Evolution in Nanomaterio: the NASCENCE project

Hajo Broersma

Abstract This chapter describes some of the work that was carried out by members of the NASCENCE project, an FP7 project sponsored by the European Community. Apart from some historical notes and background material, the chapter explains how nanoscale material systems have been configured to perform computational tasks by finding appropriate configuration signals using artificial evolution. Most of this exposition is centred around the work that has been carried out at the MESA+ Institute for Nanotechnology at the University of Twente using disordered networks of nanoparticles. The interested reader will also find many pointers to references that contain more details on work that has been carried out by other members of the NASCENCE consortium on composite materials based on single-walled carbon nanotubes.

1 A Bit of History

In the next sections, the reader will find a description of parts of the research that was carried out within the framework of the FP7 project NASCENCE (Nanoscale Engineering for Novel Computation Using Evolution) [4], funded by the European Community and running from November 1, 2012 to October 31, 2015.

This work was motivated by earlier work of Julian Miller and his coworkers (See, e.g., [17, 18, 19, 34, 35]), and Julian was also involved as one of the very active partners within the project. In fact, he was the major inspirer of the ideas that lead to the birth of the NASCENCE project, and as a matter of fact, he also invented the acronym. So, in my role as the coordinator of the project it is with great pleasure that I contribute to this volume on the occasion of Julian's 60^{th} birthday!

Hajo Broersma
Faculty of Electrical Engineering, Mathematics and Computer Science, CTIT Institute for ICT Research, and MESA+ Institute for Nanotechnology, University of Twente, Enschede, The Netherlands. e-mail: h.j.broersma@utwente.nl

The title of this chapter is a slight (nano)variation on the term *evolution-in-materio* (EIM) that was coined by Miller and Downing in [34]. The birth of the NASCENCE project originates from an inspiring seminar talk entitled "Evolution in materio: Using evolution to get matter to compute" that Julian Miller gave at Durham University, UK, on March 2, 2010. Almost at the same time, a "Big Ideas" project proposal entitled "Computational Carbon" on this topic was composed by members of two Durham research groups, one in Electronics led by Mike Petty, and the other one in Algorithms and Complexity led by myself. The purpose of these "Big Ideas" was to stimulate collaboration within the School of Engineering and Computing Sciences, a merger between the School of Engineering and the Department of Computer Science. Although our "Big Ideas" project did not get funded, it formed the basis for the NASCENCE proposal. The latter was submitted to the European Community at the time I had already left Durham to return to my former employer, the University of Twente in The Netherlands. Apart from the group at Durham led by Mike Petty and the group led by me in Twente, the NASCENCE project also involved groups led by Julian Miller in York, by Gunnar Tufte in Trondheim, and by Jürgen Schmidhuber in Lugano.

2 A Bit of Background

The key idea behind the NASCENCE project was to use nanoscale materials as a black-box for computation, and to apply artificial evolution in the form of genetic algorithms to control the external configuration stimuli on the material in order to find configurations that would enable the material to perform certain computational tasks. More details on the essential concepts will follow in the next section, but they are basically the same as in the earlier works of Julian Miller and his coworkers mentioned above, except for the down-scaling to nanoscale materials.

The advancement of nanotechnology offers promising opportunities for alternatives to digital computation. These alternatives might be one of the solutions to cope with the possible problems that have been identified with respect to the further miniaturisation and increasing energy consumption of digital circuitry and transistors.

Current digital computers are based on Turing's abstract model of computation that has been around since the 1930s [52], together with a computer architecture described as an outline of a machine to perform this type of computation proposed by Von Neumann in 1945 [44]. The latter formed the foundation of modern stored programs computers. This type of computation is possible at the scales we see it today through the invention of the transistor, and the miniaturisation, integration and cheap production techniques enabled by the advancement of nanotechnology. Since the computations are based on the abstraction to strings of zeros and ones that are realised by low and high valued electrical signals, the common term for this type of computation is digital computation. Although there existed forms of analogue computation before the introduction of digital computation, the latter is usually also

referred to as classical or conventional computation. It reflects a top down approach: first designing an architecture and the functionality of a computational device on a drawing table, and then realising it by building it from purpose-built components.

The approach of EIM is more bottom up, as we will explain shortly, and this EIM is one of many different alternative approaches that are gathered under the umbrella term unconventional computing. Since one of the key ingredients of EIM is the use of artificial, computer-controlled evolution, it is also part of a wide research area known as bio-inspired computation. It is inspired by the many complex processes that take place in living systems, and that can be seen as computations. These processes have been enabled through a long adaptive, selective and uncontrolled procedure of Darwinian evolution. The key ingredients of this evolution, like crossover, mutation, natural selection and survival of the fittest have been mimicked in computer science and operations research, and led to an optimisation technique that we refer to as computer-controlled evolution. This concept of computer-controlled evolution is known under a variety of other terms, e.g., evolutionary algorithms [13], genetic algorithms [20], and genetic programming [24, 47]. Before addressing more details about the use of computer-controlled evolution in this particular setting, it is convenient to first give a brief overview of the NASCENCE project.

3 NASCENCE in a Nutshell

The conceptual ideas and ingredients of the NASCENCE project are illustrated in Figure 1. The key idea is to use a sample of nanomaterial on a micro-electrode array (the Material Disc in Figure 1) as a black-box for computational tasks. The way to manipulate the black-box is to change the state of the material, by using the micro-electrode array to interact with the matter through programmable external stimuli, usually consisting of analogue signals, like voltages or currents. These signals are supplied by and read out by a purpose-built interface (as shown in the middle of Figure 1). This interface is equipped with analogue-digital converters and connected to a digital computer (a laptop or PC) to enable computer-controlled evolution. The computer program that is executed on this digital device acts as a discovery engine (see Figure 1) for finding configurations of the external stimuli that cause the material to perform the target computational task. Once suitable configurations have been determined for certain specified tasks, the given sample of nanomaterial on the micro-electrode array should be capable to function as a stand-alone reconfigurable device for these tasks.

Apart from the experimental work, another aim of the NASCENCE project was to model, simulate and explore this way of unconventional computation, and analyse and explain the observed phenomena.

This set-up led to a natural breakdown of the project work within NASCENCE into the following nine Work Packages and their key performance indicators (KPIs):

1: Materials processing and evaluation;
KPI: useful materials and micro-electrode arrays;

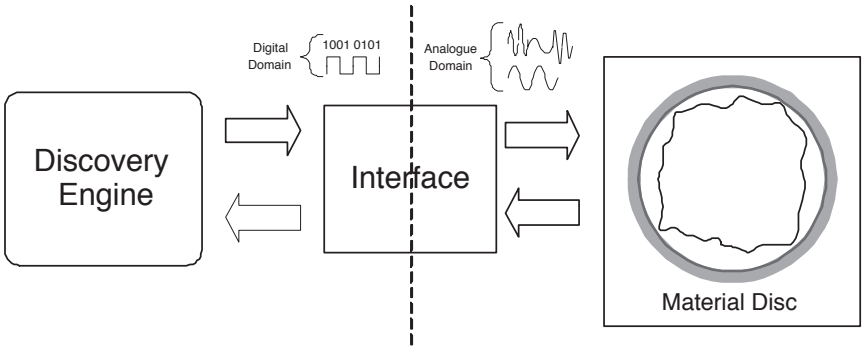

Fig. 1 Schematic conceptual overview of NASCENCE

2: Interface (hardware);
KPI: interaction PC/material; controlled stimuli;
3: Interface (software);
KPI: API; genetic algorithms;
4: Computational tasks (experimental);
KPI: suitable tasks/materials;
5: Computational tasks (simulations);
KPI: models for simulation/prediction;
6: Evaluation and data mining;
KPI: functional software for open ended experimentation;
7: Mathematical foundations;
KPI: theoretical foundation for evolvable systems;
8: Dissemination, collaboration and exploitation;
9: Project management;

In the next sections, the aim is to give a concise overview of the accomplishments of the NASCENCE project. More details can be found in [5] and the papers that are listed there. But we start with some additional remarks about the motivation behind the suggested approach of EIM, and by presenting more details on the use of evolutionary algorithms, with a specific example that was used to evolve a nanoparticle substrate.

