JOINT INSTITUTE
交大密西根学院

Accelerating Stochastic Computation for Binary Classification Applications

Lezhong Huang[†], Guanhui Chen[†], Peng Li [‡], Weikang Qian[†]

[†]University of Michigan-SJTU Joint Institute,
Shanghai Jiao Tong University, China

[‡]Intel Corporation, USA

Mar. 25, 2016
ICASSP, Shanghai, China

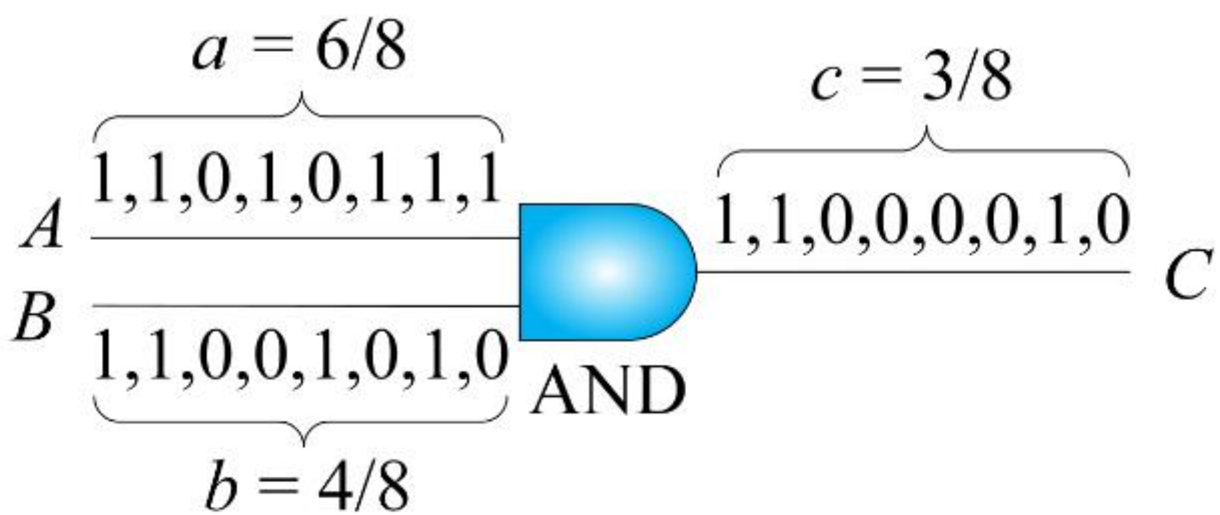
Outline

- Introduction of Stochastic Computing
- Accelerating Binary Classification Applications
- Experimental Results and Conclusion

Stochastic Computing (SC)

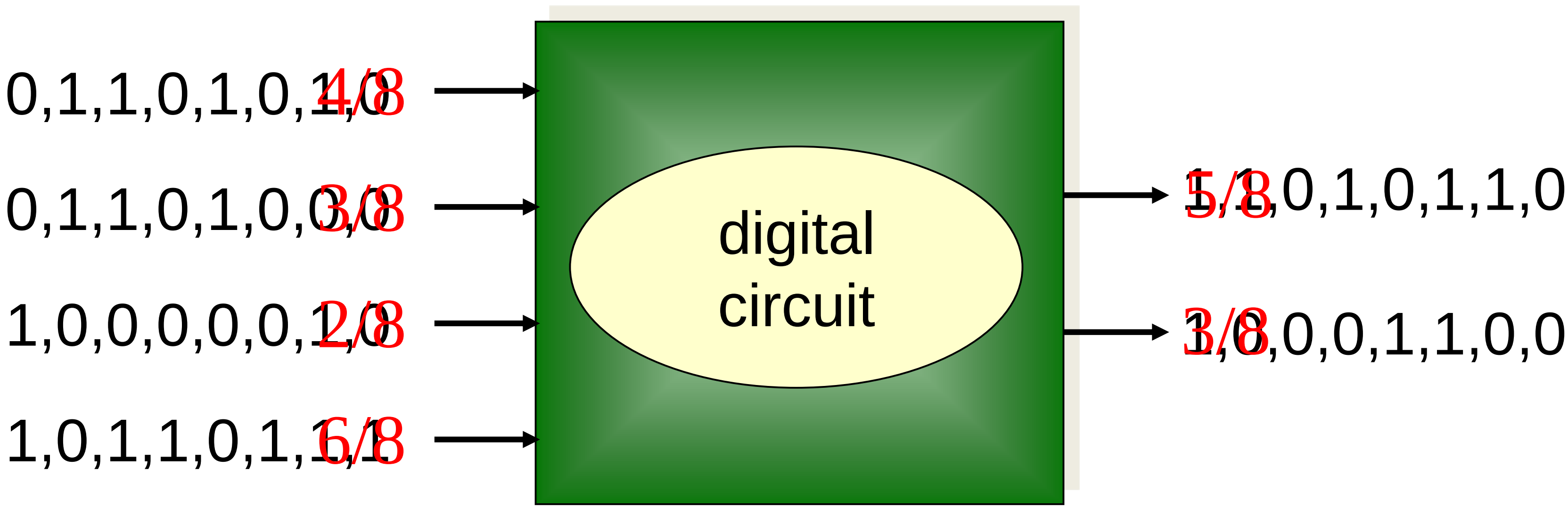
- Use digital circuit to process stochastic bit streams
- Each stochastic bit stream represents a value equal to the probability of a 1 in the stream

