merge next state tables into 256 x N RAMs
- Associate each group of N consecutive STEs with a location constraint
- Assigned each group in a zig-zag pattern

Stratix 5 A7



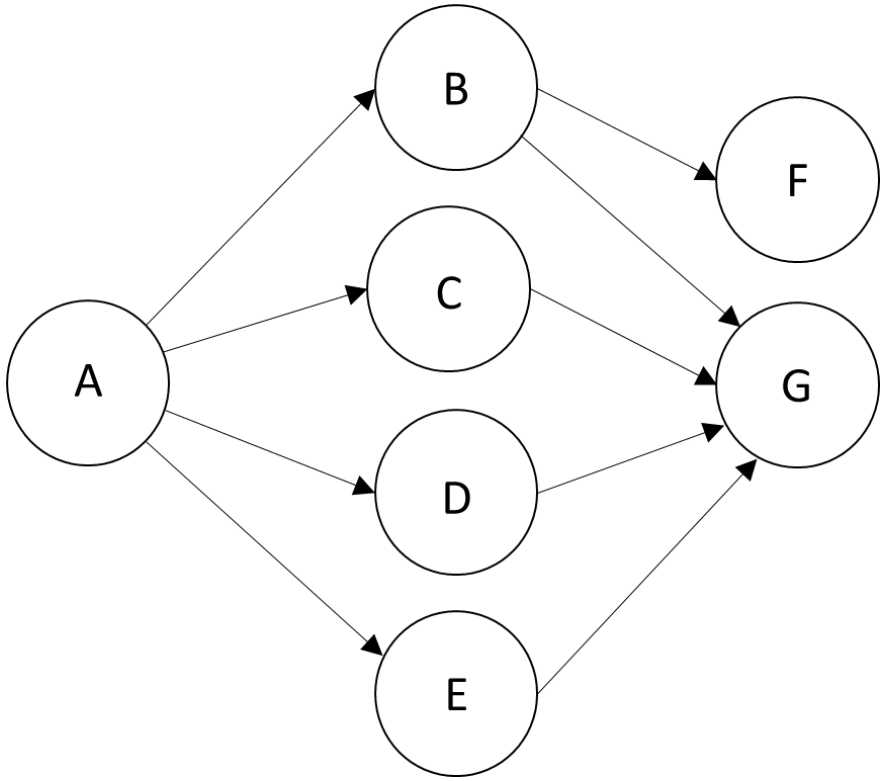
Hardware Cost

STEs (K)	Max. H/W Fan-out	Fmax (MHz)	ALMs	MLAB mem. (Mbits)	Reg. (Kbits)	Total mem. (MB)
4	24	152	42%	1	104	0.6
8	24	136	77%	2	208	1.6
12	23	122	95%	3	300	2.3
16	14	121	96%	4	256	2.9
20	8	119	93%	5	200	3.4
24	5	112	95%	6	168	4.0

Hardware Cost (16K STEs)

H/W Fan-out	Block intrcon't	Local interconnect	R24	R3	R6	# LABs utilized
6	25%	21%	25%	14%	25%	81%
10	34%	25%	33%	18%	35%	94%
11	35%	27%	31%	20%	36%	95%
12	42%	29%	38%	29%	46%	97%
14	44%	32%	41%	30%	47%	98%