4 Why and How to Evolve Dead Matter?

The work within NASCENCE has been inspired on one hand by the great achievements of Nature itself, and on the other hand by the opportunities offered by nanotechnology as a possible alternative to digital computing.

Evolution has done a great job for many living organisms, and it has done so through a conceptually rather easy but smart, albeit very slow bottom up process of

natural Darwinian evolution. There seems to be no design involved in this process: the computational instructions that underlie living creatures have not been designed but rather have (been) evolved. Two questions come to mind: Can we use something similar as evolution on 'dead' matter in order to get something useful? Can we do it quickly, via a rapidly converging evolutionary process?

The digital industry has done a great job for computation. It has developed and advanced digital circuitry extremely fast, to very dense packings of millions of transistors of very tiny size, currently in the range of 15 nm. It is not clear whether this will continue, and even the big players in this industry predict that we lose control over the production processes within the next decade. Another concern is that we are wasting a lot of energy on the production and use of digital equipment. Moreover, there are fundamental issues that have been raised with respect to the concepts behind digital computation. Due to the abstraction from electrical signal levels to logical zeros and ones, it seems plausible that digital computation is not making use of the full potential opportunities offered by physical systems. According to Conrad this leads us to pay "The Price of Programmability" [10]. Another question then emerges: Is there an alternative way to use (nanoscale) material that exploits the full potential properties of physical systems to solve computational problems rather than restricting them by abstraction to zeros and ones?

Motivated by the above observations, the NASCENCE project has been focusing on using nanoscale material in another, less controlled but hopefully more efficient way. The results that have been obtained so far by the members of the NASCENCE project look promising, as can be concluded from the later sections of this chapter. They show how computer-controlled evolution was used successfully to 'program' several functionalities in nanoscale materials by directly manipulating these physical systems. Although the results should be considered as a proof of principle, it is hoped that the approach taken in NASCENCE will inspire the creation of novel and useful devices in physical systems whose operational principles are not necessarily understood or are hitherto unknown.

The central idea of EIM is that the application of some physical 'configuration' signals to a material can cause it to alter how it affects 'input' signals and produces 'output' signals. The output signals are picked up and a fitness score is assigned, depending on how close the output signals are to the desired response. This fitness is assigned to the member of the population that supplied the configuration signals. For the purpose of using an evolutionary algorithm, the patterns of configuration signals that are applied as external stimuli have to be encoded, mimicking genomes in natural evolution. These artificial genomes enable us to mimic evolution on the codes, in order to find configurations that cause the matter to perform useful computational tasks. This approach is illustrated by the following example that is taken from [3].

4.1 An Illustrative Example: Nanoparticle Networks

Consider the following figure, in which the small spheres represent coated gold nanoparticles (NPs) that are trapped on an electrode array to form a fixed NP-network; more details can be found in [3].

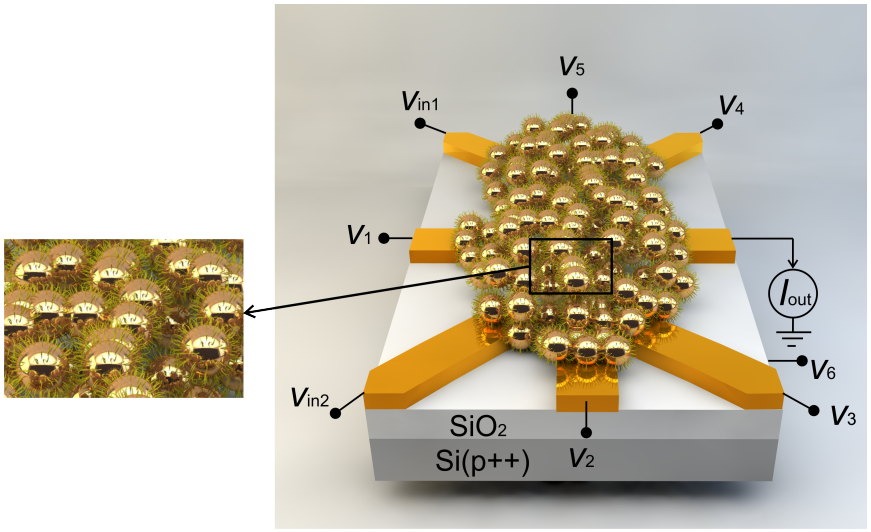

Fig. 2 Artist's impression of a disordered NP-network

Two electrodes are used for applying time-dependent voltage signals (as indicated by V_{in1}, V_{in2} in Figure 2). All but one of the remaining electrodes are used for applying static control voltages (as indicated by $V_1, \ldots, V_5$ in Figure 2), and one is used for measuring the resulting time-dependent current (as indicated by I_{out} in Figure 2). In addition, a static voltage can be applied to the back gate (as indicated by V_6 in Figure 2). The strongly non-linear switching behaviour of the NPs, that act as single-electron transistors (SETs) under the right circumstances, and their mutual interactions give rise to functionality greatly depending on the control voltages that manipulate the internal state of the NP-network.

At low temperatures, an NP with capacitance C has a charging energy $E = e^2/C$ which is larger than the thermal energy[1]. In this case NPs exhibit a phenomenon that is known as Coulomb blockade [22] and act as SETs. One electron at a time can tunnel when sufficient energy is available (ON state), either by applying a voltage across the SET or by electro-statically shifting its potential. Otherwise, the transport is blocked because of the Coulomb blockade (OFF state).

Measurements in [3] indeed indicated that the low-temperature electron transport in the NP-networks that were used for the experiments is dominated by this

[1] e is the charge on an electron

Coulomb blockade effect, and that their detailed behaviour strongly depends on the used input and output electrodes, as well as on the static voltages applied to the remaining electrodes. These characteristics lie at the basis of the ability to use these NP-networks for the realisation of computational tasks. Because the current NP-networks have a rather simple structure with not many configuration electrodes to play with, the aim in [3] was to first look for logic gates. The idea was thus to enable all possible 2-way logic gates with one and the same NP-network, by only varying the configuration voltages, without the necessity to apply a specific design of the NP-network.

So, in this set-up, fixing two of the electrodes as inputs and one as an output, the remaining electrodes together with the back gate have been used to configure the NP-network for the target functionalities, applying a genetic algorithm (GA). More details on the working of a GA, and how it was applied for our purposes, will appear in the next section, where the same GA was used for the experiments as well as the simulations. Some additional information on the simulations appears in later subsections, but next we present a short summary of the outcomes of the experimental work in [3].

4.1.1 Logic Gates

In this subsection we give a brief description of the work in [3] where the NP-networks as sketched in Figure 2 were used to evolve all Boolean logic gates.

In Figure 3(a) we see an atomic force micrograph (AFM) image of one of the real NP-networks that were used for the experiments in [3], where the two input electrodes and the output electrode are denoted by V_{IN1}, V_{IN2} and I_{OUT}, respectively. Time-dependent signals in the order of a hundred mV were applied to the input electrodes as illustrated in Figure 3(b), and a time-dependent current in the order of a hundred pA was read from the output electrode. The other five electrodes and the back gate have been used to apply different sets of static configuration voltages. Using a GA, suitable sets of configuration voltages have been found to produce the output functions of Figure 3(c,d). Red symbols are experimental data, solid black curves are expected output signals (matched to the amplitudes of the experimental data). We observe two clear negators (inverters) for the input functions P and Q in Figure 3(c), and we observe a variety of Boolean logic gates in Figure 3(d), including the universal NAND and NOR gate. Supplementary work in [3] reveals that all these gates show a great stability and reproducibility. For the exclusive gates (XOR, XNOR) spike-like features are observed at the rising and falling edges of the (1,1) input, as might have been expected for a finite slope in the input signals. More details can be found in [3].

