Logic Gate and Circuit Training on Randomly Dispersed Carbon Nanotubes

A. KOTSIALOS[1*], M. K. MASSEY[1,2], F. QAISER[1], D. A. ZEZE[1,2], C. PEARSON[1,2], M. C. PETTY[1,2]

[1] *School of Engineering and Computing Sciences, Durham University, UK*
[2] *Centre for Molecular and Nanoscale Electronics, Durham University, UK*

Received ; in draft form June 1, 2014

This paper presents results of computations based on threshold logic performed by a thin solid film, following the general principle of evolution in materio. The electrical conductivity is used as the physical property manipulated for evolving Boolean functions. The material used consists of a composite of single-wall carbon nanotubes (SWCNTs) and the polymer poly(methyl methacrylate). The SWCNTs are randomly dispersed in the polymer forming a complex conductive network at the nano-scale. The training is formulated as an optimisation problem with continuous and binary constraints and is subsequently solved by two derivative-free algorithms, the Nelder-Mead (NM) and the Differential Evolution (DE) algorithms. This approach has been used to evolve gates and circuits. The NM fails to converge for all computational tasks, whereas the DM is always successful. The computation tasks considered are simple threshold logic gates and more complicated circuits. The thin film composite is very stable and its behavior remains the same after the optimal solution has been achieved.

Key words: Evolution in materio, randomly dispersed carbon nanotubes, threshold logic circuits, optimisation

[*] email: apostolos.kotsialos@durham.ac.uk

1 INTRODUCTION

Unconventional computing methods aim at exploring alternative media and methods for performing meaningful calculations. The way such a calculation is performed, i.e. the transformation of a set of inputs to a set of uniquely defined outputs, depends on the type of material used and the way its physical properties are exploited. Evolutionary inspired algorithms have found extensive acceptance and application in the design and implementation of unconventional computing systems. One important stream of research has resulted in the concept of evolvable hardware; an evolutionary algorithm is applied to a hardware platform where a piece of material is "evolved" to a state where it is able to perform a computation. In [35], a genetic algorithm was used for evolving a Field Programmable Gate Array (FPGA) to produce circuits that are able to calculate a specific Boolean function. Much research on controlling and evolving FPGAs and other types of evolvable hardware is reported in the literature, see for example [21], [18], [36], [27], [6]. In most cases, the computation task is focused on making an FPGA form a complex digital circuit following some specifications [23], [24]. Other devices such as the Field Programmable Transistor Array (FPTA) [32] and the Field Programmable Analog Arrays (FPAA) [17] have also been used following similar principles.

An observation from [34] from this line of work was that the evolution, i.e. the stochastic search algorithm employed, converged to solutions which were trying to exploit the physical properties of the material in the FPGA, rather than use all possible options and (re)wiring available with the same preference. This comes as no surprise, since the hardware itself is included in the optimisation loop when the objective (fitness) function is calculated for every candidate solution. It is only natural that, unless directed otherwise, the optimisation will try to exploit every possible means available, as reflected on the impact on the objective function's values, irrespective of the nature of the underlying causal relationship between this function and the decision variables. This observation leads to the Field Programmable Material Array (FPMA) [21], which allows what is termed *evolution in materio*, [7], [12], [22].

The FPMA is a "device" where physically-rich material is subjected to the application of voltage, and physical changes are induced. Incorporating such a device within an optimisation loop allows the use of evolutionary algorithms for bringing the material to such a state where a computational problem is solved [21]. Evolution in materio itself is the whole process of using artificial evolution to exploit directly physical properties of the material that are not normally associated with electronic components. In other words, the highly structured and highly adaptable FPGA environment is replaced by the material substrate [31]. Its physical properties are changed by use of selective stimuli so that the material is configured to perform a computation.

The type of computation induced this way is the result of intrinsic evolution, as oposed to extrinsic, [28], because it is performed on the material itself. Candidate solutions are evaluated by direct interaction with it. The evolution performed on an FPGA is allowed to operate on a search space intentionally equipped with the capability of rewiring and adjusting following circuit logic, even if the material of a specific FPGA affects the outcome, [34]. In evolution in materio, the search space is defined directly by the morphable physical property selected to altered and shaped by a set of configuration stimuli. Hence, the work presented here belongs to the field of evolvable hardware [28], where an evolutionary optimisation algorithm is used to generate configurations on the material that are able to perform a meaningful computation.

The output of the evolved system qualifies as a computation since the input/output behavior of the material is pre-specified. Once the optimal stimuli are applied on it, the measured outputs can be interpreted in a unique way according to this input/output relationship. However, there is no such interpretation of the internal state of the material system, since there are no practical physical theories that would allow the mapping of the internal state to abstract computation states, [1], [28]. Hence, the interpretation required by [3] and [14], does not exist for the considered system. The meaning to computation here is based on the *a priori* input/output relationship specification. This allows for the design of a software and hardware platform largely independent from the material itself.

Different types of material have the potential to be used within this concept. Liquid crystals have been used to evolve a tone discriminator [11], [7], a real-time robot controller [8], [9] and logic gates [10]. Randomly generated stochastic networked structures formed inside a piece of material have been used for developing memristor networks with neuron-like behavior. Polymeric 3D networks of memristors were simulated in [30]; the fabrication of statistical networks of polymer fibers forming memristor-like junctions is described in [5] and learning methods applied to them are discussed in [4].

Other types of materials that have been proposed include electroactive polymers, voltage-controlled colloids, irradiated silicon, Langmuir-Blodgett films, nanoparticle suspensions and microbal consortia; these can be used for solving different types of computatinal tasks, see [26] for an overview.

The material used in this paper is based on randomly dispersed Single-Wall Carbon Nanotubes (SWCNTs) forming a complex conductive network. Nanotubes are finding their way to applications in electronics since they possess electrical characteristics that compare favourably with existing materials [19]. For example, they have been used for logic gates and other electronic components [15], as well as for computers [29]. It is the SWCNT network's electrical conductivity, as a physical property, that is exploited and controlled here in order to bring the material to a state that allows the performance of some elementary computing tasks. Section 2 provides the details of the material used and its electrical properties.

By applying configuration voltages to change the material's state and subsequently applying a number of input voltages, the system's response is measured. Typically, the task of finding the configuration voltages and relevant quantities is given to an evolutionary algorithm that applies selectively a number of different inputs and probes the system's response until an objective function, modelling the computation efficiency, is minimised.

The optimisation problem developed for the needs of this study is tailored to fit the computing tasks' requirements, which is to calculate values of a Boolean function using thresholds. The calculation of `AND`, `OR` or `XOR` Boolean function values are a typical benchmark problem for evolution in materio. In addition to these, more complicated circuit structures are considered, such as the half- and the full-adder. Section 3 describes the principle of operation based on threshold logic.

The calculation consists of the material having a unique response to each of the finite possible inputs. The response comes in the form of voltage measurement(s) at selected locations on the material body. Measurements within specific upper and lower bounds are assigned a unique outcome, either 0 or 1 since Boolean functions are considered. The problem formulation for training the material for different circuits along with the equipment used is described in section 4.

The details of the computational tasks and corresponding results obtained are discussed in section 5. Section 6 concludes with an outline for future research.

