The Accuracy and Scalability of Continuous-Time Bayesian Inference in Analogue CMOS Circuits

Przemyslaw Mroszczyk and Piotr Dudek
School of Electrical & Electronic Engineering
The University of Manchester
Manchester, M13 9PL, United Kingdom
przemyslaw.mroszczyk@postgrad.manchester.ac.uk
p.dudek@manchester.ac.uk

***Abstract*—This paper discusses the idea of Bayesian inference in factor graphs implemented as continuous-time current-mode analogue CMOS circuits using Gilbert multipliers for arithmetic operations. The computational accuracy, accounting for the systematic and random (fabrication mismatch) errors, and the scalability of such realisations were verified in simulations of networks consisting of 5 - 121 nodes implemented using models from a standard 90 nm CMOS technology. The obtained results show a relatively short settling time, typically below 3 µs at a power less than 7 mW, with the equivalent computational speed of over 35 arithmetic operations per nanosecond but with a limited accuracy, mainly affected by fabrication mismatch. Such realisations could be used in applications requiring fast and low power approximate Bayesian inference.**

Keywords—Bayesian inference, belief propagation, factor graphs, Gilbert multiplier, analogue computation, CMOS

I. INTRODUCTION

A Bayesian network is a compact graphical model representing casual relations among a set of random variables. The network is defined by an acyclic graph with directed links and conditional probability tables (CPTs) encoding the "strength" of the probabilistic relations between nodes (Fig. 1). Such networks provide mathematical tools for plausible reasoning (Bayesian inference) under uncertainty based on the incorporated knowledge and the observational support, and can be used in decision support systems, information retrieval, pattern recognition, in various areas including robotics, bioinformatics or management. The computational task is to determine the conditional probabilities of random variables under constraints given by the priors and evidence. In terms of the computational complexity, Bayesian inference is an NP-hard problem and hence quickly becomes intractable for larger networks. To address this issue, apart from the computationally demanding inference methods such as global marginalisation, conditioning or clustering, some approximate solutions based on the Monte Carlo approach (e.g. Gibbs sampling) or the message-passing and belief propagation algorithms were proposed [1], [2]. For specific applications e.g. in artificial intelligence, signal processing and communication, the direct hardware realisation of belief propagation in analogue VLSI circuits has been suggested as much more advantageous than software solutions in terms of power and processing speed [5]-[7]. In particular, the sum-product scheme and the factor graphs for iterative decoding algorithms such as "turbo" codes and low-density parity-check (LDPC) codes [4] were successfully realised as continuous time circuits operating in the current mode domain and using arrays of Gilbert multipliers [5]. These decoders, however, were used for processing binary data which makes the computation, and the obtained results, less prone to errors caused by circuit nonlinearities, temperature and random device variability (fabrication mismatch). Belief propagation on signals representing probability distributions requires more precise operation and some preliminary research on that topic has been presented in [7], however this is limited to the analysis of a single network with only three nodes.

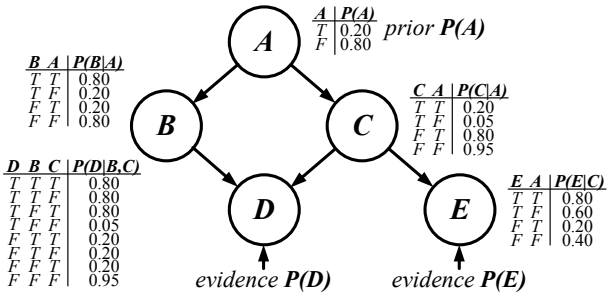


Fig. 1. Example of a Bayesian network from [1, p. 196] with variables *A*, *B*, *C*, *D* and *E* having two states: *true* (*T*) and *false* (*F*).

In this paper we provide a more detailed analysis of the factor graph realisations in standard CMOS technologies addressing the key aspects of VLSI design such as reliability under fabrication mismatch and scalability in terms of accuracy, power, area and circuit complexity. Guidelines towards more robust arithmetic circuit design are presented and the complexity scaling issues are discussed.