Example: Multiplication with an AND gate



Stochastic Computing (SC)

Probability values <u>are</u> the input and output signals



Digital circuits now compute on "probabilities"

Advantage of SC

- Simple digital circuit for many arithmetic functions

- Fault tolerance $\underbrace{1010111001}_{0.6} \rightarrow \underbrace{1010011001}_{0.5}$

  – In contrast, binary radix encoding

$$\underbrace{(1010)_2}_{10} \rightarrow \underbrace{(0010)_2}_{2}$$

Applications of SC

- Artificial neural networks
  - Tomberg and Kaski, 1990; Brown and Card, 2001
- Image processing
  - Li and Lilja, 2011; Alaghi *et al.*, 2013
- Decoding modern error-correcting codes (LDPC, turbo code, polar code, etc.)
  - Tehrani *et al.*, 2008; Dong *et al.* 2010; Naderi *et al.*, 2011

Precision versus Bit Length

- Binary radix encoding

  Example: $(1001)_2 \rightarrow 9$

  – Positional and compact:

    To represent 2^n different values, need n bits

- Stochastic encoding

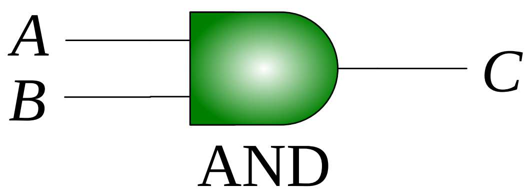
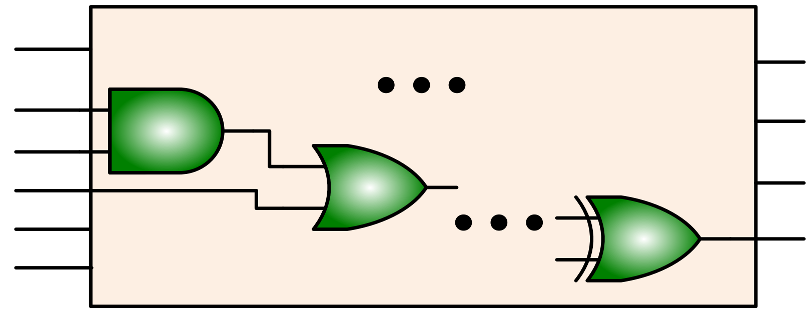
  Example: $(0101100010) \rightarrow 0.4$

  – Uniform and not compact:

    To represent 2^n different values, need 2^n bits

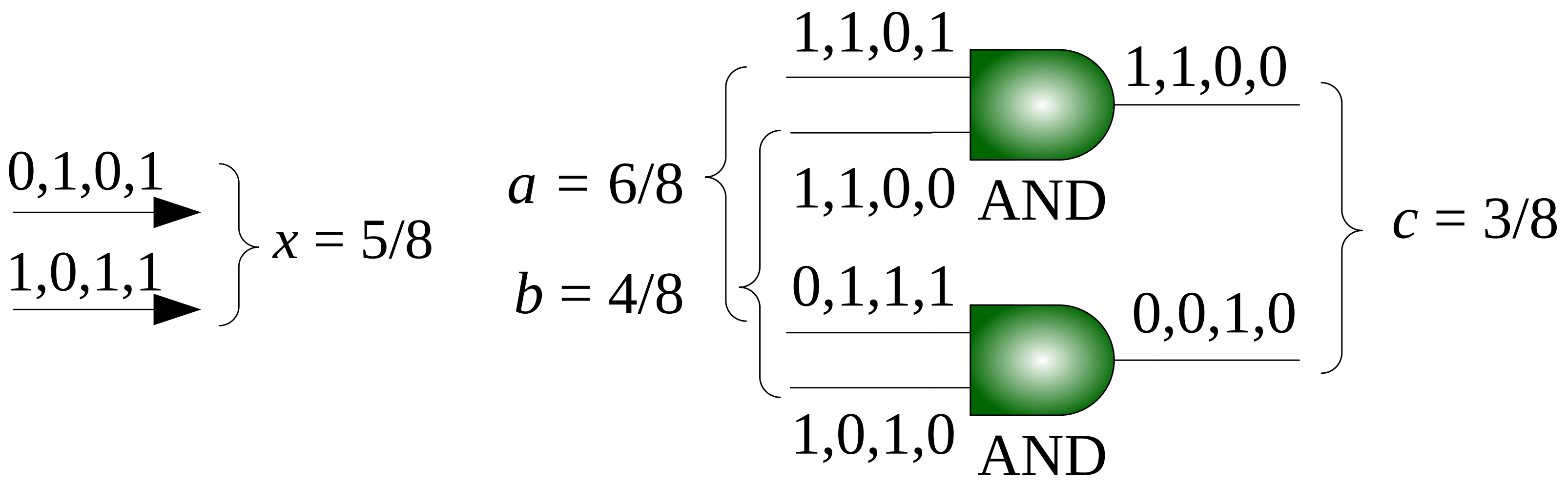
Target at applications that do not require high precision or can tolerate small errors. We don’t need a large n.

Computation Time Comparison

	Stochastic Implementation		Conventional Implementation
			
Critical Path			
Clock Period	T_S	$\ll$	T_C
Computation Time	NT_S	$\gg$	T_C

$$N = 2^n \gg 1$$

Reducing Computation Time: Parallel Implementation



Trade-off area with time

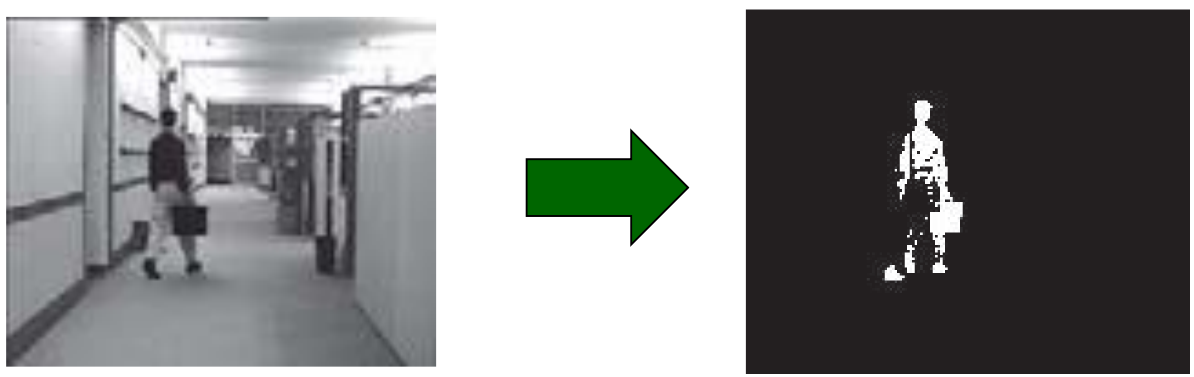
<u>Question</u>: in what situation, can we even reduce time without sacrificing area?