2 MATERIAL DESCRIPTION

2.1 Thin Film Formation

The materials used in this work are based on composites containing SWCNTs and poly(methyl methacrylate) (PMMA). The concentration of SWCNTs (as a percentage of the PMMA) and the viscosity of the PMMA/SWCNT mixture are important properties that determine how the material behaves. The composites were created following methods widely reported in the literature. First, SWCNTs were dispersed in Anisole (VWR, analytical reagent grade) with the aid of an ultrasonic probe at a power of 20% (Cole-Parmer 750W ultrasonic homogenizer). The PMMA (Aldrich, M_w = 93,000) was then added and additional sonication was performed to create a visually uniform mixture. This material was deposited on suitable gold microelectrode arrays (section 2.2) by a simple drop casting technique. To promote quick drying and a more uniform coverage, the substrate was heated to 100°C prior to drop dispensing the material, and left for 30 minutes to drive off any remaining solvent.

2.2 Micro-electrode Arrays

Micro-electrode arrays were fabricated in gold, on slide glass substrates, using conventional etch-back lithographic techniques. The arrays were designed with small electrode separations (22 μm) so that high strength electric fields (5×10^{-5} V/m) could be applied even with the modest voltages (10.8 V). Figure 1(a) shows a detailed view of the area in contact with the material being tested. A Scanning Electron Microscope (SEM) image of the electrodes is shown in Figure 1(b). An optical micrograph of the SWCNT material deposited on the electrodes is given in Figure 1(c).

2.3 Material Electrical Characterisation

In-plane, two-terminal electrical measurements were undertaken on the SWCNT / PMMA composite to help understand how the material responds to electrical signals. Different concentrations of SWCNTs (expressed as a weight percentage of the PMMA solid) were investigated, since the number of interconnected SWCNTs plays an important role in defining the material's electrical properties.

Greater current carrying capacity is observed with a greater loading of SWCNTs. The linearity of the *IV* characteristic varies as the concentration of SWCNTs is increased; this is to be expected as at higher concentrations the probability of metallic pathways dominating is increased giving the more linear response. At all concentrations, the current increases monotonically with voltage. The SWCNTs in use are unsorted, i.e. the material contains a mixture of semiconducting and metallic varieties so, at higher concentrations, there are likely to be more metallic percolating pathways, yielding the significantly higher current. Two types of material were chosen for further study, with SWCNT concentrations of 0.23% and 0.53% by weight of the polymer respectively.

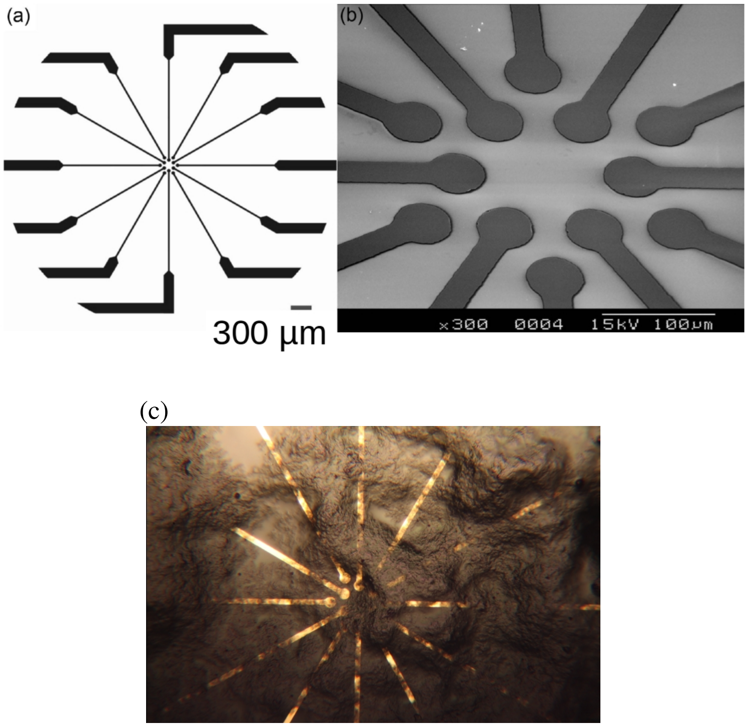


FIGURE 1
(a) Design of the microelectrode array (b) SEM image of electrode array (c) SWCNT/PMMA deposited on gold electrode arrays.

3 THRESHOLD LOGIC GATES USING SWCNT

Threshold Logic Gates (TLG), Figure 2, are devices used for computing Boolean functions [13]. Different implementations and designs exist using a variety of electronic components and materials, [2]. In its simplest form, a TLG has n binary inputs x_i, $i = 1, \ldots, n$ and a single binary output y; it is characterised by a set of weights w_i, $i = 1, \ldots, n$ (each associated with input x_i) and a threshold $\theta \in \mathbb{R}$. The binary output y is calculated by comparing the outcome of a weighted linear combination of the inputs to the threshold,

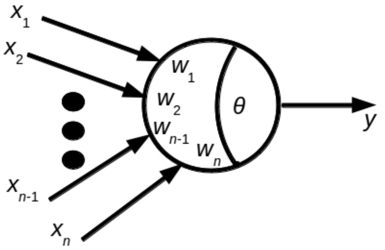


FIGURE 2
A Theshold Logic Gate (TLG) with n binary inputs x_i and weights w_i, $i = 1, \ldots, n$, a single output y and threshold θ.

i.e. according to

$$y = \begin{cases} 1 & \text{if} \quad \sum_{i=1}^{n} w_i x_i \geq \theta \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

The operation of a TLG can be generalised by a system with m binary outputs using $(L_j + 1) \in \mathbb{N}^*$, $j = 1, \ldots, m$ thresholds based on the following expression

$$y_j = Y_{j,p} \text{ if } \theta_{j,p} \leq \sum_{i=1}^{n} w_i x_i < \theta_{j,p+1} \tag{2}$$

where $Y_{j,p} \in \{0, 1\}$ and $\theta_{j,p} \in \mathbb{R}$, $p = 0, \ldots, L_j - 1$, are the known thresholds for each output.

The TLG concept concept provides the motivation for choosing logic gates and circuits for the initial computation tasks considered here. However, TLG do not provide a good explanatory model of the material's internal state, since the mapping between the device's internal state as given by eqn. (2) is lost in the material's response seen from the perspective of (3). Different types of computational tasks are going to be considered later, see [26].

Instead of designing hardware and circuits with specific electrical elements and network connections, the randomly dispersed network of carbon nanotube material, described in section 2, is used. In other words, instead of designing a circuit and explicitly characterising it with weights and thresholds, measurements at different locations on the material's body are used. The

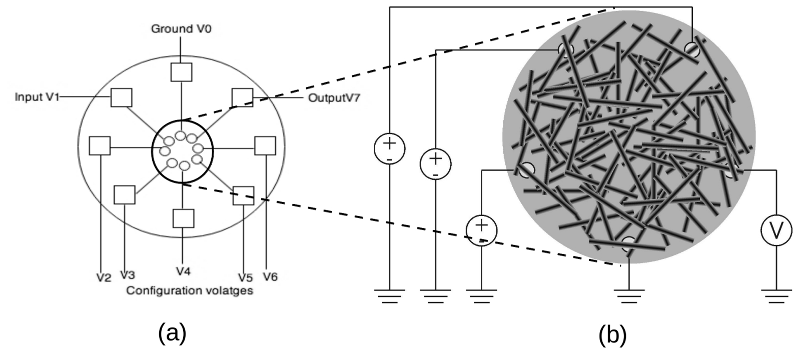


FIGURE 3
(a) Arrangement of the thin film of SWCNTs over electrode pads. (b) Randomly dispersed SWCNT network formed over the pads.

ranges of these measurements are partitioned in non-overlapping staggered bands that allow the implementation of eqn. (2). It is therefore straighforward to obtain the *a priori* specification of the desired input/output relationship that allows the material's response to be considered as a computation. This relationship is defined by the following expression.

$$y_j = Y_{j,p} \text{ if } \theta_{j,p} \leq M_j < \theta_{j,p+1}, j = 1, \ldots, m, p = 0, \ldots, L_j - 1 \qquad (3)$$

where $M_j \in \mathbb{R}$ is a measurement at location j (a location is a place where the electrodes enter into the SWCNT material). For the system considered here, voltage measurements are used. The principle is depicted in Figure 3 (a). One or more locations are selected to be charged with input voltages (e.g. V_1) and the voltage measurement obtained from another location (e.g. V_7) is the output. Configuration voltages are applied to other locations on the material in order to control its conductivity properties and effectuate the desired output for a given input.