The remarkable thing here is not that we can produce logic gates using the electrical and physical properties of charge transport in neighbouring NPs. What is remarkable, is that we can do this with one and the same sample of a disordered NP-network in a circular region of about 200nm in diameter, and by using only six configuration voltages. This shows the great potential for our approach. Note that a

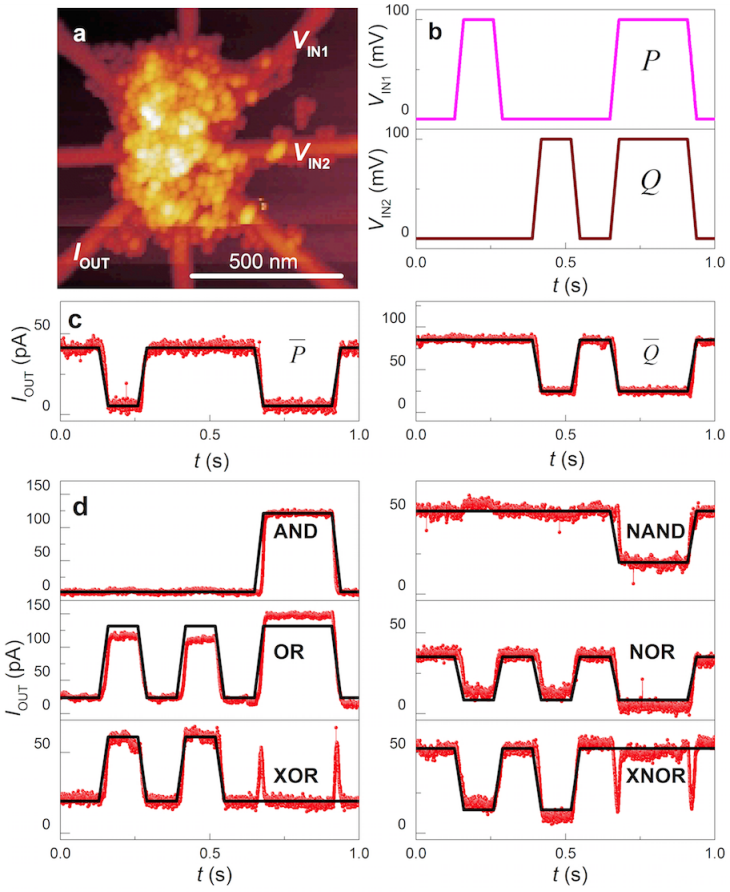

Fig. 3 AFM image of an NP-network (**a**), the input voltages in mV applied to V_{IN1} and V_{IN2} (**b**) and the different logic outputs in pA read from I_{OUT} (**c** and **d**) [3]

similar designed reconfigurable device based on today's transistor technology would require about the same space, and it would also require rewiring of the input signals to multiple inputs.

4.1.2 Behaviour of Gold NPs

The electrical properties and physical effect of Coulomb blockade behind the charge transport in the used gold NP-networks are pretty well understood. As described above, under suitable energy conditions and restrictions, the charge transport is governed by the Coulomb blockade effect [22, 53]. The particles act as SETs with a high ON/OFF ratio and strong non-linear behaviour. This makes them potentially good

candidates for interesting nontrivial functionalities. As an illustration, in Figure 4 we included some I-V characteristics of one of the NP-networks we used in [3].

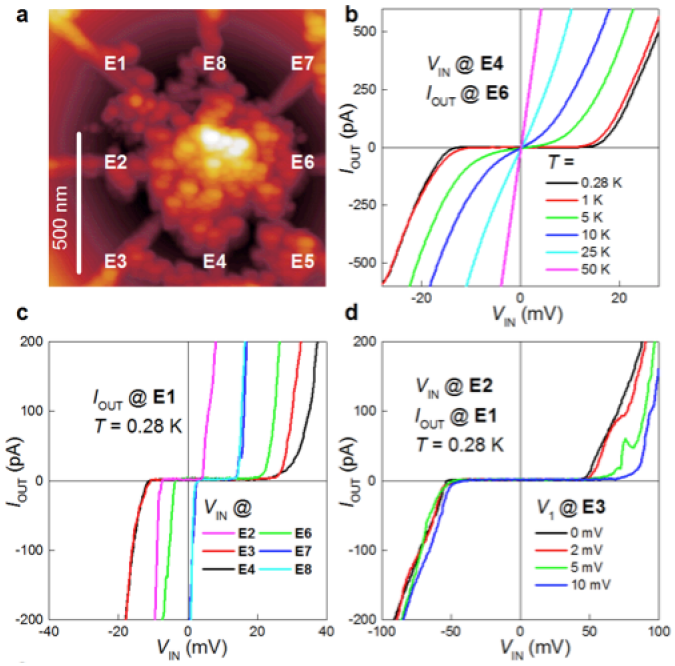

Fig. 4 AFM image of an NP-network (**a**), the input voltages in mV applied to electrode E_4 and the output current in pA read at electrode E_6 at different temperatures (**b**) the outputs in pA read from E_1 for different input electrodes at 0.28 Kelvin (**c**), and the effect of a static voltage applied at E_3 on the I-V curves of a fixed pair at 0.28 Kelvin (**d**) [3].

The nonlinear behaviour is very clear from the figures. We also observed a special form of nonlinearity usually referred to as negative differential resistance (NDR), as shown in Figure 5: a gate that was evolved to be a negator (inverter) for $0mV < V_{IN} < 100mV$ exhibits NDR within the considerably larger range $-50mV < V_{IN} < 100mV$.

This behaviour is interesting and we think it plays a key role in the potential evolvability of more complex functions. For instance, if we compare an XOR with an OR, then it can be observed that the OR could in principle be based on simple linear behaviour, where a high input signal gives rise to a high output signal, no matter whether both input signals are high or just one of the input signals. In case of an XOR this is different: we should only have a high output signal if precisely one of the input signals is high and the other is low; two high input signals should yield a low output signal. This is clearly more likely to be a reachable target functionality if the evolvable system exhibits NDR behaviour.

The above experimental results give supporting evidence for our expectation that much more is possible in terms of more complicated functionalities, enabled by

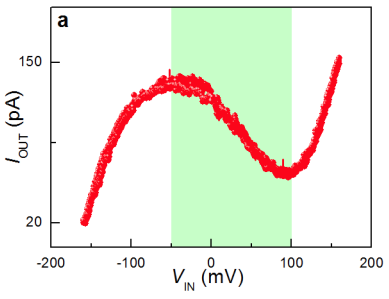

Fig. 5 NDR behaviour of a gold NP-network: the output current I_{OUT} increases with an increasing input voltage V_{IN} in the interval $-150mV < V_{IN} < -50mV$, but the I-V curve bends down in the region of $-50mV < V_{IN} < 100mV$ [3]

such disordered NP-networks with more (smaller) particles and a more sophisticated electrode array and back gate structure. This is part of our current research at the MESA+ Institute for Nanotechnology in Twente.