II. FACTOR GRAPHS

A Bayesian network can be represented as a factor graph where each Bayesian node consists of a pair of variable and factor nodes [4]. In the belief propagation scheme, nodes perform certain arithmetic operations related to Bayesian inference to calculate probabilities (beliefs) of the corresponding variables, and exchange computed results (messages) with their nearest neighbours. The messages circulate around the graph until convergence when no changes in the network state are observed. In a digital implementation this requires iterative computation whereas in an analogue circuit, the message exchange is continuous and the network settles to a solution. It is important to note that for networks

with loops where more than one path may exist between two nodes (e.g. nodes *A* and *D* in Fig. 1), the computed beliefs may only be approximations of Bayesian inference, with accuracy depending on the network, conditional probabilities and the inserted data [1], [2]. The circuit implementations of the factor graphs discussed in this paper are limited to 3-way factor and variable nodes operating on two-state variables with two element vectors representing normalised probabilities of *true* (*T*) and *false* (*F*). The mechanism of message passing and the arithmetic operations carried out by a 3-way node *N* (variable or factor) with neighbours X_1, X_2 and X_3 are presented in Fig. 2. The neighbours X_1-X_3 send messages *Xn* to node *N* and receive messages *Nx* from this node. In general, variable nodes are used for belief calculations and evidence insertion whereas factor nodes perform operations on messages using CPTs.

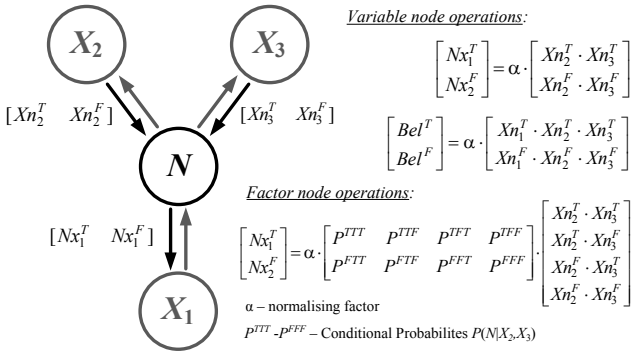


Fig. 2. Message passing scheme in factor graphs and the corresponding arithmetic operations for neighbour X_1.

III. CIRCUITS

The schematic diagram of the multiplier circuit used in this work for operations on real numbers is shown in Fig. 3. Similarly to previous works [5], [7], this implementation is based on a Gilbert multiplier [8] built on transistors M_1-M_4. Additional cascode current mirrors for the input current I_0 and the output currents I_{D1}, I_{D2}, and a modified biasing scheme are used to improve the computational accuracy. In the following we assume that the transistors M_1-M_4 operate in weak inversion (subthreshold) and saturation. The drain current I_D can be calculated using the following simplified equation [3]:

$$I_D = I_S \exp\left(\frac{\kappa V_{GS}}{U_T}\right) \tag{1}$$

where V_{GS} is the gate-source voltage, I_S is the characteristic current of MOS transistor, $U_T \approx 25.85$mV is the thermal voltage and κ is the slope factor. Based on (1), the diode-connected transistors M_3 and M_4 work as logarithmic *I-V* converters of the input currents I_{X1} and I_{X2} generating voltages $V_{X1} \sim \ln(I_{X1})$ and $V_{X2} \sim \ln(I_{X2})$, which in turn control the differential pair M_1-M_2. Due to the exponential characteristics of M_1 and M_2, the output currents I_{D1} and I_{D2} have the same proportions as the respective input currents I_{X1}, I_{X2} and their sum is constant and equal to I_0 (the input current providing the second multiplicand). The circuit from Fig. 3 performs multiplication with normalisation given by:

$$\begin{bmatrix} I_{D1} \\ I_{D2} \end{bmatrix} = \alpha \cdot I_0 \cdot \begin{bmatrix} I_{X1} \\ I_{X2} \end{bmatrix} \tag{2}$$

where $\alpha = 1/(I_{X1} + I_{X2})$ is the normalisation factor. This inherent normalisation is advantageous in the probabilistic calculus (e.g. in Bayesian inference), requiring computation on real numbers in the unity interval (0...1).