From the physical point of view, the material works as a network of randomly dispersed nonlinear resistors, Figure 3 (b). Hence, the material itself does not undergo any mechanical change and as a result it is very stable; the results obtained are reproducible over time, as no significant degradation has been observed. It is its electrical conductance that is changed and morphed in order to obtain responses whose interpretation using the pre-selected thresh-

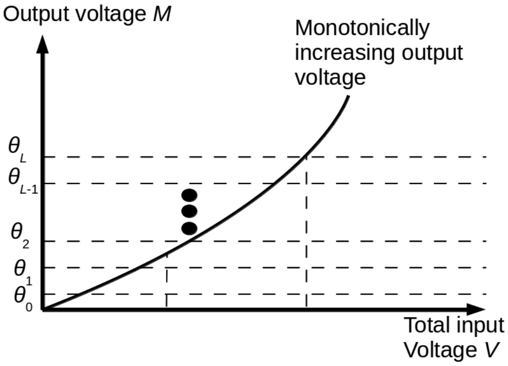


FIGURE 4
Division of output measurement range M into bands to be assigned to a possible input combination implementing eqn. (3).

old scheme complies with the desired input/output relationship.

Input voltages are applied to one or more of the input electrodes with an input logical 1 represented by a voltage V_{b_1} and a logical 0 by a voltage V_{b_0}. Configuration voltages V_q, $q = 1, \ldots, r$, are applied at r different locations on the material in order to affect the measured outputs M_j, $j = 1, \ldots, m$ at m different locations. For each of those m output measurements, the partitioning dictated by eqn. (3) is followed. The outcome of this operation must be in agreement with the TLG's or circuit's truth table $\mathbf{Y}(\mathbf{A}) \in \{0,1\}^m$, where $\mathbf{A} \in \{0,1\}^n$ is the circuit's binary input and $Y_j(\mathbf{A})$ is the binary output j. Hence, the computation performed by the material consists of mapping a particular input combination to the corresponding operating band of eqn. (3). This mapping takes the general form

$$H_j = F_j\left[M, \mathbf{Y}(\mathbf{A}), \theta_{j,0}, \ldots, \theta_{j,L_j}\right] \in \{0,1\} \tag{4}$$

where F_j maps measurement M_j to a band assigned according to (3) and as illustrated in Figure 4. A different mapping may be applied at the measurements collected at a different location on the material's body, i.e. $F_{j_1} \neq F_{j_2}$.

An important issue regarding the use of thresholds in the way described is the actual choice of their values. Pre-selected thresholds may be irrelevant to the specific material whereas trial–and–error could result in some suitable values, but this can be time consuming. Instead, in our approach, the thresholds are incorporated as functions of the configuration voltages. In other words, for a specific training problem, the thresholds used for measurements from output location j are calculated from

$$\theta_{j,p} = f_{j,p}\left(V_{b_0}, V_{b_1}, V_1, \ldots, V_r, \beta\right), j = 0, \ldots, m, p = 0, \ldots, L_j \tag{5}$$

where β a scaling parameter. Implicitly, it is the task of the optimisation algorithm to decide on selecting the configuration voltages for affecting all M_j concurrently and determining the thresholds, used in eqn. (3) for a fixed ordering of the θ_j's. For a given circuit, the thresholds of each particular output j are organised in the vector $\boldsymbol{\theta}_j$.

4 OPTIMISATION PROCEDURE

4.1 Problem Formulation

In this subsection, the optimisation problem that needs to be solved for making the material behave as a TLG or a more complicated circuit is formulated. The objective function and the constraints are expressed in terms of problem's decision variables and parameters.

The problem's parameters are the number of binary inputs n; the number of configuration voltages r; the number of binary outputs m of the TLG or circuit; the number of thresholds $L_j + 1$ used for measurements at output $j = 1, \ldots, m$; the lower and upper bounds of the voltages applied at the electrodes, $V_{\min}$ and $V_{\max}$, respectively; the truth table of the desired logical circuit $\mathbf{Y}(\mathbf{A}) \in \{0,1\}^m$, where $\mathbf{A} = [A_1 \ldots A_n]^T \in \{0,1\}^n$ is the binary input vector; the number of examples K used for training the material; and the upper bound of the scaling factor β, $B_{\max}$.

The decision variables are the voltage levels V_{b_1} and V_{b_0} which signify a logical 1 or a 0, respectively, at the input electrodes (typically $V_{b_0} = 0$ V); the configuration voltages V_q, $q = 1, \ldots, r$ used for affecting the measurements at the material's output locations; and the scaling factor $\beta \in [0, B_{\max}]$, without units, used for calculating threshold values. Hence, a candidate solution is a vector of the form

$$\mathbf{x} = \left[V_{b_0} V_{b_1} V_1 \ldots V_r \beta\right]^T . \tag{6}$$

Training examples are randomly pre-selected and given as inputs. A training example with index $k = 1, \ldots, K$ is a pair of the form $\left(\mathbf{A}^{(k)}, \mathbf{Y}\left(\mathbf{A}^{(k)}\right)\right)$, e.g. for the AND gate an instant of a training example with index k is $\left((1,1)^{(k)}, 1^{(k)}\right)$. Obviously $K > 2^n$ in order to get a sufficient number of training examples representing equally each possible input in the formed objective function.

The objective function selected is a quadratic expression of the error when a potential solution $\mathbf{x}$ is applied to the material. Let $M_j^{(k)}\left(\mathbf{x}, \mathbf{A}^{(k)}\right)$ denote the measured voltage at output electrode j when binary inputs $\mathbf{A}^{(k)}$ are applied and $H_j\left(\mathbf{x}, \mathbf{A}^{(k)}\right)$ the corresponding binary outcome when eqn. (4) is applied. Then, the least squares optimisation problem at hand is the following

$$\min_{\mathbf{x}} J = \sum_{k=1}^{K} \sum_{j=1}^{m} \left[H_j\left(\mathbf{x}, \mathbf{A}^{(k)}\right) - Y_j\left(\mathbf{A}^{(k)}\right)\right]^2 \tag{7}$$

subject to

$$\mathbf{b}_\ell \leq \mathbf{x} \leq \mathbf{b}_u \tag{8}$$

$$\theta_{j,p}\left(\mathbf{x}\right) = f_{j,p}\left(\mathbf{x}\right), j = 1, \ldots, m, p = 0, \ldots, L_j \tag{9}$$

$$H_j\left(\mathbf{x}, \mathbf{A}^{(k)}\right) = F_j\left[M_j\left(\mathbf{x}, \mathbf{A}^{(k)}\right), \boldsymbol{\theta}_j(\mathbf{x})\right] \in \{0, 1\} \qquad j = 1, \ldots, m \tag{10}$$