Outline

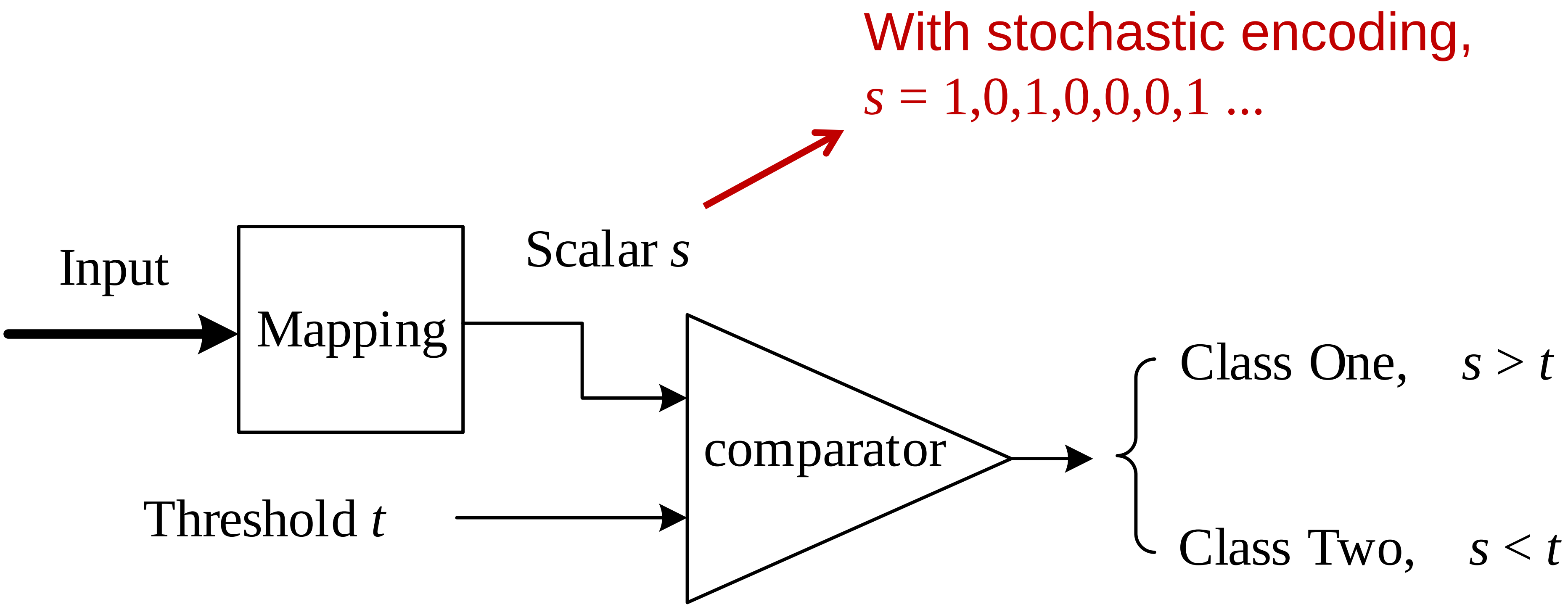
- Introduction of Stochastic Computing
- Accelerating Binary Classification Applications
- Experimental Results and Conclusion

Our Contribution

- Accelerate stochastic computation used for binary classification applications
  - E.g., image segmentation



Binary Classification



Rationale of Acceleration

$s = 1,0,1,0,0,0,1 \ldots$

- Only need to know if $\mathrm{Prob}(s) > t$
  - No need to know the exact $\mathrm{Prob}(s)$

- Estimate if $\mathrm{Prob}(s) > t$ based on a sub-sequence
  - If the probability of 1's in an L-bit sub-sequence is $\gg t$ (or $\ll t$), we are "pretty sure" that $\mathrm{Prob}(s) > t$ (or $< t$)
  - If we can infer the result from the L-bit segment, we stop the computation

$$\ldots,\overbrace{1,1,1,0,1,1,1,1}^{L=8},\ldots \quad \Rightarrow \quad \mathrm{Prob}(s) > t$$

$t = 0.5$

Methodology

- Transform comparison with t to comparison with 0.5
- To reduce error, choose a "larger" threshold $p > 0.5$
  - Infer the result only when we are "pretty sure"
- If cannot infer, continue to check next L bits

$$\boxed{p = 0.8}$$

$\ldots,\overbrace{1,1,1,0,1,1,1,1}^{L=8},\ldots \Rightarrow \mathrm{Prob}(s) > p \Rightarrow \mathrm{Prob}(s) > 0.5$

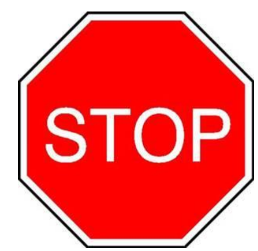
$\ldots,0,1,0,0,0,0,0,0,\ldots \Rightarrow \mathrm{Prob}(s) < 1 - p \Rightarrow \mathrm{Prob}(s) < 0.5$

$\ldots,1,0,1,0,1,0,1,1,\ldots \Rightarrow 1 - p < \mathrm{Prob}(s) < p \Rightarrow$ No inference

Methodology

$$\overbrace{1,\ 0,\ \ldots,\ 1,}^{L}\ \overbrace{0,\ 1,\ \ldots,\ 0,}^{L}\ \overbrace{1,\ 1,\ \ldots,\ 1,}^{L}\ \ldots$$

Can we infer?
($\mathrm{Prob} > p$, or
$\mathrm{Prob} < 1 - p$)



If all segments are checked but fail to infer, obtain $\mathrm{Prob}(s)$ and compare it with 0.5.