Physical Mapping



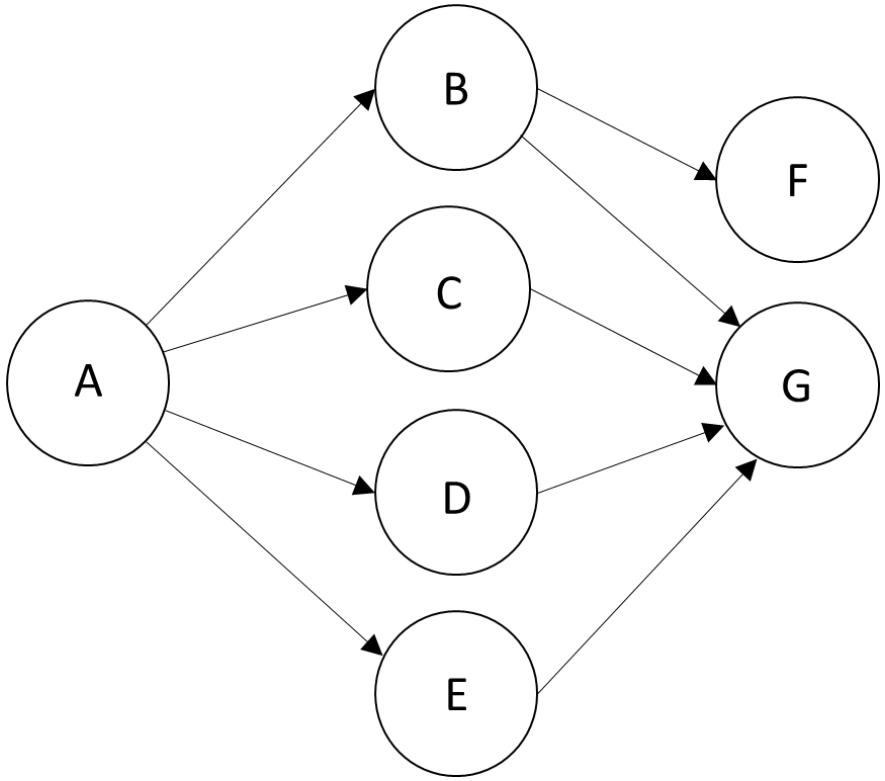
- Logical fan-in = 4
- Logical fan-out = 4

- Hardware fanout = 9
  - range = [-4,4]

- STE Mapping:

STE	State
0	A
1	B
2	C
3	D
4	E
5	F
6	G

Physical Mapping

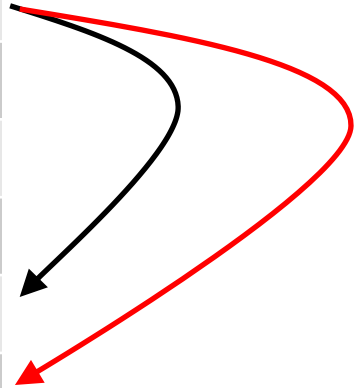


- Logical fan-in = 4
- Logical fan-out = 4

- Hardware fanout = 9
  - range = [-4,4]

- STE Mapping:

STE	State
0	A
1	B
2	C
3	D
4	E
5	F
6	G



Mapping Algorithm

1. Initially map states to STEs in order listed in ANML file
2. For each edge S -> D , if there is a mapping violation, move either S or D in a way that minimizes the resulting score

STE	State
0	A
1	B
2	C
3	D
4	E
5	F
6	G

Edge	Distance	Edge	Distance
A->B	1	B->G	5
A->C	2	C->G	4
A->D	3	D->G	3
A->E	4	E->G	2
B->F	4		

Option 1:

Move B to STE 2

STE	State
0	A
1	C
2	B
3	D
4	E
5	F
6	G

Edge	Distance	Edge	Distance
A->B	+1	B->G	-1
A->C	-1	C->G	+1 (5)
A->D	0	D->G	0
A->E	0	E->G	0
B->F	-1		

Relative score = -1

(but one new violation)

Mapping Algorithm

1. Initially map states to STEs in order listed in ANML file
2. For each edge S -> D , if there is a mapping violation, move either S or D in a way that minimizes the resulting score

STE	State
0	A
1	B
2	C
3	D
4	E
5	F
6	G

Edge	Distance	Edge	Distance
A->B	1	B->G	5
A->C	2	C->G	4
A->D	3	D->G	3
A->E	4	E->G	2
B->F	4		

Option 2:

Move B to STE 3

STE	State
0	A
1	C
2	D
3	B
4	E
5	F
6	G

Edge	Distance	Edge	Distance
A->B	+2	B->G	-2
A->C	-1	C->G	+1 (5)
A->D	-1	D->G	+1
A->E	0	E->G	0
B->F	-2		

Relative score = -2

(but one new violation)

Mapping Algorithm

1. Initially map states to STEs in order listed in ANML file
2. For each edge S -> D , if there is a mapping violation, move either S or D in a way that minimizes the resulting score

STE	State
0	A
1	B
2	C
3	D
4	E
5	F
6	G

Edge	Distance	Edge	Distance
A->B	1	B->G	5
A->C	2	C->G	4
A->D	3	D->G	3
A->E	4	E->G	2
B->F	4		

Option 6:

Move G to STE 5

STE	State
0	A
1	B
2	C
3	D
4	E
5	G
6	F

Edge	Distance	Edge	Distance
A->B	0	B->G	-1
A->C	0	C->G	-1
A->D	0	D->G	-1
A->E	0	E->G	-1
B->F	+1 (5)		

Relative score = -3

(but one new violation)

Mapping Algorithm

1. Initially map states to STEs in order listed in ANML file
2. For each edge S -> D , if there is a mapping violation, move either S or D in a way that minimizes the resulting score

STE	State
0	A
1	B
2	C
3	D
4	E
5	F
6	G

Edge	Distance	Edge	Distance
A->B	1	B->G	5
A->C	2	C->G	4
A->D	3	D->G	3
A->E	4	E->G	2
B->F	4		

Option 11:

Move G to STE 0

STE	State
0	G
1	A
2	B
3	C
4	D
5	E
6	F

Edge	Distance	Edge	Distance
A->B	0	B->G	-3
A->C	0	C->G	-1
A->D	0	D->G	+1
A->E	0	E->G	+3 (5)
B->F	0		

Relative score = 0

(but one new violation)

Mapping Algorithm

Movement	Relative Score	# violations	Movement	Relative Score	# violations
State B from STE 1 to STE 2	-1	1	State G from STE 6 to STE 5	-3	1
State B from STE 1 to STE 3	-2	1	**State G from STE 6 to STE 4**	**-5**	**2**
State B from STE 1 to STE 4	-3	1	State G from STE 6 to STE 3	-5	2
State B from STE 1 to STE 5	-3	2	State G from STE 6 to STE 2	-3	2
State B from STE 1 to STE 6	-4	1	State G from STE 6 to STE 1	0	1
			State G from STE 6 to STE 0	0	1

Results

ANML Benchmarks	#STEs	Maximum Logical Fan-in	Maximum Logical Fan-out	Minimum Hardware Fan-out Achieved
Brill	26668	4	4	42
Clam AM	49538	11	2	22
Levenshtein	2784	8	5	22
Hamming	11346	4	2	85
SPM	100500	3	2	22
EntityResolution	95136	28	5	200
RandomForest	75340	2	2	7
PowerEN	40513	4	3	cannot place

Results

ANML Benchmarks	#STEs	Maximum Logical Fan-in	Maximum Logical Fan-out	Minimum Hardware Fan-out Achieved
Snort (after removing special elements)	69029	19	19	cannot place
Fermi	40783	2	2	27
DotStar (after removing special elements)	96438	2	2	cannot place
Protomota (after removing special elements)	42061	3	9 (optimized)	cannot place

Conclusions

- AP overlay on moderate-sized FPGA (Stratix-5 GX A7) can fit ~1/2 the STEs on a Micron AP ASIC, using 95% of its LAB/MLAB resources
- Abstracted programmable interconnect is non-switched, point-to-point, relies on mapping algorithm to resolve routing constraints
- Proposed mapping algorithm maps most ANMLZoo benchmarks but generally requires more interconnect complexity than is feasible
- Future work: improve mapping algorithm, leverage NFA partitioning