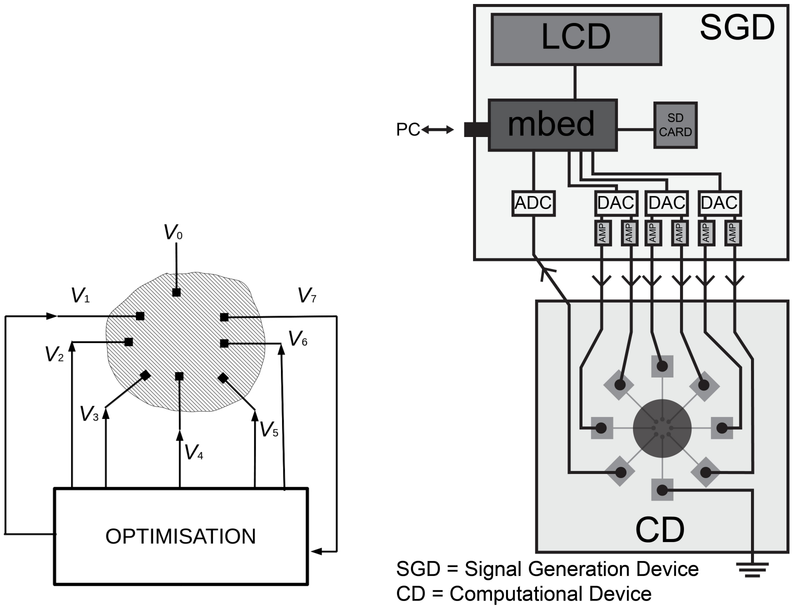
where $\mathbf{b}_\ell = \left[V_{\min}, \ldots, V_{\min}, B_{\min}\right]^T$ and $\mathbf{b}_u = \left[V_{\max}, \ldots, V_{\max}, B_{\max}\right]^T$. Notice there are no ordering constraints for the thresholds θ_j. The algorithms used converge to a solution where this ordering holds true.

Different optimisation methods can be used for solving problem (7)–(10). However, any implementation must have the hardware in the loop for calculating the values of the objective function for any candidate solution $\mathbf{x}$. Hence, the value of the objective function will always be corrupted with measurement noise because of (10). In addition, the binary nature of H does not allow the use of derivatives. Therefore, derivative-free population based algorithms are the first choice for solving problem (7)–(10).

Here, two algorithms are employed chosen for their simplicity of implementation. The well-known Nelder-Mead (NM) algorithm [25] with random periodic restart as suggested in [16] and the Differential Evolution (DE) algorithm suggested in [33] with parameters based on the discussion in [20].

4.2 Hardware Implementation

The general system concept using any optimisation search algorithm for training the material is shown in Figure 5(a). The case with eight electrodes connected to the material, shown as the hatched area, is illustrated. Voltage V_0 is the ground and all the other voltages are set and measured with respect to it. There is only one output measured, voltage V_7, and it is to this measurement



(a) (b)

FIGURE 5
Optimisation with hardware in the loop for material training. (a) System structure. (b) Hardware implementation.

that eqn. (10) is applied, i.e. $V_7 = M_1$. Assuming there are $n = 2$ inputs to the logic circuit, e.g V_1 and V_2, the four configuration voltages available are V_3–V_6. The values of V_1 and V_2 can only be V_{b_0} or V_{b_1}. V_{b_0}, V_{b_1} and V_3–V_6 can take any real value from the interval $[V_{\min}, V_{\max}]$ according to (8).

The optimisation search algorithm runs on a PC connected with the elecrtronics using a virtual serial connection over a USB port, as shown in Figure 5(b). At each iteration, the search algorithm needs to evaluate the fitness function (7) for a number of candidate solutions in the form of (6). This is done by setting the values of the configuration voltages V_3–V_6 and keeping them constant as sequence of K known but randomly selected binary pairs is sent to the two inputs V_1 and V_2. For each of these training input pairs, a number of measurements are taken at V_7 and their average is used in eqn. (10). The outcome of the application of (10) contributes the corresponding term in the total summation of the fitness function (7). The calcualtion of the total fitness function is performed on the board (the mbed microcontroller) and the value is sent back to the PC and the optimisation algorithm over the USB serial port.

The microcontroller (mbed) is based on the NXP LPC1768 system and is used to drive the various inputs to the material and monitor the outputs. The optimisation algorithm sends to the board real values for the elements of vectors $\mathbf{x}$ whose fitness is evaluated on the board, which are applied to material through digital to analogue converters with a sample size equal to 1 mVolt. The digital to analogue outputs and analogue to digital inputs are all buffered with suitable op-amps so as to isolate the electronics from the material under test. A schematic of the system is shown in Figure 5(b). This equipment allows DC voltage in the range of 0–11.2 V to be applied to the material on up to 10 outputs; as many as 4 inputs can be monitored if required.

5 RESULTS AND DISCUSSION

5.1 Training TLG

The general optimisation problem described in the previous section can be formulated for particular TLGs, i.e. make the material behave as a stand alone gate. The AND, OR and XOR gates are considered here. For these three gates and for the binary input pair $\mathbf{A}$ the output is given by (omitting the index of outputs j, since $j = 1$)

$$H(\mathbf{x}, \mathbf{A}) = \begin{cases} 1 & \text{if} \quad M(\mathbf{x}, \mathbf{A}) \geq \theta \\ 0 & \text{if} \quad M(\mathbf{x}, \mathbf{A}) < \theta. \end{cases} \tag{11}$$

This rule applies for the logic gates OR and AND, as shown in Figure 6 (a) and (b), respectively. Assuming the same configuration voltages and pair of V_{b_1} and V_{b_0}, the threshold of the OR gate is smaller than that of the AND gate, due to the monotonically increasing output voltage (the threshold value θ is not necessarily equal to the input logic 1 voltage V_{b_1}, although in some cases this can be imposed as a constraint). For AND and OR a single threshold is enough to distinguish the different binary input pairs. The threshold for AND should be high enough to cover the cases when only one of the inputs is zero; the OR's threshold should be lower, since only one input with value 1 is sufficient to achieve a binary output equal to 1. However, the single threshold scheme cannot always work, e.g. for the XOR gate or more complicated circuits.

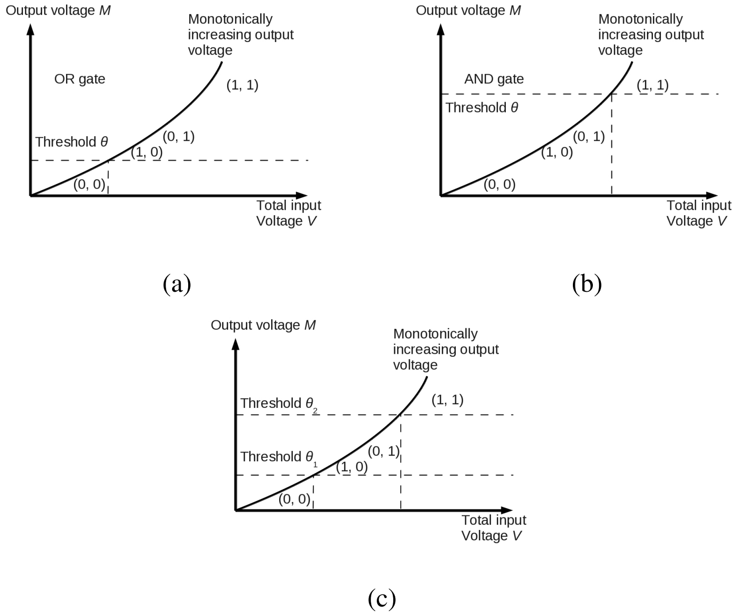


FIGURE 6
Threshold operation for (a) OR, (b) AND and (c) XOR gates.

