Introduction to Stochastic Computing and its Challenges

(Invited Paper)

John P. Hayes
Department of Electrical Engineering and Computer Science
University of Michigan,
Ann Arbor, MI 48109, USA
jhayes@eecs.umich.edu

ABSTRACT

We give a short overview of stochastic computing (SC) and its uses. SC computes with randomized bit-streams that loosely resemble the neural spike trains of the brain. Its key feature is the use of low-cost and low-power logic elements to implement complex numerical operations in a highly error-tolerant fashion. These advantages must be weighed against SC's inherently slow computing speed and low precision. Although studied sporadically since its invention in the 1960s, SC has regained interest recently as potentially suited to some emerging nanotechnologies, and to applications such as ECC decoding and biomedical image processing. However, a number of major challenges must be overcome if this potential is to be fully realized.

Categories and Subject Descriptors

B.2 [**Arithmetic and Logic Structures**]: Design styles.

General Terms

Algorithms, design, theory.

Keywords

Approximate, computing, logic design, neural networks, stochastic computing.

1. INTRODUCTION

The neural signals processed by the brain consist of long sequences of noisy voltage spikes of the kind shown in Figure 1 which resemble 0-1 sequences or *bit-streams*. The timing of the spikes appears random, but a significant amount of the information they convey is in their rate or frequency over some window of time [8]. Stochastic computing or SC [2] processes random bit-streams called *stochastic numbers* (SNs), whose information content lies in the frequency of the 1's. In SC's basic *unipolar* format, a bit-stream X of length N has the value $p_X = N_1/N$, where N_1 is the number of 1's appearing in X. For instance, the upper bit-stream in Figure 1 has the value 9/19. The positions of the 1's are not prescribed, so many different bit-streams can have the same p_X. X can be obtained from a stochastic number generator (SNG) of the kind shown in Figure 2*a*, which produces 1's and 0's with probabilities $p_X = B/2^k$ and $1 - p_X$, resp. In many applications, SC can be seen as computing with digitized probabilities defined by bit-streams.

DAC '15, June 07 - 11, 2015, San Francisco, CA, USA

http://dx.doi.org/10.1145/2744769.2747932

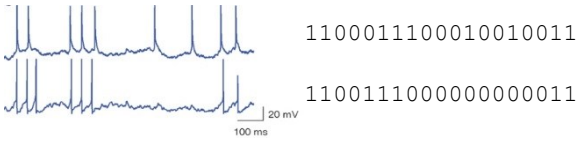

Figure 1: Neural spike trains and artificial bit-streams.

What made SC interesting when it first appeared in the 1960's [9] was its ability to perform complex numerical operations on SNs by means of tiny logic circuits. The iconic example is multiplication, which can be implemented by the single AND gate of Figure 2*b*. Suppose X and Y are applied to the AND, producing the bit-stream $Z = X \wedge Y$. A bit of Z is 1 if and only if X and Y are both 1, so that $p_Z \approx p_X \times p_Y$ provided X and Y are (nearly) statistically independent or un-correlated, and the bit-stream length N is sufficient. Another fundamental SC component appears in Figure 2*c*. A two-way multiplexer (MUX) plays the role of adder and calculates $p_z = 0.5(p_{x_1} + p_{x_2})$. The constant SN r = 0.5 applied to the MUX's select input ensures that p_z lies in the unit interval [0,1] and so can be treated as a probability.

Returning to the SNG of Figure 2*a*, it consists of a pseudo-random number source such as a linear feedback shift register (LFSR) and a magnitude comparator; it converts an unsigned k-bit binary number B to an N-bit stochastic bit-stream X. The SNG samples a signal of fixed probability $B/2^k$ and, at a rate of one bit per clock cycle, it produces an approximation $p_X \approx B/2^k$ whose precision tends to improve with N. However, to increase precision from k to $k + 1$ bits, N must be at least doubled. This points to a fundamental tradeoff associated with SC: the computing elements are orders of magnitude simpler than those of conventional (weighted binary) logic, but the operations themselves may take orders of magnitude longer to execute. For this reason, SC is best suited to applications requiring low numerical precision.

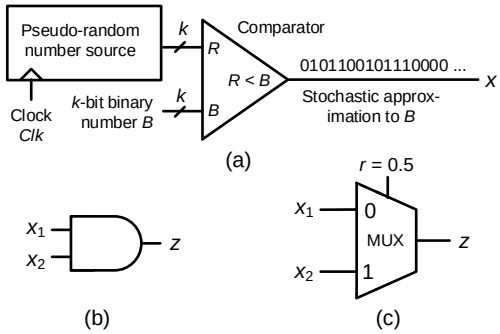

Figure 2: Key stochastic components: (a) stochastic number generator (SNG); (b) multiplier; (c) adder.