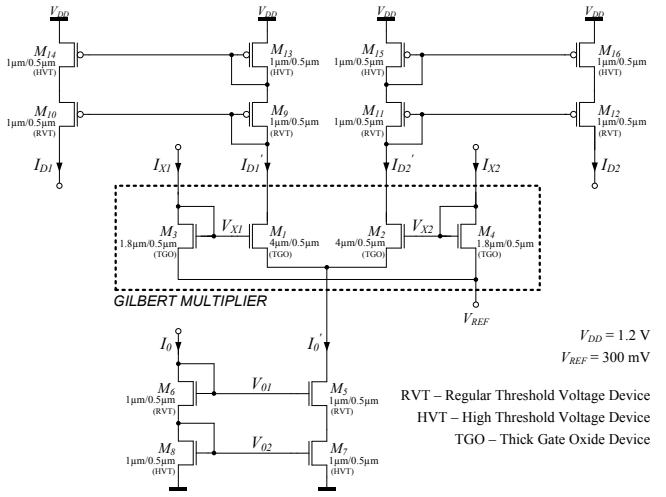


Fig. 3. Schematic diagram of the basic multiplier circuit used for analogue computations and implemented in a standard 90 nm CMOS technology.

The disparities between the actual and the ideal results observed in the multiplier circuit are caused by design issues (non-ideal current mirrors, operating point violations, leakage) and the non-ideally exponential characteristics of MOS devices in subthreshold region. Also, random parameter variability affects the symmetry of the circuit, which further increases the computational errors. In order to reduce the level of systematic errors, cascode current mirrors exhibiting higher output resistance and better linearity are used. They however require a higher voltage headroom necessary to keep transistors in saturation. This is critical especially for the bottom current source on transistors M_5 and M_7. Therefore, the sources of transistors M_3 and M_4 (the logarithmic *I-V* converters) are connected to the reference voltage V_{REF} (rather than directly to the ground potential) which can be used to adjust the operating point of the bottom current source preventing M_5 and M_7 entering the linear region. Also, the leakage currents of the MOS transistors in the current mirrors determine the lower bound of the operating signal level which can be correctly replicated. In the proposed solution, to assure the correct operation of the circuit for currents in range 1 nA - 1 µA (necessary to encode probabilities within range 0.1% - 100%), transistors $M_{7\text{-}8}$ and $M_{13\text{-}16}$ were implemented as high threshold voltage devices (HVT). Yet another source of systematic errors results from the second order effects in MOS transistors such as the Early effect and the variable slope factor κ which depends on the operating point of the transistor [3]. The level of errors caused by such effects was observed to be smaller for thick gate oxide devices (TGO) and was further reduced by proper scaling of M_1-M_4; however, this could be a technology-dependent issue.

The absolute computational error, defined as the difference between the normalised output current [I_{D1} I_{D2}] and its corresponding value calculated from (2), was obtained using scripts for combined Matlab-Hspice simulations. The obtained histogram of the absolute systematic error distribution of the

multiplier from Fig. 3 generated based on 5000 random pairs $[I_{X1}, I_{X2}]$ with I_0 in the range 50 nA - 1 μA is shown in Fig. 4.

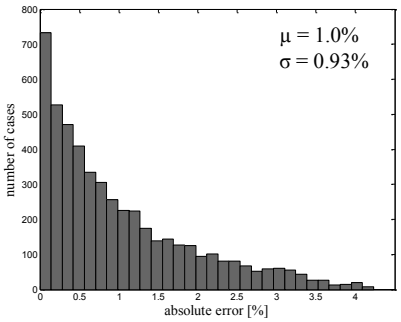


Fig. 4. Histogram of the absolute systematic error of the circuit from Fig. 3.