4.1.3 Simulations of NP-networks

Apart from the experimental work and results, the NASCENCE consortium has also worked on the theoretical underpinning of the experimental work and on simulations. The latter are based on physical or mathematical models of the material systems. In case of the NP-networks the physics is pretty well understood. In this subsection, a short description of the physical model that underlies the behaviour of jumping electrons in NP-networks will be presented, as well as an alternative approach to simulating these NP-networks using neural networks.

As we explained earlier, the charge transport in the NP-networks we have been using in the experiments that have led to [3] is based on a physical phenomenon that is known as the Coulomb blockade effect [22, 53]. The individual gold NPs act as SETs. Electrons can jump between neighbouring particles when the energy conditions are favourable. One electron at a time can tunnel between two particles if sufficient energy is available (ON state), either by applying a voltage across the particle or by electro-statically shifting its potential; otherwise, the transport is blocked due to Coulomb blockade (OFF state). These disordered assemblies of NPs therefore provide an almost random network of interconnected robust, non-linear, periodic switches, as a result of the Coulomb oscillations of the individual NPs. We have observed experimentally that electron transport below 5 Kelvin is dominated by Coulomb blockade, and strongly depends on the used input and output electrodes, as well as on the static voltages applied to the remaining electrodes.

Due to the high costs and time consuming experiments involved in the experimental work, it was highly desirable to develop a simulation tool to explore the potential functionalities of such NP-networks without the burden of spending many

hours in the lab and wasting expensive resources to look for such functionalities experimentally. In addition, the simulations can also inform us on the minimum requirements that are needed for obtaining the targeted functionality if we were able to produce these NP-networks according to a predetermined design. This could lead to new devices for the digital industry, possibly replacing purpose-built assemblies of transistors. Moreover, simulations can provide us with evidence concerning the scalability of our approach. Simulations can also give us new insights into the dynamics of the charge transport that might lead to a better understanding as to why and how the networks reveal the functionalities we observe. Furthermore, there are many questions on the use of these networks that are difficult to answer experimentally, because there are serious challenges in fabricating examples with smaller central gaps or with more control electrodes using the same area.

The simulation tool we developed in [12] is an extension of existing tools for simulating NP interactions, like SPICE [43] or SIMON [55]. Since the dynamics of our NP-networks is governed by stochastic processes (electrons on particles can tunnel through junctions with a certain probability), there are basically two simulation methods to our disposal: Monte-Carlo Methods and the Master Equation Method [53, 54]. Since the number of particles is large, this rules out the second approach, hence the Monte-Carlo Method is the only suitable candidate. This method simulates the tunneling times of electrons stochastically. To get meaningful results, one needs to run the algorithm in the order of a million times. Doing so, the stochastic process gives averaged values of the charges, currents, voltages, etc. More details on this physical-model based simulation tool can be found in [12].

We have validated our tool for designed systems with small numbers of particles that are experimentally known from literature, and that have also been simulated before [54]. We have also used our tool to examine other structures of NP-networks. Interestingly, we have shown through simulations that all Boolean logic gates that we evolved experimentally in [3] can be evolved in a regular 4x4 grid consisting of only 16 NPs. We refer to [12] for more details. Currently, we are not aware of any production techniques for constructing these regular grids of NPs.

Although our simulation tool can in principle handle arbitrary systems of any size, scalability is a serious issue if we consider the computation time. Even a parallellised CUDA code we have developed for a GPU does not really solve the problem if we want to simulate networks consisting of hundreds of particles. Moreover, as the networks in [3] cannot be produced according to a predefined specific design, it is not possible to use an accurate physical model for such systems.

With these drawbacks in mind, we have taken an alternative approach. This novel approach is based on training artificial neural networks in order to model and investigate the NP-networks.

Neural networks have proven to be powerful function approximators and have been successfully applied in a wide variety of domains [6, 25, 49, 51]. Being essentially black-boxes themselves, neural networks do not facilitate a better understanding of the underlying quantum-mechanical processes. For that purpose the physical model we described before is more appropriate. But in contrast to physical models,

neural networks provide differentiable models and thus offer interesting possibilities to explore the computational capabilities of the nano-material.

Before this exploration can take place, a neural network must first be trained, using data collected from the material. In our case, since we already have a physical model and an associated validated simulation tool for the NP-networks, to show that this approach is useful we can restrict ourselves in the first instance to training data obtained from the simulated material. This gives us the opportunity to predict functionalities in small NP-networks, also networks that have not been fabricated yet, like the 4x4 grid structure we mentioned above. This in turn can inform electrical engineers on the minimum requirements necessary for obtaining such functionalities without the burden of costly and time-consuming fabrication and experimentation.

One of the advantages of the neural network approach is that we do not need to have any detailed information on the structure or physical properties of the material. We only need as many input-output data combinations as we can get from the simulation tool or from measurements on a particular material sample, in order to train a neural network that models this specific sample. The more independent data we use, the more accurate the trained neural network is expected to model the sample.

Another advantage of the neural network approach is that one can optimise the input configuration through gradient descent instead of performing a black-box optimisation. In other words, as soon as we have trained the neural network with sufficiently many input-output combinations, searching for arbitrary functions is very fast and can happen independently of the material or the physical model.

To show that this approach is worthwhile, in [15] we used data obtained from the physical-model based simulations that we mentioned above to train a neural network. We show in [15] that the neural network can model the simulated nano-material quite accurately. The differentiable neural network model of the evolvable NP-network is then used to find logic gates, as a proof of principle.

This shows that the new approach has great potential for partly replacing costly and time-consuming experiments. We are currently also using the neural network approach on real data collected from samples of the NP-networks.

5 Back to the General Method

No matter what matter one uses, the approach of EIM requires a method to optimise or at least find good configurations for a specific target functionality of the evolvable system. But to be able to do this, first of all there is a need for a translation or mapping between the physical domain and the computer domain, and vice versa, as indicated in Figure 6 that was taken from [5].

In the physical domain, there is a material (in our earlier example an NP-network) to which physical stimuli can be applied (in our example voltages) or measured (in our example currents). In the NP-network example we have just given, the stimuli signals were static voltages and time-dependent voltages and currents, but the type of signals could be different, depending on the type of material and its characteristic

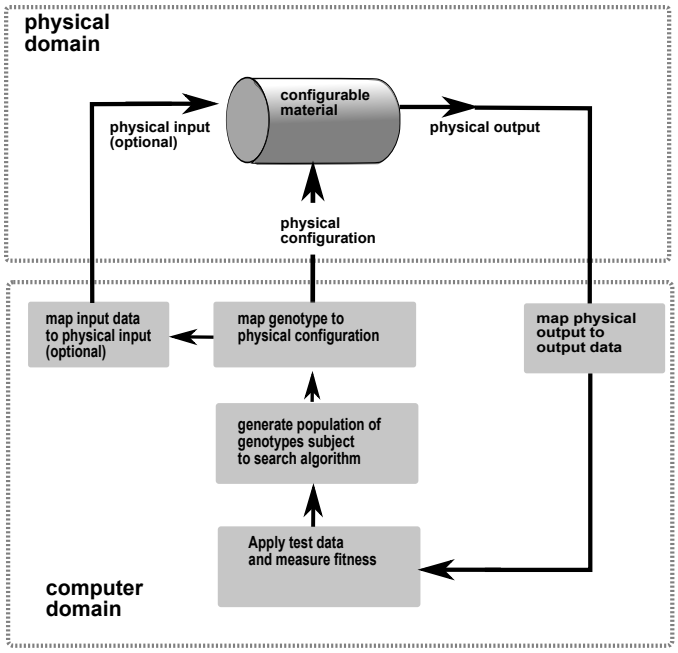

Fig. 6 Concept of EIM: physical and computer domain [5]

behaviour under influence of external stimuli. The signals are either input signals, output signals or configuration signals. With the help of a suitable interface made up of conventional electronics, in the computer domain a digital computer controls the application of physical inputs applied to the material (the time-dependent voltages in our example), the reading of physical signals from the material (the time-dependent currents in our example), and the application of other physical inputs to the material known as configurations (the static voltages in our example). Since we are using a digital computer to control the signals, we need to encode and decode between physical signals and representations of these signals by strings of numerical values (and at a lower level bits), with one entry for each of the applied (or measured) signal. Such strings can be interpreted as genotypes of the state of the physical material when supplied with the according levels of the applied signals. These genotypes of numerical data are stored and manipulated on the computer and, if necessary, again transformed back into configuration signals that physically affect the material, when applied. In order to find suitable signals that induce a target functionality between inputs and outputs of the evolvable system, the genotypes are subject to a GA that we are going to explain shortly.