In the case of XOR, the last entry of the truth table is problematic when only one threshold is used, since a low value is sought when the input is strongest, i.e. $\mathbf{A} = (1, 1)$. This requires the use of two, instead of one, thresholds as shown in Figure 6(c). Following the rationale of eqn. (3), an output voltage value within the range $[\theta_1, \theta_2)$ is assigned a 1, otherwise it is assigned a 0. In other words, the following rule is applied

$$H(\mathbf{x}, \mathbf{A}) = \begin{cases} 0 & \text{if} \quad M(\mathbf{x}, \mathbf{A}) \geq \theta_2 \\ 1 & \text{if} \quad \theta_1 \leq M(\mathbf{x}, \mathbf{A}) < \theta_2 \\ 0 & \text{if} \quad M(\mathbf{x}, \mathbf{A}) < \theta_1. \end{cases} \tag{12}$$

The AND and OR gate are easy to resolve, since it takes just one configuration voltage to achieve that even when

$$\theta = f(\mathbf{x}) = V_{b_1} \tag{13}$$

i.e. the same voltage is used to denote a logic 1 in the input and the output in

A	B	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

TABLE 1
Truth tables for gates AND, OR and XOR (SWCNT concentration 0.23%).

view of eqn. (11). For the XOR gate, the following thresholds are used.

$$\theta_1 = f_{1,1}(\mathbf{x}) = V_{b_1} \tag{14}$$

$$\theta_2 = f_{1,2}(\mathbf{x}) = V_1. \tag{15}$$

Table 2 provides the optimal solutions for AND, OR and XOR when $n = 1$ and $n = 2$ configuration voltages are used on the thin film with 0.23% SWCNT concentration. The number of function evaluations (f.e.) required by the NM algorithm to reach a solution is also given, but this is only indicative, since convergence is affected by the initial simplex selected. Generally, when $n = 2$, the desired behavior is obtained at the very early steps of the NM algorithm. The DE algorithm performs similarly but results to different solutions, i.e. there are local minima to the problem, as should be expected.

More complicated logic circuits, are discussed next.

5.2 Training Threshold Logic Circuits

Section 5.1 was concerned with the problem of identifying one or two configuration voltages (along with a scaling parameter in the XOR case) for making the material behave as an elementary TLG. This is done by measuring the output voltage at a single point on the material. A logic circuit, representing a Boolean function, with n binary inputs and m outputs is a more complicated task as it requires voltage measurements at m locations. A number of different circuits are considered, two two-input-two-output, one three-input-single-output and one three-input-two-output (full adder). Figure 7 depicts them and Table 3 provides their truth tables.

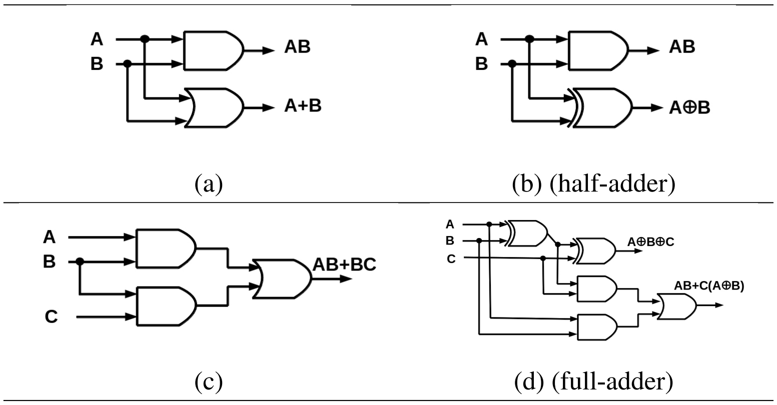
The principle of how the material system operates is illustrated in Figure 8 for the circuit shown in Figure 7(a). The two connections charged with voltages V_1 and V_2 are where the inputs A and B, respectively, are applied;

gate	n	**optimal solution**	**f.e.**
AND	1	$V_{b_1}^* = 0.291$	
		$V_1^* = 4.105$	1461
	2	$V_{b_1}^* = 1.172$	
		$V_1^* = 3.692$	
		$V_2^* = 3.692$	7
OR	1	$V_{b_1}^* = 0.731$	
		$V_1^* = 1.984$	1,599
	2	$V_{b_1}^* = 1.172$	
		$V_1^* = 3.692$	
		$V_2^* = 5.600$	7
XOR	1	$V_{b_1}^* = 2.445$	
		$V_1^* = 4.780$	110
		$\beta^* = 0.216432$	
	2	$V_{b_1}^* = 1.862$	
		$V_1^* = 1.372$	
		$V_2^* = 1.834$	121
		$\beta^* = 0.506318$	

TABLE 2
AND, OR and XOR stand alone gates optimal solutions and number of function evaluations needed (SWCNT concentration 0.23%).

Circuit (#)	**Inputs**			**Outputs**	
(a)	A	B	–	AB	$A+B$
	0	0	–	0	0
	0	1	–	0	1
	1	0	–	0	1
	1	1	–	1	1
(b)	A	B	–	carry	sum
(half-adder)				AB	$A \oplus B$
	0	0	–	0	0
	0	1	–	0	1
	1	0	–	0	1
	1	1	–	1	0
(c)	A	B	C	$AB+BC$	–
	0	0	0	0	–
	0	0	1	0	–
	0	1	0	0	–
	0	1	1	1	–
	1	0	0	0	–
	1	0	1	0	–
	1	1	0	1	–
	1	1	1	1	–
(d)	A	B	C	carry	sum
(full adder)				$AB+C(A \oplus B)$	$A \oplus B \oplus C$
	0	0	0	0	0
	0	0	1	0	1
	0	1	0	0	1
	0	1	1	1	0
	1	0	0	0	1
	1	0	1	1	0
	1	1	0	1	0
	1	1	1	1	1

TABLE 3
Logic circuits' truth tables.



(a) (b) (half-adder)

(c) (d) (full-adder)

FIGURE 7
Logic circuits.

hence V_1 and V_2 can only take values V_{b_0} or V_{b_1}. The two circuit outputs are measured at two distincive locations on the material and the voltage measurements $M_1 = V_7$ and $M_2 = V_6$ are used in eqn. (10). The configuration voltages are V_3–V_5 for the depicted system.

It should be noted that a test took place where the material was replaced by a random network of fixed resistors and the optimisation algorithms were applied to it, without success. In other words, it was not possible to obtain a circuit (half-adder) from resistors randomly wired on the board. The complex network, interconnectivity and nonlinear electrical behavior of the SWCNT thin film appear to facilitate the optimisation of TLG circuity.

Only two material compositions were studied as part of this work: one with a SWCNT concentration of 0.23% and the other 0.53%. The link between the material formulation and the effectiveness of the materials' ability to behave as TLG circuits remains inconclusive, although the results in this section focused on the formulation with the higher SWCNT concentration. Work is ongoing to understand and relate the materials' electrical/physical properties to the mechanisms involved with optimising TLG using the DE and NM algorithms.