For more complex operations the equivalent hardware structures can be built using the circuit from Fig. 3. In order to reduce the complexity, the input signals should be distributed using their voltage representations rather than currents. The structures of two such systems for dot product and vector-matrix multiplications are presented in Fig. 5.

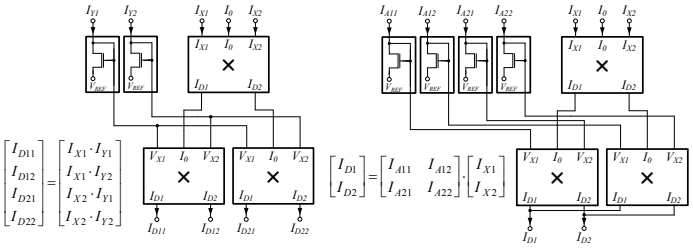


Fig. 5. Block diagrams of the arithmetic circuits for a) dot product and b) matrix-vector multiplications.

IV. RESULTS

A. Accuracy

The accuracy of Bayesian inference implemented in analogue hardware using the multiplier cell form Fig. 3 was verified in the simulations of several synthetic networks of a regular structure shown in Fig. 6 and generated for different numbers of nodes from 9 (3×3) to 121 (11×11) and random CPTs with entries within the range 5% - 95%. The input test vector consisted of the prior probability of the middle top node *A* and the evidence for the middle bottom node *B* (Fig. 6). These nodes were chosen to enforce the message propagation in the entire network to verify the settling time, however in practice it may depend on the test conditions and CPTs, and cannot straightforwardly be determined from the network structure. The simulation results of 6 different networks consisting of 5 to 121 nodes implemented using 3-way factor and variable nodes are presented in Table 1. The reported number of transistors and the DC supply currents in the steady state account for the entire circuit with additional terminating blocks along the borders and banks of current sources for CPTs. The absolute computational errors were calculated by comparing the currents representing beliefs of all the nodes in the circuit network obtained from simulations with the results generated by the software implementation of the message passing algorithm. In the simulations 100 random input vectors were generated for each network.

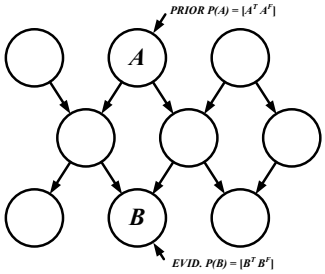
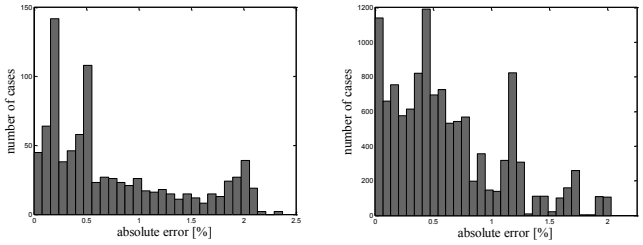


Fig. 6. The structure of the Bayesian network *TN*-3×3 consisting of 9 nodes generated on a regular grid (the structure of other test networks is similar).

It can be observed that the mean value and the spread of the absolute error remain at the same level despite the increased complexity of the computing hardware. This is mainly because the belief propagation requires only local computation and the respective arithmetic circuits perform normalisations of the intermediate results at each processing step which keeps the corresponding currents within the proper operating range. Histograms showing the distribution of the absolute systematic computational error for networks *TN*-3×3 and *TN*-11×11 are presented in Fig. 7. The convergence time observed in the simulations varied between 100 ns - 3 μs for the implemented networks. This means that the network *TN*-11x11 (requiring over 4800 multiplications for one iteration and at least 22 iterations in a software implementation to attain convergence) performs over 35 multiplications per nanosecond at 6.6 mW of power (including also additions, normalisations and the data traffic). This is equivalent to the computational performance of at least 35GOPS/6.6mW, i.e. > 5TOPS/W.

TABLE I. SIMULATION RESULTS OF THE TEST NETWORKS SHOWING THE COMPLEXITY, PERFORMANCE AND ACCURACY SCALING ISSUES.