Physical output signals, under the influence of certain input and configuration signals, are read from the material and converted to output data in the computer. Comparing this measured output to the ideal output of a target functionality under the given input signals, a fitness value is obtained from the measured output data.

This fitness value should be a good measure for how close the evolvable system is to performing the target functionality. This fitness of the measured performance and the corresponding genotype of the current state of the evolvable system is then used in the GA, with the goal to optimise (or at least determine sufficiently good settings of) the configuration signals for the target functionality.

To illustrate the steps in the GA, here are the ingredients of a generic GA in pseudocode, as adapted from [5].

Algorithm (Genetic Algorithm)

1: Generate an initial generation of size p. Set the number of generations as $g = 0$
2: **repeat**
3: Calculate the fitness of each member of the generation
4: Select a number of parents according to quality of fitness
5: Recombine some, if not all, parents to create offspring genomes
6: Mutate some parents and offspring
7: Form a new generation from mutated parents and offspring
8: Optional: promote a number of unaltered parents from step 4 to the new generation
9: Increment the number of generations $g \leftarrow g + 1$
10: **until** (g equals the number of generations required) **or** (the fitness is acceptable)

In Line 1 of the pseudocode, the initial generation of size p corresponds to p (possibly randomly chosen, but usually based on earlier experiments to characterise the typical behaviour of the system) initial genomes that are translated into configuration and input signals, and tried on the evolvable system.

Based on the measured outputs, for all the p choices, a fitness value is obtained in Line 3, and a selection of a number of genomes that correspond to the best fitness values is taken in Line 4.

To obtain new choices for genomes, the next generation of genomes to try on the material system is based on combining and manipulating the entries (genes) of the selected genomes, using principles that mimic natural evolution, like recombination, crossover, mutation, as in Lines 5-7. The "survival-of-the-fittest" principle of Darwinian evolution is implemented by using a form of fitness-based selection that is more likely to choose solutions for the next generation that are fitter rather than poorer. Mutation is an operation that changes a genome by making random alterations to some genes, with a certain probability. This is usually a suitable tool to avoid running into a bad local optimum. Recombination is a process of generating one or more new genomes by recombining genes from two or more genomes. Sometimes, genomes from one generation are promoted directly to the next generation; this is referred to as elitism (see the optional step in Line 8 of the algorithm).

It is clear that there are many choices involved in specifying a GA for a particular optimisation problem or for a particular evolvable system and target functionality, as in our research. These choices are usually based on trial and error, until a satisfactory convergence to acceptable solutions is observed.

Figure 7 illustrates the GA that was used for the NP-networks in [3]. Like every GA, this GA searches for optimum genomes by iterating over sufficiently many

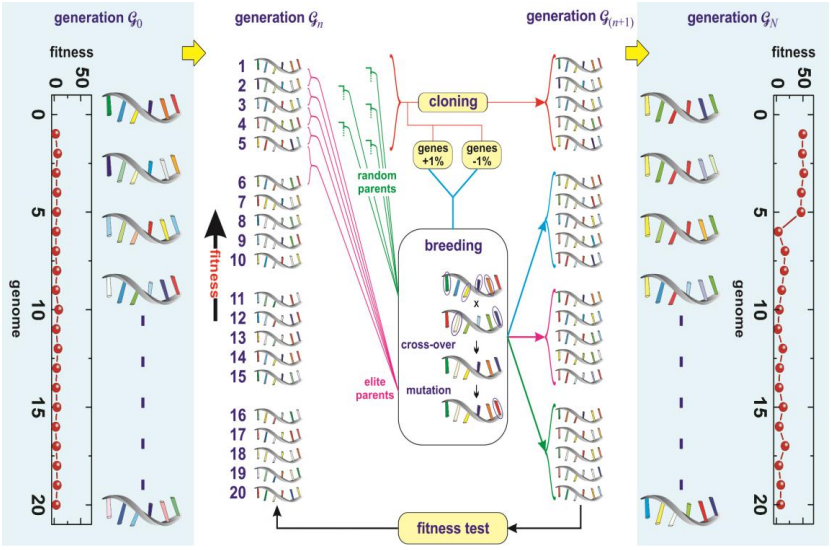

Fig. 7 The GA that was used for the NP-networks in [3]

generations. In this specific GA, every generation G_n has a fixed population of 20 genomes (represented by the strands in Figure 7) that are each composed of six genes (configuration voltages V_1 up to V_6, represented by coloured bars in the figure). The GA starts with a random initial generation G_0, whose fitness is generally low, as indicated in the leftmost column (for a GA search for an AND gate). With a composite cloning-breeding procedure (as illustrated in Figure 7 (**b**); see [3] for details) for evolving generation G_n into G_{n+1}, we eventually reach a final generation G_N whose top genomes have a (much) higher fitness than the top genomes of G_0 and the intermediate generations (as illustrated in the rightmost column of Figure 7 (**c**)). For more details on the settings of the GA that were chosen for the evolution of Boolean logic in these NP-networks we refer the reader to the supplementary material of [3].

6 Other Examples from the NASCENCE Project

Before we are going to present some of the results for other functionalities that have been experimentally tried out using the EIM approach within the NASCENCE project, we need to point out that there are basically two main ways to approach computational problems with EIM.

In the approach that we took in our earlier example, the GA determines configurations that in principle allow the material to act as a stand-alone reconfigurable logic device. This is a device which, provided with the appropriate evolved configuration signals, carries out the desired target computational mapping. In our earlier

example, one and the same NP-network was able to perform all 2-way Boolean logic gates, with different settings of the static configuration voltages, after removing the digital computer that was used to find suitable configuration settings. Within NASCENCE, a number of applications for which we took the same approach have been considered. We will shortly describe some of the other applications.

We note here that the term configuration of a material can have a number of meanings. It can merely be the application of physical signals to the material so that some underlying physical properties change, e.g. conductance or resistance. As a result, the material is put into a state that allows the desired computation to take place. This was the case in our earlier NP-network example. However, alternatively it may be that when the physical signals are applied to the material, the material physically changes in some way. For instance, the underlying (electrical) network might be rearranged, or the nanoscale elements could self-organise to a desired state so that the target computational functionality is obtained. An example of the latter is provided by earlier work with liquid crystal of Harding and Miller [16]. In this case, applied configuration signals caused liquid crystal molecules to twist, thus there was a physical change in the material when configuration signals were applied. As a matter of fact, both of the above effects may happen at the same time.