5.3 Logic Circuit $(AB, A + B)$

The first circuit has two binary inputs, A and B, and two binary outputs, AB and $A + B$. The optimisation algorithm was able to locate a minimum at the

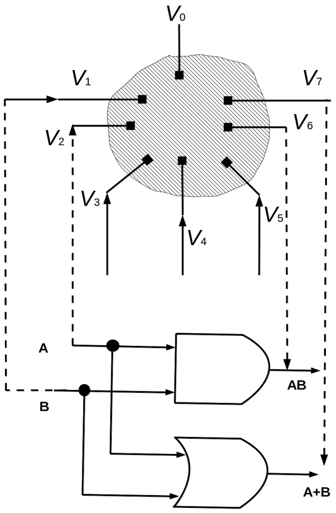


FIGURE 8
Material acting as logic circuit.

point shown in Table 4. This solution provides a voltage V_{b_1} used to denote 1 at the input and also use the same value as threshold at the output. In other words, for both outputs, the threshold θ is the same and it is $\theta = f(\mathbf{x}) = V^*_{b_1} = 3.279$ V. Figure 9 depicts the measured voltages at the two outputs. The x-axis represents the order over time of binary inputs (A, B) given to the two input locations. The y-axis shows the voltage levels measured at both output locations. On the left side the binary inputs corresponding to the `AND` gate are given and on the right side those of the `OR` gate. The solid line running through the diagram is the threshold $\theta = 3.279$ V. It can be seen that for both output locations the voltages measured tend to take values from specific bands and are different for each output.

Hence, for the $(0, 0)$ input which is first in the sequence of test binary inputs (x-axis), the measurement for the `AND` is 3.055 V and for the `OR` is 3.248 V. Since both are less than $\theta = 3.279$ V, the corresponding output in both is a 0. The next two inputs are the same pair $(0, 1)$ and the measured voltage at the `AND` is 3.138 V and at the `OR` is 3.338 V. For the `AND` gate this is below the threshold, whereas for the `OR` it is above, hence a $(0, 1)$

$V_{b_1}^* = 3.279$	$V_1^* = 11.200$	$V_2^* = 2.447$	$V_3^* = 3.200$
$V_4^* = 11.200$	$V_5^* = 5.054$	$V_6^* = 7.588$	$V_7^* = 9.556$

TABLE 4
Optimal solution for the half-adder circuit $(AB, A + B)$ in V, (SWCNT concentration 0.53%).

output is registered. Next is a $(0, 0)$ input and after that a $(1, 1)$, where the measurement at the AND is 3.318 V and at the OR is 3.442, both larger than θ, hence a $(1, 1)$ is registered.

Both outputs display a consistent behavior for the different test inputs and they map uniquely to the corresponding levels. For the AND gate, the output voltages for the $(0, 1)$ and $(1, 0)$ inputs are quite far away, whereas the for OR they are closer. This kind of spreading along with the threshold, are decided by the optimisation and the material's properties, which are the result of the specific conductive network formed by the carbon nanotubes.

5.4 Half-Adder

The half-adder circuit, Figure 7(b) is a more difficult case due to the requirement the XOR gate imposes on the output voltage when it is to be translated into a 0 for a binary input $(1, 1)$.

Table 5 gives the optimal solution when the threshold for the AND output is

$$\theta_{\text{AND}} = \theta_{1,1} = f_{1,1}(\mathbf{x}) = \beta V_1 \tag{16}$$

and the two thresholds for the XOR are

$$\begin{aligned} \theta_{\text{XOR},1} &= \theta_{2,1} = f_{2,1}(\mathbf{x}) = \beta V_2 && (17) \\ \theta_{\text{XOR},2} &= \theta_{2,2} = f_{2,2}(\mathbf{x}) = \beta V_3. && (18) \end{aligned}$$

The measured voltages at the two output locations can be seen in Figure 10. The AND terminal operates at much higher range than the XOR. There is a clear gap between the output voltages for the two locations. The threshold value of the AND gate θ^*_{AND} is placed at a level where there is clear distinction between the $(1, 1)$ input and the rest possible inputs. The $(1, 0)$ and $(0, 1)$ inputs are barely distinguishable in this case. On the contrary, in the XOR output, all the possible inputs are clearly distinguishable, but the thresholds are

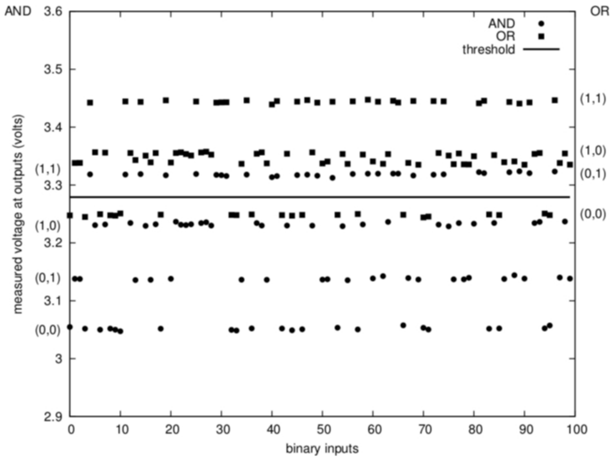


FIGURE 9
Output voltage measurements for arbitrary binary inputs for circuit $(AB, A + B)$ (SWCNT concentration 0.53%).

tighter. This is particularly true for the threshold distinguishing the $(1, 1)$ input, where the $\theta^*_{\mathrm{XOR},2}$ is placed at a distance of a few mV below the measured output.

5.5 Logic Circuit $AB + BC$

The logic circuit $AB + BC$ has three inputs and a single output, hence only one threshold is enough to calculate it. The threshold is calculated by $\theta = f(\mathbf{x}) = V_1\beta$. The maximum voltage was set to $V_{\max} = 11.2$ V and $B_{\max} = 4$ as well. Both the NM and the DE algorithms were able to idenify a solution. Table 6 provides the DE optimal solution and Figure 11 shows the corresponding measured output for an arbitrary string of 100 binary triplets. The difficulty in identifying an optimal solution lies to the fact that the binary triplets $(0, 1, 1)$ and $(1, 1, 0)$ result to a 1 at the output, whereas the triplet $(1, 0, 1)$ is mapped to a 0. Hence, the solution must be able to differentiate between input signals with similar strength at the input. This is done so, but only by a very small margin, separating $(1, 0, 1)$ from the other two. The measured output when $(1, 1, 0)$ is given as input is 3.650 V, for $(0, 1, 1)$ is 3.598 V and for $(1, 0, 1)$ is 3.571 V. It is only 1 mV that makes the difference for $(0, 1, 1)$ and allows it to map to 1 following the threshold rule with

$V_{b_1}^* = 3.965$	$V_1^* = 4.861$	$V_2^* = 2.794$	$V_3^* = 3.158$
$V_4^* = 2.733$	$V_5^* = 3.654$	$V_6^* = 5.116$	$V_7^* = 1.114$
$\beta^* = 0.621$	(no units)		
$\theta_{\mathtt{AND}}^* = 3.0187$	$\theta_{\mathtt{XOR},1}^* = 1.7354$	$\theta_{\mathtt{XOR},2}^* = 1.9614$	

TABLE 5
Optimal solution for circuit $(AB, A \oplus B)$ in V, material used is (SWCNT concentration 0.53%).

$V_1^* = 8.859$	$V_2^* = 3.693$	$V_3^* = 0.000$
$V_4^* = 0.172$	$V_5^* = 7.462$	$V_6^* = 7.577$
$V_{b_1}^* = 8.672$	$\beta^* = 0.406$	$\theta^* = 3.597$

TABLE 6
Optimal solution for circuit $AB + BC$ in V (except for β^*) (SWCNT concentration 0.53%).

$\theta^* = 3.597$ V. The same behavior is also exhibited in the solution of the NM algorithm.