Signed arithmetic can be performed by interpreting the value of bit-stream X as $2p_X - 1$, called the *bipolar* value, rather than the unipolar value p_X. Thus, 01101000, which has the unipolar value 3/8 = 0.375, has the bipolar value –0.25. An XNOR gate multiplies two bipolar SNs, while a MUX again acts as a scaled adder. Other operations like division are more difficult to implement in SC and may require the use of sequential logic components [9]. SC can be summarized as an unconventional and approximate computing technique where bit-streams denote probability-style stochastic numbers, and logic circuits perform arithmetic operations.

For a particular SN format, such as unipolar, bipolar or inverse bipolar [1], every logic function $z(x_1,x_2,\ldots, x_n)$ implements a well-defined arithmetic function $Z(X_1,X_2,\ldots, X_n)$ which constitutes its stochastic behavior [4]. To see this, consider z's canonical sum-of-minterms form

$$z(x_1, x_2 \ldots, x_n) = \bigvee_{i=0}^{2^n-1} c_i \wedge m_i$$

Here c_i is 0 or 1, and m_i is a minterm $\tilde{x}_{i,1} \wedge \tilde{x}_{i,2} \wedge \cdots \wedge \tilde{x}_{i,n}$ where $\tilde{x}_{i,j}$ is $x_{i,j}$ or $\bar{x}_{i,j}$. For an XOR gate, we have $z_{\mathrm{XOR}}(x_1, x_2) = m_1 \vee m_2 = (\bar{x}_1 \wedge x_2) \vee (x_1 \wedge \bar{x}_2)$. Now suppose n SNs are applied to the inputs of z. Let X_i denote the (unipolar) value p_{X_i} of the SN applied to x_i, and let $\bar{X}_i$ denote $1 - p_{X_i}$. Assuming all the input SNs are independent, the stochastic function implemented by z is

$$Z(X_1, X_2 \ldots, X_n) = \sum_{i=0}^{2^n-1} c_i M_i$$

where $M_i = \widetilde{M}_{i,1}\widetilde{M}_{i,2} \cdots \widetilde{M}_{i,n}$ with $\widetilde{M}_{i,j} = X_{i,j}$ if the corresponding minterm m_i has $\tilde{x}_{i,j} = x_{i,j}$, and $\widetilde{M}_{i,j} = 1 - X_{i,j}$ if $\tilde{x}_{i,j} = \bar{x}_{i,j}$. Hence, the XOR gate realizes the multivariate polynomial $Z_{\mathrm{XOR}}(X_1, X_2) = X_1 + X_2 - 2X_1X_2$ in unipolar format, while it implements $-X_1X_2$, i.e., multiplication with negation, in bipolar.

2. APPLICATIONS

Up to now, the main advantage seen for stochastic circuits has been their small size, which indicates their suitability for applications needing massive numbers of small circuits. However, this low hardware cost may be offset by the need for large numbers of SNGs to create independent input SNs and subsequently re-randomize them to reduce the effects of correlation. Smallness also suggests low power, but the longer computation times of SC may result in higher energy costs [16].

The practical application of SC has been stymied in the past by the lack of general procedures to design and optimize stochastic circuits. Recent research has shed light on this problem, although much remains to be learned. Several general design approaches have appeared recently [1][17]. In [1], for instance, the spectral transforms of logic functions are used to synthesize circuit implementations of relatively low area.

SNs are subject to design-related errors from several sources [6]. These include insufficient bit-stream length, quantization errors during binary-to-stochastic number conversion, and correlation (insufficient independence) among interacting bit-streams. On the other hand, stochastic circuits tolerate environmental errors that seriously affect the behavior of conventional binary circuits. For example, flipping a bit, especially a high-significance bit, has a drastic effect on the accuracy of a binary circuit. On the other hand, flipping a few bits of a long SN X often has little effect on p_X. Moreover, if 0-to-1 and 1-to-0 bit-flips are equally likely, such errors tend to cancel. Thus SC seems very well-suited to applications like space-craft electronics which operate under severe, radiation-induced error conditions.

Early SC research mostly addressed applications having some or all of the following characteristics: small component size, low power, massive parallelism, low precision needs, and high error tolerance. It is worth noting that these are also typical features of analog computation [14], reflecting the fact that the probabilities represented by SNs are basically analog quantities. Among the very early applications of SC are artificial neural networks [5] and control systems [18], which have many analog features.