There is another approach in which EIM could be applied with the aid of evolvable systems. In this alternative approach, a material is merely used for the mapping of genomes to fitness values. The material is seen as an assistant in an evolutionary search process. It provides a "black-box" mapping from genome to output data (from which fitness is assessed). The thinking behind this is that the material may provide a more evolvable genotype-to-phenotype mapping, since physical variables can be exploited that could not be exploited if a purely algorithmic mapping was used (as is standard in evolutionary computation). In this type of hybrid system, much of the data required for solving a particular problem would remain on a digital computer. The role of the material would be to improve the search process itself. Thus in this case the material does not necessarily require any input data. Examples of computational problems that can be tackled using this approach are: Travelling Salesman Problem (TSP), Function Optimisation and Bin-packing. The TSP is the well-known problem of determining the shortest tour through a number of cities. Function optimisation is the problem of determining a vector of numbers which minimises a complex function. Bin-packing is the problem of packing a number of items into as few bins as possible, assuming that each bin has a fixed weight capacity. To obtain solutions to such problems using EIM requires that a set of configuration signals are determined that cause the material to output a suitable vector of measured values. The solutions are not necessarily optimal solutions, but usually approximate solutions. Generally, this method is difficult to scale and has at the time only resulted in solutions for rather small instances of the problems.

6.1 Materials and Interfaces used within NASCENCE

As we noted before, computational materials may be configured by different kinds of stimuli, e.g., electrical signals, magnetic fields, temperature variations, light, etc. However, it was decided at an early stage to only manipulate electrical signals within the NASCENCE project. Two types of evolvable material systems were constructed for this purpose. Both are based on electrode arrays. A material is deposited in the vicinity of the electrodes. Some of the electrodes are chosen as inputs (if the computational problem demands inputs), some are chosen as outputs, and a number of electrodes are chosen as configuration electrodes. In one system that we showed before, the material consisted of functionalised 20 nm AU NPs interconnected by insulating molecules (1-octanethiols) that were trapped in a circular region (200 nm in diameter) between radial metal (Ti/Au) electrodes on top of a highly doped Si/SiO_2 substrate, which functions as a back gate. This device operates at temperatures below 1°K [3]. In the second system, the material deposited was a mixture of single-walled carbon nanotubes (SWCNT) randomly mixed in an insulating material. An example of this other system is shown in Figure 8. The insulating material was either PMMA/PBMA (Polymethy/butyl methacralate) [42]. The material in the centre is a mixture of SWCNT and PMMA. The concentration of SWCNT is 0.05% by weight. SWCNTs are mixed with PMMA or PBMA and dissolved in anisole (methoxybenzene). 20 μL of material is drop dispensed onto the electrode array. This is dried at 100°C for 30 min to leave a film over the electrodes. Carbon nanotubes are conducting or semi-conducting and the role of the PMMA/PBMA is to introduce insulating regions within the nanotube network, to create non-linear current versus voltage characteristics.

Fig. 8 Circular twelve electrode array with a mixture of SWCNT and PMMA in the centre [42]

In order to be able to apply a GA to determine a set of signals that should be applied to the electrode arrays, one requires a hardware interface system between a computer and the material. The hardware system needs to allow a variety of signals to be applied to the electrodes. In the NASCENCE project the signals used were one of the following:

- Digital voltages
- Analogue voltages
- Square-wave signals

One also needs to be able to sample and record voltages and currents detected on electrodes, since from these measurements a fitness value is determined. Thus one needs equipment that allows the user to choose a sampling frequency and store the values (in a buffer). Since it is not known in advance to which electrodes input signals should be applied, generally one needs a way of allowing the GA to choose which electrodes will receive inputs (if the computational problem requires inputs) and which electrodes will be designated as outputs, and finally which electrodes will be the configuration inputs. A variety of different hardware systems have been explored for doing this.

- Digital acquisition cards together with programmable switch arrays [8]
- Mbed microcontrollers with digital to analogue converters [33]
- Purpose built platforms [31, 3]

6.2 Computational Problems

The NASCENCE consortium investigated a diverse range of computational problems. The list of problems with references to the associated papers is given below. We will refrain from giving details here, but refer the interested reader to the listed papers instead. Except for the results in [3] that were obtained using the NP-networks we showed before, all other results were obtained by using different versions of the mentioned SWCNT-composites.

1. Logic gates
   a. Two-input single output Boolean functions (e.g. (N)AND, (N)OR, XOR) [31, 23, 3]
   b. Three/Four input single output Boolean functions (e.g. even-3 and 4 parity) [37]
   c. Two-input two-output Boolean functions (e.g. half adder) [23, 3, 33]
   d. Three-input, two-output Boolean functions (e.g. full-adder)
2. Travelling Salesman

   This has no inputs and as many outputs as there are cities [8]
3. Classification
   a. Standard machine learning benchmarks (Iris, Lens, banknote): number of inputs equals the number of attributes, number of outputs is equal to the number of classes [38, 7]
   b. Frequency classification: this requires one input for carrying the source signal whose frequency is to be classified and two outputs which are used to decide the class of the frequency (high or low) [42, 39]
   c. Tone discriminator: this has the same number of inputs and outputs as the frequency classifier [42]

4. Function Optimisation
    a. This has no inputs and as many outputs as there are dimensions in the function to be optimised [41, 42]
5. Bin-Packing
    a. This has no inputs and as many outputs as there are items to be placed into bins [40]
6. Robot control
    a. This has as many inputs as robot sensors and as many outputs as robot actuators (e.g. motors) [36]
7. Graph colouring
    a. This has been looked at with a single input (graph select) and as many outputs as there are nodes to be coloured. Each output selects the colour of the node [30]

The seven classes of problems cover many types of problems involving markedly different numbers of inputs, outputs and number of instances. Some problems like TSP, Function Optimisation and Bin-packing have no inputs. The material acts like a form of genetic programming and via evolved configurations generates an approximate solution from its outputs. This is standard practice in genetic programming. In this type of approach a material is used in the genotype-phenotype mapping. However, one must be careful that in problems that have no inputs, evolution is not merely evolving configuration signals to produce outputs that are desired. In other words, that it is not directly wiring configuration signals to outputs, thus effectively ignoring the material.

In the work on evolving logic gates various functions present much greater difficulty due to their inherent non-linearity. It is well-known that parity functions are difficult to evolve non-linear functions. Indeed, they have been used as benchmark problems in genetic programming for some time.

6.3 Electrical Behaviour of SWCNT-composites

Whereas the physics behind the non-linear behaviour of the NP-networks that were used within NASCENCE is pretty much understood, the case is quite different for the material samples that are based on SWCNTs. Within NASCENCE several models at different levels of abstraction have been used for describing and analysing the electrical behaviour, and for predicting and simulating the computational capabilities of SWCNT-composites. As mentioned earlier, this SWCNT-based material may be configured by different kinds of stimuli, e.g., electrical signals, magnetic fields, temperature variations, light, etc., but within NASCENCE we have restricted these signals to three types:

- Static voltages;
- Square waves;
- A mixture of the two.

In order to be able to perform any kind of computation with the help of SWCNT-based composites, the input data and the configuration data must allow the exploitation and manipulation of underlying physical properties in the SWCNT-composite material. Moreover, such manipulated properties must be observable and give a measurable response, i.e., output response. As such, the choice of the input signal and configuration types play an important role and define which physical properties are available and utilised. In [30], a comparison between different signal types and their effect has been made for the evolution of solutions for small instances of the graph colouring problem.

In the case of static voltages, the parameters under evolutionary control are typically the electrode to which the signal is applied to, the starting time, the ending time, and the voltage amplitude. In the experiments in [30], the range of amplitude was limited from 0V to 3.3V. In the case of square wave signals, the evolved parameters are the electrode to which the signal is applied to, the starting time, the ending time, the frequency, and the duty cycle. In the experiments in [30], the square wave amplitude was fixed at 0V and 3.3V. The results there show that it was possible to evolve satisfactory solutions using all three types of signals (static voltages only, square waves only, and a mixture of static voltages and square waves). However, the choice of signal types influenced the evolutionary results. Square wave signals showed more promising potential than the other signal types, and the ability to produce richer dynamics, as one can also conclude from [45].