5.6 Full-Adder

The full-adder circuit is a three-input two-output system. From its truth table shown in Table 3, it can be seen that the carry requires only one threshold that is able to differentiate between inputs that have at least two binary 1s in the input triplet. The carry threshold θ_c is calculated by

$$\theta_c = \theta_{1,1} = f_{1,1}\left(\mathbf{x}\right) = V_4. \tag{19}$$

However, the sum has a different behavior. When there is a single 1 in the input triplet, the result is a 1, but when there are two 1s the result is a 0 and when there are three 1s in the triplet, the result is again 1. Hence, there is a need for a three threshold rule for classifying the measured output voltage. This leads to the introduction of the thresholds $\theta_{s,1}$, $\theta_{s,2}$ and $\theta_{s,3}$ calculated

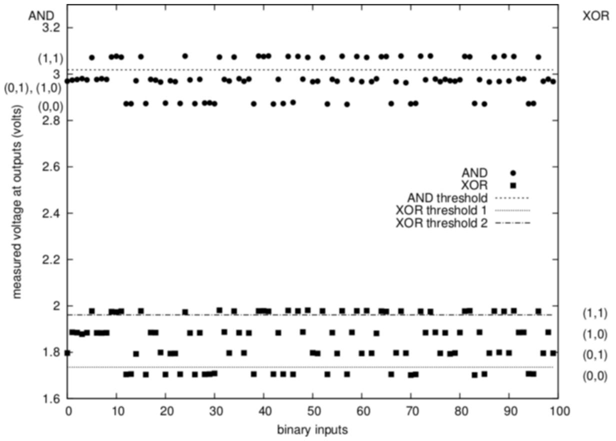


FIGURE 10
Output voltage measurements for random binary inputs for the half-adder circuit $(AB, A \oplus B)$ (SWCNT concentration 0.53%).

as follows.

$$\theta_{s,1} = \theta_{2,1} = f_{2,1}(\mathbf{x}) = V_1\beta \tag{20}$$

$$\theta_{s,2} = \theta_{2,2} = f_{2,2}(\mathbf{x}) = V_2 \tag{21}$$

$$\theta_{s,3} = \theta_{2,3} = f_{2,3}(\mathbf{x}) = V_3. \tag{22}$$

The threshold function mapping the measured voltages to the binary outputs when binary triplet $\mathbf{A} = (x_1, x_2, x_3)$ is given as input for the carry is

$$H_1(\mathbf{A}) = \begin{cases} 1 & \text{if} \quad M_1 \geq \theta_c \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

and for the sum is

$$H_2(\mathbf{A}) = \begin{cases} 1 & \text{if} \quad M_2 \geq \theta_{s,1} \\ 0 & \text{if} \quad \theta_{s,2} \leq M_2 < \theta_{s,1} \\ 1 & \text{if} \quad \theta_{s,3} \leq M_2 < \theta_{s,2} \\ 0 & \text{if} \quad M_2 < \theta_{s,3}. \end{cases} \tag{24}$$

The NM algorithm fails to find a solution in this case. However, the DE algorithm found the solution given in Table 7. Figure 12 shows that the thresholds are well separated and provide clear cut areas for classifying the input

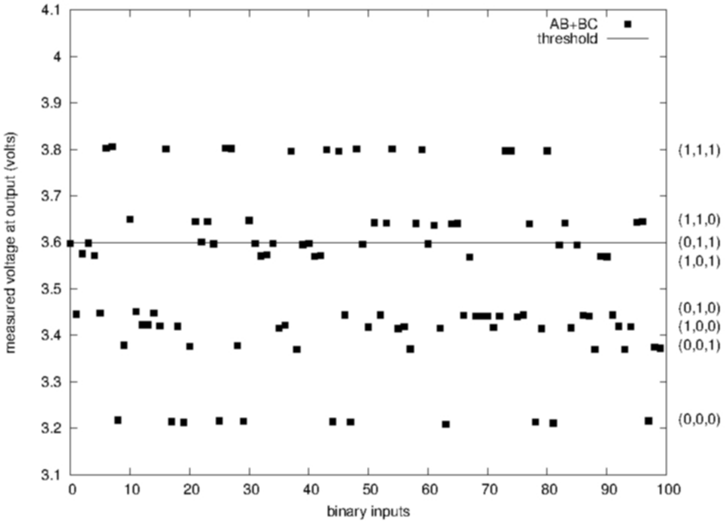


FIGURE 11
Output voltage measurements for random binary inputs for circuit $AB+BC$ (SWCNT concentration 0.53%).

triplet. It is interesting to note the variance of the measured voltage for the two outputs. The carry has significantly less variance, since only one threshold is necessary to differentiate between inputs with at least two 1s. The requirement of three thresholds for the sum results to a wider spread in order to differentiate the inputs with one 1 from two or three 1s.

6 CONCLUSION AND FUTURE WORK

This paper has presented results of computations based on threshold logic performed by a piece of material that consists of SWCNTs and poly(methyl methacrylate). Following the general principle of evolution in materio, the material's conductivity was used as the physical property manipulated for evolving Boolean functions, whose evaluation is based on threshold logic.

The material training problem is formulated as an optimisation problem with continuous and binary constraints. This formulation has been used to evolve gates and circuits on a piece of material placed on a board. By selectively applying and holding the optimal configuration voltages on certain

$V_1^* = 2.447$	$V_2^* = 3.710$	$V_3^* = 3.161$
$V_4^* = 2.968$	$V_5^* = 10.718$	$V_6^* = 5.060$
$V_{b_1}^* = 7.717$	$\beta^* = 1.7$	$\theta_c^* = 2.968$
$\theta_{s,1}^* = 4.160$	$\theta_{s,2}^* = 3.710$	$\theta_{s,3}^* = 3.161$

TABLE 7
Optimal solution for circuit $AB + BC$ in V (except for β^*) (SWCNT concentration 0.53%).

locations, a one-to-one mapping between set inputs and measured outputs is established. The material is very stable and this mapping is preserved over time, as no significant degradation was observed.

In this study, two different concentrations of SWCNT, 0.23% and 0.53% of the polymer's weight, were used. Both were able to be tuned to behave as TLGs, but the material with 0.53% was the only one that succeeded in behaving as a logic circuit for all cases considered. A possible factor contributing to this behavior may be the more uniform dispersion of nanotubes. A definite factor is the optimisation algorithm used for solving the training problem. From this perspective, the Differential Evolution is superior to the Nelder-Mead algorithm. The NM algorithm was not that efficient when called to address more complex Boolean functions than simple gates, whereas the DE has shown consistenly better results.

Currently, one line of our work is directed towards developing and customising optimisation algorithms for training different types of material to perform computational tasks.

Another line of investigation is directed towards the identification of suitable computational material and matching them with computational tasks. The SWCNT mix used here was suitable for the particular task of evolving Boolean circuits, but more complex computational task may require different properties of a material to be subjected to evolution.

Hence, there is a three dimensional space outlined by (a) computation material (b) computational task, and (c) optimisation method used for evolving the material. Future research aims at exploring this space and provide new paradigms of computation.

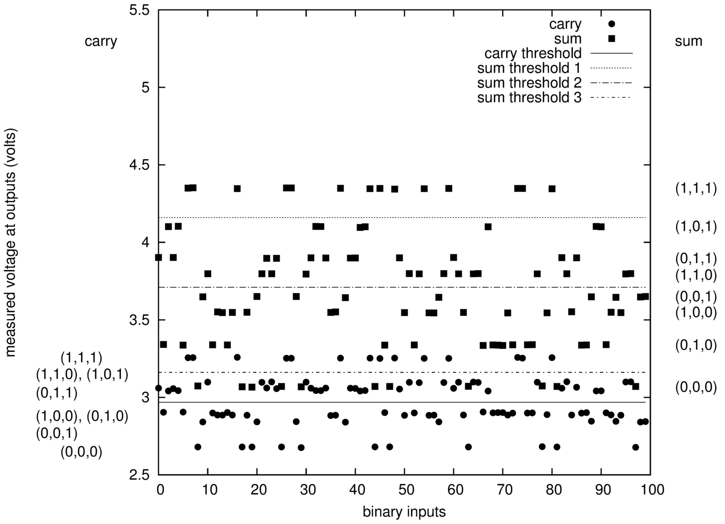


FIGURE 12
Output voltage measurements for random binary inputs for the full-adder (SWCNT concentration 0.53%).