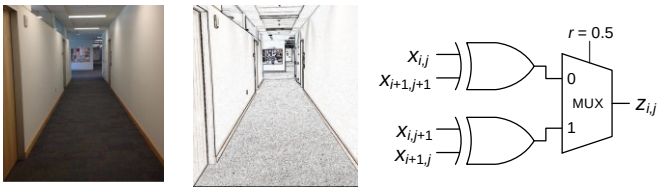

Figure 3. Illustration of edge detection: (a) original image; (b) processed image; (c) stochastic edge-detector circuit [3].

The revival of interest in SC since 2000 can be traced to the decoding needs of newer classes of error-correcting codes, especially low-density parity-check LDPC codes [10]. Because they handle very long code words (thousands of bits), LDPC decoders require extremely complex logic if implemented by conventional binary means; they also employ probabilistic decoding algorithms. These features suggest that LDPC decoders can exploit the small size, error tolerance, and probabilistic aspects of SC to achieve high area efficiency and throughput, as has been demonstrated very recently [12].

Another area to which SC has been applied with some success in recent years is image processing [3][13]. One such application is to the design of retinal implants [15]. These are ICs that are surgically placed in the eye to sense and process images, and transfer the results via electrical pulses directly to the brain through the optic nerve. SC circuits can meet the very severe size and power constraints imposed by retinal implants, while also performing useful image-processing tasks [3]. The implant comprises a large rectangular array of pixel sensors, each of which provides an input bit-stream $X_{i,j}$ to a local stochastic processor $SP_{i,j}$. This processor implements stochastic functions like the Roberts cross function

$$Z_{i,j} = 0.5 \times \left(\left|X_{i,j} - X_{i+1,j+1}\right| + \left|X_{i,j+1} - X_{i+1,j}\right|\right)$$

which performs edge detection on four bit-streams $X_{i,j}$, $X_{i+1,j}$, $X_{i,j+1}$ and $X_{i+1,j+1}$ obtained from neighboring pixels; see Figure 3. An efficient implementation of $Z_{i,j}$ is shown in Figure 3*c* [3]. Its cost is substantially less than that of a conventional, non-stochastic implementation of the same edge-detection algorithm.

3. CHALLENGES

Many questions concerning the theory and applicability of SC are still open fifty years after the technique was first introduced. Much of what we know at present applies to relatively small and mainly combinational circuits, reflecting the fact that few large-scale stochastic systems have actually been built.

SC is essentially a hybrid technology that combines various features of analog, digital and probabilistic computing, often in subtle ways. These features include the following:

1. *Circuit size*: The use of tiny components for addition and multiplication is SC's most obvious advantage. However, adders must be scaled to keep their results in in the unit interval, and single-gate multipliers need input bit-streams that are uncorrelated. This implies using large numbers of SNGs driven by independent random sources. SNGs are complex devices (Figure 2*a*) and can

account for as much as 80% of the total circuit cost [17]. Thus reduction of SNG count is a significant challenge for SC.

2. *Operating speed*: An obvious disadvantage of SC is the need for long SNs to achieve adequate numerical precision. This is compensated for to some extent by short clock cycles, and the possibility of dividing very long bit-streams into pieces that can be processed in parallel. Massive parallelism of this type has often been suggested, but its feasibility remains to be proven. Another interesting possibility needing further investigation is progressive precision, where bit-stream length is varied dynamically in response to application needs. Its potential value has been shown for some types of image processing [3].

3. *Power and energy*: The small size of stochastic components implies they have relatively low power needs. However, the power consumption of a circuit's SNGs also must be accounted for. Noting that *energy* = *power* × *time*, the long run-times of stochastic circuits can lead to higher energy use than their conventional counterparts [16]. Thus, minimizing energy usage in SC is an significant open problem.

4. *Design issues*: Considerable progress has been made in developing design methods for SC over the last few years [1][4][17], but challenging questions remain. Although every logic function z corresponds to some stochastic arithmetic function Z, as explained in Section 2, the converse is not true. Some functions, even simple ones like the sum $X_1 + X_2$, are not stochastically realizable. In such cases, the target function Z must be replaced by a scaled or approximate version Z^* that is stochastically realizable, but only heuristic ways to do this are known. Another little explored aspect of SC design is the fact that when the input streams include constant SNs (such as $r = 0.5$ in Figure 2*c*) two different logic functions z_1 and z_2 may have the same stochastic behavior Z [7]. Correlation, which tends to reduce the efficiency of multiplication and other stochastic operations, has been employed to decrease circuit size, as in the edge detector of Figure 3*c* [3]; general ways to exploit correlation are not known, however. Finally, the role of sequential logic in the implementation of stochastic functions has many open questions. Overall, SC appears to offer a richer and much more challenging range of design problems than traditional logic synthesis.