Network	#MOS	I_{VDD}	mean(abs. error)	std(abs. error)
TN-1(Fig. 1)	3.290	232.2 μA	1.007 %	0.596 %
TN-3×3	5.918	431.7 μA	0.734 %	0.635 %
TN-5×5	16,406	1.166 mA	0.645 %	0.522 %
TN-7×7	32,126	2.256 mA	0.711 %	0.584 %
TN-9×9	53.078	3.701 mA	0.619 %	0.440 %
TN-11×11	79,262	5.500 mA	0.618 %	0.470 %



(a) (b)

Fig. 7. Histograms of the absolute systematic error: a) *TN*-3×3 b) *TN*-11×11.

The operation of the VLSI realisation of the test network from Fig. 1 and its software implementation using the message passing algorithm (MPA) were compared with the method using global marginalisation algorithm (GMA) for exact inference [2]. The statistical parameters of the absolute errors based on 500 random input vectors for nodes *A*, *D* and *E*, are presented in Table 2. It can be observed that the mean absolute error of the VLSI realisation is slightly higher when compared with the exact solution (GMA) but more importantly, the MPA also exhibits a considerable level of inaccuracy when applied to networks with loops. This indicates the existence of a certain lower bound of the attainable precision when using belief

propagation and it should be considered when attempting further circuit optimisation, which may not necessarily improve the overall computational accuracy.

TABLE II. THE COMPUTATIONAL ACCURACY OF THE VLSI AND MPA REALISATIONS IN REFERENCE TO THE EXACT INFERENCE METHOD (GMA)

Case	mean(abs. error)	std(abs. error)
VLSI - MPA	1.200 %	0.613 %
VLSI - GMA	1.250 %	0.762 %
MPA - GMA	0.548 %	0.660 %

The influence of fabrication mismatch on the absolute computational error was verified in simulations of the circuit implementation of the network in Fig. 1. The results showing the mean value of the absolute error with bars representing the standard deviations, obtained assuming averaging of the currents generated by a certain number of identical networks, are presented in Fig. 8. At each point, the networks were tested for 100 random sets of input vectors. The corresponding histograms of the absolute error assuming no averaging and for averaging over 100 network copies are shown in Fig. 9. It can be observed that the mean value of the error can, to some extent, be reduced by averaging over multiple hardware copies of the same network. The error reduces slowly, therefore a large number of structures may be required to obtain satisfying precision. In practical solutions, rather than building a set of fixed copies of the same network, it may be advantageous to consider a generic reconfigurable array of nodes where the same factor graph could be implemented in various ways, so that more precise results could be obtained reusing the same hardware only at the expense of processing time.

B. Scalability

The computational complexity and hence the size and the power consumption of the analogue hardware depend on the number of nodes but also on the complexity of each node determined by the number of its neighbours and the number of states of the processed variables. The implementations of several example Bayesian networks (*Mendel Genetics*, *Car Diagnostic*, *Alarm* and *Hail Finder*) [9] were analysed in terms of the complexity in the analogue hardware and the computational requirements in software (Table 3). The circuit from Fig. 3, in a standard 90 nm CMOS technology, occupies a 6.8 µm × 10 µm area (comparable with three D flip-flops). Using the number of multipliers required for a network realisation, the necessary silicon area can be estimated (the number of arithmetic multiplications is not equivalent to the number of analogue multiplier blocks, see Fig. 5).