In [29], the model used to describe the SWCNT-networks suggests these networks behave as networks of resistors if only static DC voltages are applied. Moreover, it was shown there that TSP problems as introduced and solved in [8] could be solved using a SPICE model of the SWCNT-material as a 'cloud' of resistors [43]. In case of applied square wave signals, the material could be seen as an RC circuit, i.e., the SWCNT-material also holds capacitance. Inspection of evolved solutions in [46] reveals that the exploited physical properties are often unanticipated. In particular, it was observed that evolution was able to create and exploit signal delays, signal inversions and signal canceling. All the mentioned properties may provide a source of non-linearity and rich dynamics that may be potentially exploited for physical implementations of reservoir computing [21] [32] in SWCNT-networks.

We conclude this subsection with a short description of the four models that have been considered within NASCENCE to model and analyse possible behaviour of the SWCNT-composites.

6.3.1 Models of SWCNT-composites

Modelling of any material system consisting of nanoscale elements may be performed at several abstraction levels, ranging from the low level local interactions between neighbouring elements to the high level black-box behavioural models.

Other intermediate levels may also be important, particularly for describing emergence of properties at different intermediate scales. Four different models have been considered and analysed within NASCENCE.

1. *Models based on collective electrodynamics within the SWCNT-composites.*
   Such models are described in more detail in [26], and are based on Ashby's systems theory [1], as introduced in classical cybernetics. Electrical properties exploited for computation arise as an emergent property of the stimulated material. The transport of charge in the SWCNT-composite can be considered as a manifestation of a collective emergent property of electrons in the computing substrate. Within this framework, a future research direction is proposed in which one may consider the manipulation of quantum properties of electrons in the material so that the emerging electromagnetic properties can be used for computation. Another aspect of the proposed framework is to allow the manipulation of other parameters, e.g., temperature, in the description of the system state to create a bigger choice of variables. This approach is related to polymorphic electronics [50], where there may be different functionalities for different operating temperatures.
2. *Models based on DC or AC circuits.*
   Observing the behaviour of SWCNT materials under varying inputs, e.g., static voltages or square wave signals, allows macroscopic modelling of pin-to-pin characteristics with simple RC circuits. In [30], two SPICE models [43] are presented, one for describing the electrical behaviour when stimulated with static voltages applied to input pins, and one for capturing the behaviour when square waves are used as manipulation signals.
3. *Models based on dynamical hierarchies and cellular structure.*
   The approach in [27] aims at modelling conductivity dependence on the concentration of SWCNTs and varying electric potential in the material. The approach is based on two main paradigms: dynamical hierarchies [48] and cellular computation [9]. Each material sample is divided into a grid of cells, in which each cell represents a subarea of the sample with a specific content, i.e., polymer molecules, SWCNT bundles, or electrodes. Each cell behaves according to the physics of the elements it contains and the interactions with neighbouring cells. Results show that higher concentrations of SWCNTs lead to more percolation paths and consequently more current flow. Different cell shapes may be considered for future research.
4. *Models based on cellular automata.*
   The wide variety of problems solved with SWCNT-composites does not give any direct indication of the computational properties and computational power of the materials used. However, it is clear that the materials can be exploited at the computational level required to solve the given task. Cellular automata (CA) offer a broader knowledge of different complexity levels and computational classes, e.g., Wolfram classes [56]. As such, CA models of the material may enable the development of a framework that relates measurable physical properties to abstract CA behaviour. In [14], cellular automata transition tables of different complexities

have been evolved in-materio. An interesting future direction is the possibility to evolve universal cellular automata [11] [2] with the SWCNT-composites. In addition, ongoing work attempts to relate the evolved in-materio cellular automata with CA parameters [28], and relate the behaviour of the material to the notion of "edge of chaos".

7 Conclusions

This chapter tried to give the reader an overview of the accomplishments of the EC-funded NASCENCE project, with the relevant historical notes, background and the general concepts and principles. The author centred all material and examples around the work that was done within the framework of NASCENCE at his home university in Twente, notably the CTIT Institute for ICT Research and the MESA+ Institute for Nanotechnology. Details of this work and of the work done at the other partner's institutions can be found by following the pointers to the references that have been provided. The goal of the NASCENCE project was to demonstrate that computer-controlled evolution could exploit the physical properties of material systems based on carbon nanotubes and nanoparticles for solving computational problems. Experimental results have shown that this is indeed a plausible, competitive and efficient method for executing computational functions. Proof of concept has been given on several instances of problems within various complexities, and different number of inputs and outputs. The results are very promising and lay the foundation for further work. Future work includes the investigation of novel materials and larger and more sophisticated systems. One of our future goals is to apply these material systems for real world applications, and to develop stand-alone devices based on these principles. The long term goal of this research is to build information processing devices by exploiting bottom-up architectures without a predefined design and without reproducing individual components.

Acknowledgements The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 317662. It is with great pleasure that the author of this chapter thanks all the members of the NASCENCE project for the wonderful collaboration in this adventurous endeavour, and in particular Julian Miller for his inspiration and positive attitude. Happy birthday, Julian!

References

1. Ashby, W.R.: Design for a Brain, the origin of adaptive behaviour. Chapman & Hall Ltd. (1960)
2. Berlekamp, E.R., Conway, J.H., Guy, R.K.: Winning ways for your mathematical plays, volume 4. AMC **10**, 12 (2003)
3. Bose, S.K., Lawrence, C.P., Liu, Z., Makarenko, K.S., van Damme, R.M.J., Broersma, H.J., van der Wiel, W.G.: Evolution of a designless nanoparticle network into recon-

figurable boolean logic. Nature Nanotechnology **207**, 1048–1052 (2015). DOI 10.1038/NNANO.2015.207
4. Broersma, H., Gomez, F., Miller, J.F., Petty, M., Tufte, G.: Nascence project: Nanoscale engineering for novel computation using evolution. International Journal of Unconventional Computing **8**(4), 313–317 (2012)
5. Broersma, H..J., Miller, J.F., Nichele, S.: Computational matter: Evolving computational functions in nanoscale materials. In: A. Adamatzky (ed.) Advances in Unconventional Computing Volume 2: Prototypes, Models and Algorithms, pp. 397–428 (2016)
6. Ciresan, D.C., Meier, U., Masci, J., Schmidhuber, J.: A committee of neural networks for traffic sign classification. In: International Joint Conference on Neural Networks (IJCNN), pp. 1918–1921 (2011)
7. Clegg, K., Miller, J., Massey, M., Petty, M.: Practical issues for configuring carbon nanotube composite materials for computation. In: Evolvable Systems (ICES), 2014 IEEE International Conference on, pp. 61–68 (2014)
8. Clegg, K.D., Miller, J.F., Massey, M.K., Petty, M.C.: Travelling salesman problem solved 'in materio' by evolved carbon nanotube device. In: Parallel Problem Solving from Nature - PPSN XIII - 13th International Conference, Proceedings, *LNCS*, vol. 8672, pp. 692–701. Springer (2014)
9. Codd, E.F.: Cellular Automata. Academic Press (1968)
10. Conrad, M.: The price of programmability. In: R. Herken (ed.) The Universal Turing Machine A Half-Century Survey, pp. 285–307. Oxford University Press (1988)
11. Cook, M.: Universality in elementary cellular automata. Complex Systems **15**(1), 1–40 (2004)
12. van Damme, R., Broersma, H., Mikhal, J., Lawrence, C., van der Wiel, W.: A simulation tool for evolving functionalities in disordered nanoparticle networks. IEEE Congress on Evolutionary Computation (CEC 2016), 24-29 July 2016, Vancouver, Canada pp. 5238–5245 (2015)
13. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. SpringerVerlag (2003)
14. Farstad, S.: Evolving cellular automata in-materio. In: Master Thesis Semester Project, Norwegian University of Science and Technology, Supervisor: Stefano Nichele, Gunnar Tufte. NTNU (2015)
15. Greff, K., van Damme, R., Koutník, J., Broersma, H., Mikhal, J., Lawrence, C., van der Wiel, W., Schmidhuber, J.: Unconventional computing using evolution-in-nanomaterio: Neural networks meet nanoparticle networks. The Eighth International Conference on Future Computational Technologies and Applications, FUTURE COMPUTING (2016)
16. Harding, S., Miller, J.F.: Evolution in materio: A tone discriminator in liquid crystal. In: In Proceedings of the Congress on Evolutionary Computation 2004 (CEC'2004), vol. 2, pp. 1800–1807 (2004)
17. Harding, S., Miller, J.F.: Evolution in materio. In: R.A. Meyers (ed.) Encyclopedia of Complexity and Systems Science, pp. 3220–3233. Springer (2009)
18. Harding, S.L., Miller, J.F.: Evolution in materio: Evolving logic gates in liquid crystal. International Journal of Unconventional Computing **3**(4), 243–257 (2007)
19. Harding, S.L., Miller, J.F., Rietman, E.A.: Evolution in materio: Exploiting the physics of materials for computation. International Journal of Unconventional Computing **4**(2), 155–194 (2008)
20. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Press, Cambridge, MA, USA (1992)
21. Jaeger, H.: The echo state approach to analysing and training recurrent neural networks-with an erratum note. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report **148**, 34 (2001)
22. Korotkov, A.: Coulomb Blockade and Digital Single-Electron Devices, pp. 157–189. Blackwell, Oxford (1997)
23. Kotsialos, A., Massey, M.K., Qaiser, F., Zeze, D.A., Pearson, C., Petty, M.C.: Logic gate and circuit training on randomly dispersed carbon nanotubes. International Journal of Unconventional Computing **10**, 473–497 (2014)