Design Parameters

- Two important parameters:
  - The segment length L
  - The threshold p
- They affect error rate and mean computation time
  - E.g., if $p \uparrow$, error rate $\downarrow$, but mean computation time $\uparrow$
- Choice of L and p
  - Error rate and mean computation time can be expressed in terms of L and p
  - Formulate and solve an optimization problem: Given error rate limit, minimize the mean computation time

Outline

- Introduction of Stochastic Computing
- Accelerating Binary Classification Applications
- Experimental Results and Conclusion

Case Study

- Accelerating stochastic implementation of kernel density estimation (KDE)-based image segmentation algorithm

$$\mathrm{PDF}(X_t) = \frac{1}{n}\sum_{i=1}^{n} e^{-4|X_t - X_{t-i}|}$$

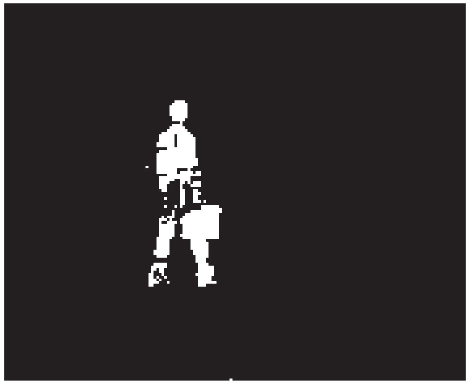
If $\mathrm{PDF}(X_t) < Th$, the pixel is background; otherwise, it is foreground

Case Study: Image Segmentation

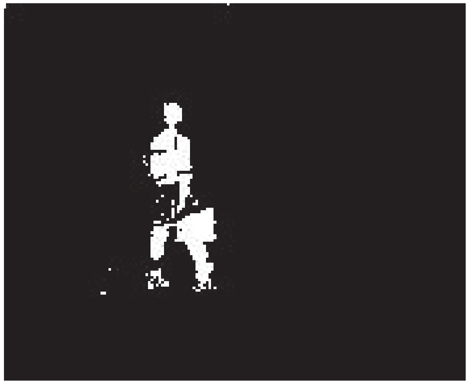
- Original stochastic implementation: 1024 bits
- Accelerated stochastic implementation
  - Design parameters: $L = 104$, $p = 0.654$
  - Average # bits checked = 211; Speedup = 4.86
  - Error rate = 0.347%

Input Image

Baseline Conventional Impl.



Accelerated Stochastic Impl.



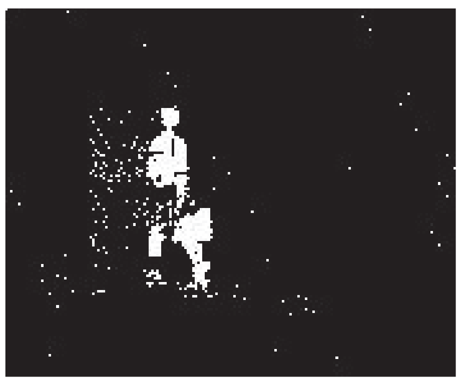
Case Study: Image Segmentation

- Compare to stochastic implementation using a simple acceleration method: check first k bits.

Proposed method

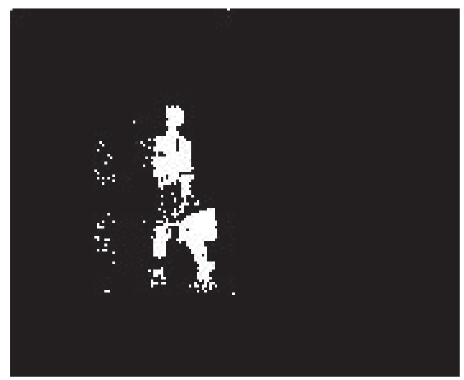
Speedup: 4.86x
Error rate:0.35%

First 150 bits



6.83x
1.52%

First 250 bits



4.10x
0.71%

First 350 bits

2.93x
0.50%

Case Study: Image Segmentation

- Hardware resource usage comparison
  - Synthesized on FPGA

	Basic Stochastic Implementation	Accelerated Stochastic Implementation	Overhead (%)
# Slices	6581	6631	0.76
# Slices Flip Flops	5640	5686	0.82
# LUTs	12230	12354	1.01

Conclusion

- Accelerate stochastic computation for binary classification applications
  - Only need to know whether probability of a bit stream is larger than a threshold, no need to get exact value
  - Method: check L-bit sub-sequences one by one until we can infer the result
  - Case study on KDE-based image segmentation application shows the effectiveness

Thank You!

Questions?