7 ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 317662 (NAnoSCale Engineering for Novel Computation using Evolution – NASCENCE, http://www.nascence.eu).

REFERENCES

[1] Randy A Bartels, Margaret M Murnane, Henry C Kapteyn, Ivan Christov, and Herschel Rabitz. (2004). Learning from learning algorithms: Application to attosecond dynamics of high-harmonic generation. *Physical Review A*, 70(4):043404.

[2] V. Beiu, J.M. Quintana, and M.J. Avedillo. (2003). VLSI implementations of threshold logic-a comprehensive survey. *Neural Networks, IEEE Transactions on*, 14(5):1217–1243.

[3] B Jack Copeland. (1996). What is computation? *Synthese*, 108(3):335–359.

[4] Victor Erokhin. (2013). On the learning of stochastic networks of organic memristive devices. *International Journal of Unconventional Computing*, 9.

[5] Victor Erokhin, Tatiana Berzina, Svetlana Erokhina, and Marco P Fontana. (2009). Organic memristors and adaptive networks. In *Nano-Net*, pages 210–221. Springer.

[6] P.C. Haddow and A.M. Tyrrell. (2011). Challenges of evolvable hardware: past, present and the path to a promising future. *Genetic Programming and Evolvable Machines*, 12(3):183–215.

[7] S. Harding and J.F. Miller. (2004). Evolution in materio: A tone discriminator in liquid crystal. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 2, pages 1800–1807. IEEE.

[8] S. Harding and J.F. Miller. (2005). Evolution in materio: A real-time robot controller in liquid crystal. In *Evolvable Hardware, 2005. Proceedings. 2005 NASA/DoD Conference on*, pages 229–238. IEEE.

[9] S. Harding and J.F. Miller. (2005). Evolution in materio: investigating the stability of robot controllers evolved in liquid crystal. In *Evolvable Systems: From Biology to Hardware*, pages 155–164. Springer.

[10] S. Harding and J.F. Miller. (2007). Evolution in materio: Evolving logic gates in liquid crystal. *International Journal of Unconventional Computing*, 3(4).

[11] S.L. Harding and J.F. Miller. (2004). Evolution in materio: Initial experiments with liquid crystal. In *Evolvable Hardware, 2004. Proceedings. 2004 NASA/DoD Conference on*, pages 298–305. IEEE.

[12] S.L. Harding, J.F Miller, and E.A. Rietman. (2008). Evolution in materio: Exploiting the physics of materials for computation. *International Journal of Unconventional Computing*, 4(2).

[13] S.L. Hurst. (1971). *Threshold Logic: An Engineering Survey*. Mills and Boon.

[14] Colin G Johnson. (2004). What kinds of natural processes can be regarded as computations? In *Computation in Cells and Tissues*, pages 327–336. Springer.

[15] S.J. Kang, C. Kocabas, T. Ozel, M. Shim, N. Pimparkar, M. A Alam, S.V. Rotkin, and J.A. Rogers. (2007). High-performance electronics using dense, perfectly aligned arrays of single-walled carbon nanotubes. *Nature Nanotechnology*, 2(4):230–236.

[16] C. T. Kelley. (1999). *Iterative methods for optimization*, volume 18. Siam.

[17] D. Keymeulen, A. Stoica, R. Zebulum, S. Katkoori, P. Fernando, H. Sankaran, M. Mojarradi, and T. Daud. (2008). Self-reconfigurable mixed-signal integrated circuits architecture comprising a field programmable analog array and a general purpose genetic algorithm ip core. In *Evolvable Systems: From Biology to Hardware*, pages 225–236. Springer.

[18] J.D. Lohn and G.S. Hornby. (2006). Evolvable hardware: using evolutionary computation to design and optimize hardware systems. *Computational Intelligence Magazine, IEEE*, 1(1):19–27.

[19] P.L. McEuen, M.S. Fuhrer, and H. Park. (2002). Single-walled carbon nanotube electronics. *IEEE transactions on nanotechnology*, 1(1):78–85.

[20] Pedersen M.E.H. (2010). Good parameters for differential evolution. Technical Report HL1002, Hvass Laboratories.

[21] J.F. Miller and K. Downing. (2002). Evolution in materio: Looking beyond the silicon box. In *Evolvable Hardware, 2002. Proceedings. NASA/DoD Conference on*, pages 167–176. IEEE.

[22] J.F. Miller, S.L. Harding, and G. Tufte. (2014). Evolution-in-materio: evolving computation in materials. *Evolutionary Intelligence*, pages 1–19.

[23] J.F. Miller, D. Job, and V.K. Vassilev. (2000). Principles in the evolutionary design of digital circuits–Part I. *Genetic programming and evolvable machines*, 1(1-2):7–35.

[24] J.F. Miller, D. Job, and V.K. Vassilev. (2000). Principles in the evolutionary design of digital circuits–Part II. *Genetic programming and evolvable machines*, 1(3):259–288.

[25] J. A. Nelder and R. Mead. (1965). A simplex method for function minimization. *Computer journal*, 7(4):308–313.

[26] NASCENCE project (ICT 317662), (2013). Report on suitable computational tasks of various difficulties. Deliverable D4.2.

[27] K. Raman and A. Wagner. (2011). The evolvability of programmable hardware. *Journal of The Royal Society Interface*, 8(55):269–281.

[28] Lukáš Sekanina. (2009). Evolvable hardware: from applications to implications for the theory of computation. In *Unconventional Computation*, pages 24–36. Springer.

[29] M.M. Shulaker, G. Hills, N. Patil, H. Wei, H.-Y. Chen, H.-S.P. Wong, and S. Mitra. (2013). Carbon nanotube computer. *Nature*, 501(7468):526–530.

[30] R Sigala, A Smerieri, A Schüz, P Camorani, and V Erokhin. (2013). Modeling and simulating the adaptive electrical properties of stochastic polymeric 3d networks. *Modelling and Simulation in Materials Science and Engineering*, 21(7):075007.

[31] S. Stepney. (2008). The neglected pillar of material computation. *Physica D: Nonlinear Phenomena*, 237(9):1157–1164.

[32] A. Stoica, D. Keymeulen, R. Zebulum, A. Thakoor, T. Daud, Y. Klimeck, R. Tawel, and V. Duong. (2000). Evolution of analog circuits on field programmable transistor arrays. In *Evolvable Hardware, 2000. Proceedings. The Second NASA/DoD Workshop on*, pages 99–108. IEEE.

[33] R. Storn and K. Price. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359.

[34] A. Thompson. (1997). An evolved circuit, intrinsic in silicon, entwined with physics. In *Evolvable Systems: From Biology to Hardware*, pages 390–405. Springer.

[35] A. Thompson, I. Harvey, and P. Husbands. (1996). Unconstrained evolution and hard consequences. In E. Sanchez and M. Tomassini, editors, *Towards Evolvable Hardware, The Evolutionary Engineering Approach*, pages 136–165. Springer, New York, NY.

[36] X. Yao and T. Higuchi. (1999). Promises and challenges of evolvable hardware. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 29(1):87–97.