5. *Nanotechnologies*. Some of the nanoscale semiconductor technologies that are being investigated as possible replacements for CMOS circuits (in light of the anticipated demise of Moore's Law) seem well-suited to SC. These include memristors, which have been called inherently stochastic devices [11]. The success of these nanotechnologies in the construction of practical stochastic circuits remains to be seen, however. A related problem is how to take advantage of the similarity between SC and the neural bit-streams illustrated by Figure 1 in biomedical applications. Can stochastic processors be built that directly interact with the brain?

In summary, the central challenge facing SC is to determine cost-effective design techniques, technologies and applications that maximize SC's advantages (such as powerful basic operations, the possibility of massive parallelism, and inherent error tolerance) while minimizing its disadvantages (low precision, long bit-streams, and costly randomness requirements).

4. ACKNOWLEDGMENT

This work was supported by Grant CCF-1318091 from the National Science Foundation.

5. REFERENCES

[1] Alaghi, A. and Hayes, J.P. 2012. A spectral transform approach to stochastic circuits. In *Proc. Intl. Conf. Computer Design* (*ICCD*), 315-321.

[2] Alaghi, A. and Hayes, J.P. 2013. Survey of stochastic computing. *ACM Trans. Embedded Computing Systems*, 12, no. 2s, 92:1-92:19.

[3] Alaghi, A., Li, C. and Hayes, J.P. 2013. Stochastic circuits for real-time image-processing applications. In *Proc. Design Automation Conf.* (*DAC*), 1-6.

[4] Alaghi, A. and Hayes, J.P. 2015. On the functions realized by stochastic computing circuits. In *Proc. Great Lakes VLSI Symp.* (*GLSVLSI*). In press.

[5] Brown, B.D. and Card, H.C. 2001. Stochastic neural computation. I. Computational elements. *IEEE Trans. Comp.*, 50, 891-905.

[6] Chen, T-H., Alaghi, A. and Hayes, J.P. 2014. Behavior of stochastic circuits under severe error conditions. *Information Technology*, 56, 182-191.

[7] Chen, T-H. and Hayes, J.P. 2015. Equivalence among stochastic logic circuits and its application. In *Proc. Design Automation Conf.* (*DAC*). These proceedings.

[8] Cunningham, J.P. et al. 2011. Methods of estimating neural firing rates, and their application to brain-machine interfaces. *Neural Networks*, 22, 1235-1246.

[9] Gaines, B.R. 1969. Stochastic computing systems. In *Advances in Information Systems Science*, 2, 37-172.

[10] Gaudet, V. and Rapley, A. 2003. Iterative decoding using stochastic computation. *Electron. Letters,* 39, 299-301.

[11] Knag, P. et al. 2014. A native stochastic computing architecture enabled by memristors. *IEEE Trans. Nanotech*. 13, 283-293.

[12] Lee, X-R. et al. 2015. A 7.92 Gb/s 437.2 mW stochastic LDPC decoder chip for IEEE 802.15.3c applications. *IEEE Trans Circuits and Systems I: Regular Papers*, 62, 507-516.

[13] Li, P. et al. 2014. Computation on stochastic bit streams digital image processing case studies. *IEEE Trans. VLSI Systems*, 22, 449-462.

[14] MacLennan, B.J. 2009. Analog computation. In *Encyclopedia of Complexity and System Science*, Springer, 271-294.

[15] Mokwa, W. 2011. Retinal implants to restore vision in blind people. In *Proc. Transducers'11,* 2825-2830.

[16] Moons B. and Verhelst, M. 2014. Energy-efficiency and accuracy of stochastic computing circuits in emerging technologies. *IEEE Jour. Emerg. & Sel. Topics in Ccts. and Syst.*, 4, 475-486.

[17] Qian, W. et al. 2011. An architecture for fault-tolerant computation with stochastic logic. *IEEE Trans. Comp.*, 60, 93-105.

[18] Zhang, D. and Li, H. 2008. A stochastic-based FPGA controller for an induction motor drive with integrated neural network algorithms. *IEEE Trans. Industrial Electron*. 55, 551–561.