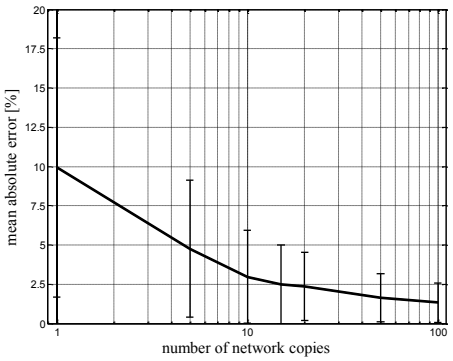
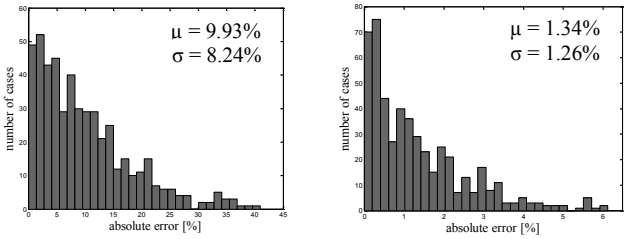


Fig. 8. The mean value of the absolute error of the network fom Fig. 1 accounting for the fabrication mismatch in terms of the number of network copies used for averaging.



(a) (b)

Fig. 9. Histograms of the absolute error assumig fabrication mismatch for: a) no averaging, and b) averaging over 100 results.

TABLE III. IMPLEMENTATION REQUIREMENTS OF THE EXAMPLE BAYESIAN NETWORKS WITH BINARY VARIABLES ONLY

Network		Analogue hardware			Operations/iteration		
Name	Comp.[a]	#MOS	I_{VDD}	Area [µm^2]	#mul	#add	#norm
Mendel	6/2/2	1,706	$140 \cdot I_0$	90×90	104	38	33
Car Diag.	18/5/3	14,244	$520 \cdot I_0$	260×260	1,656	572	104
Alarm	37/3/5	19,676	$1{,}182 \cdot I_0$	304×304	1,598	594	226
HailFinder	56/4/16	34,344	$2{,}034 \cdot I_0$	401×401	2,804	906	330

[a] Complexity: no. of nodes, max. no. of parents and max. no. of children.

V. CONCLUSIONS

In this paper the accuracy and scaling issues of the hardware implementations of factor graphs for Bayesian inference using belief propagation mechanism in analogue continuous-time circuits have been discussed. In particular, design guidelines towards improved computational accuracy of the arithmetic circuit for vector and matrix operations have been proposed. The reliability and scalability of such systems was verified in simulations using models from a standard 90 nm CMOS technology. It has been observed that efficient implementations of larger networks may be difficult due to the large area occupation and high computational inaccuracy. Nevertheless, the proposed idea of averaging and the use of reconfigurable systems provides a promising solution dedicated for smaller networks, with applications in low power systems requiring real-time Bayesian inference.

REFERENCES

[1] J. Pearl, “Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference”, Morgan Kaufmann Publishers, 1988.

[2] F. V. Jensen, T. D. Nielsen, “Bayesian Networks and Decision Graphs Second Edition”, Information Science & Statistics, Springer, 2007.

[3] C. Mead, "Analog VLSI and Neural Systems", Addison-Wesley, 1989.

[4] F. R. Kschischang, B. J. Frey, H. A. Loeliger, "Factor Graphs and the Sum-Product Algorithm", IEEE Transactions on Information Theory, Vol. 47, No. 2, pp. 498-519, Feb. 2001.

[5] H. A. Loeliger, F. Lustenberger F, M. Helfenstein, F. Tarkoy, "Probability Propagation an Decoding in Analogue VLSI", IEEE Transactions on Information Theory, Vol. 47, pp. 837-843, Feb. 2001.

[6] M. S. Zaveri, D. Hammerstrom, "CMOL/CMOS Implementations of Bayesian Polytree Inference: Digital and Mixed-Signal Architectures and Performance/Price" IEEE Transactions on Nanotechnology, vol. 9, no. 2, pp. 194-211, Mar. 2010.

[7] S. B. Luckenbill, "Building Bayesian Networks with Analog Subthreshold CMOS Circuits", Dept. of Electrical Engineering, Yale University.

[8] B. Gilbert, "A Precise Four-Quadrant Multiplier with Subnanosecond Response", IEEE Journal of Solid State Circuits, Vol. 3, No. 4, pp. 365-373, Dec. 1968.

[9] Resources of Norsys Software Corp. (networks library) available at: http://www.norsys.com/netlibrary/index.htm