24. Koza, J.: Genetic Programming: On the Programming of Computers by Natural Selection. MIT Press, Cambridge, Massachusetts, USA (1992)
25. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NIPS 2012), p. 4 (2012)
26. Laketić, D., Tufte, G., Lykkebø, O.R., Nichele, S.: An explanation of computation - collective electrodynamics in blobs of carbon nanotubes. In: Proceedings of 9th EAI International Conference on Bio-inspired Information and Communications Technologies (BIONETICS), in press. ACM (2015)
27. Laketić, D., Tufte, G., Nichele, S., , Lykkebø, O.R.: Bringing Colours to the Black Box - A Novel Approach to Explaining Materials for Evolution-in-Materio. In: Proceedings of 7th International Conference on Future Comp. Tech. and Applications. XPS Press (2015)
28. Langton, C.G.: Computation at the edge of chaos: phase transitions and emergent computation. Physica D: Nonlinear Phenomena **42**(1), 12–37 (1990)
29. Lykkebø, O., Nichele, S., Tufte, G.: An investigation of square waves for evolution in carbon nanotubes material. In: Proceedings of the 13th European Conference on Artificial Life (ECAL2015), pp. 503–510. MIT Press (2015)
30. Lykkebø, O., Tufte, G.: Comparison and evaluation of signal representations for a carbon nanotube computational device. In: Evolvable Systems (ICES), 2014 IEEE International Conference on, pp. 54–60 (2014)
31. Lykkebø, O.R., Harding, S., Tufte, G., Miller, J.F.: Mecobo: A hardware and software platform for in materio evolution. In: O.H. Ibarra, L. Kari, S. Kopecki (eds.) Unconventional Computation and Natural Computation, LNCS, pp. 267–279. Springer International Publishing (2014)
32. Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural computation **14**(11), 2531–2560 (2002)
33. Massey, M.K., Kotsialos, A., Qaiser, F., Zeze, D.A., Pearson, C., Volpati, D., Bowen, L., Petty, M.C.: Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites. Journal of Applied Physics **117**(13), 134903 (2015)
34. Miller, J.F., Downing, K.: Evolution in materio: Looking beyond the silicon box. Proceedings of NASA/DoD Evolvable Hardware Workshop pp. 167–176 (2002)
35. Miller, J.F., Harding, S.L., Tufte, G.: Evolution-in-materio: evolving computation in materials. Evolutionary Intelligence **7**, 49–67 (2014)
36. Mohid, M., Miller, J.: Evolving robot controllers using carbon nanotubes. In: Proceedings of the 13th European Conference on Artificial Life (ECAL2015), pp. 106–113. MIT Press (2015)
37. Mohid, M., Miller, J.: Solving even parity problems using carbon nanotubes. In: Computational Intelligence (UKCI), 2015 15th UK Workshop on, pp. xx–xx. IEEE Press (2015). In press
38. Mohid, M., Miller, J.: Evolving solution to computational problems using carbon nanotubes. International Journal of Unconventional Computing **xx**, xx–xx (2016). In press
39. Mohid, M., Miller, J., Harding, S., Tufte, G., Lykkebø, O., Massey, M., Petty, M.: Evolution-in-materio: A frequency classifier using materials. In: Proceedings of the 2014 IEEE International Conference on Evolvable Systems (ICES): From Biology to Hardware., pp. 46–53. IEEE Press (2014)
40. Mohid, M., Miller, J., Harding, S., Tufte, G., Lykkebø, O., Massey, M., Petty, M.: Evolution-in-materio: Solving bin packing problems using materials. In: Proceedings of the 2014 IEEE International Conference on Evolvable Systems (ICES): From Biology to Hardware., pp. 38–45. IEEE Press (2014)
41. Mohid, M., Miller, J., Harding, S., Tufte, G., Lykkebø, O., Massey, M., Petty, M.: Evolution-in-materio: Solving function optimization problems using materials. In: Computational Intelligence (UKCI), 2014 14th UK Workshop on, pp. 1–8. IEEE Press (2014)
42. Mohid, M., Miller, J., Harding, S., Tufte, G., Massey, M., Petty, M.: Evolution-in-materio: Solving computational problems using carbon nanotube-polymer composites. Soft Computing **xx**, xx–xx (2016). In press

43. Nagel, L., Pederson, D.: Simulation program with integrated circuit emphasis. Memorandum ERL-M382, University of California, Berkeley (1973)
44. Neumann, J.v.: First draft of a report on the edvac. Tech. rep., University of Pennsylvania (1945)
45. Nichele, S., Laketić, D., Lykkebø, O.R., , Tufte, G.: Is there chaos in blobs of carbon nanotubes used to perform computation? In: Proceedings of 7th International Conference on Future Comp. Tech. and Applications. XPS Press (2015)
46. Nichele, S., Lykkebø, O.R., Tufte, G.: An investigation of underlying physical properties exploited by evolution in nanotubes materials. In: Proceedings of 2015 IEEE International Conference on Evolvable Systems, IEEE Symposium Series on Computational Intelligence, in press. IEEE (2015)
47. Poli, R., Langdon, W.B., McPhee, N.F.: A Field Guide to Genetic Programming. Lulu Enterprises, UK Ltd (2008)
48. Rasmussen, S., Baas, N.A., Mayer, B., Nilsson, M., Olesen, M.W.: Ansatz for dynamical hierarchies. Artificial Life **7**(4), 329–353 (2001)
49. Sak, H., Senior, A.W., Beaufays, F.: Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. CoRR **abs/1402.1128** (2014). URL http://arxiv.org/abs/1402.1128
50. Sekanina, L.: Design methods for polymorphic digital circuits. In: Proc. of the 8th IEEE Design and Diagnostics of Electronic Circuits and Systems Workshop DDECS, pp. 145–150 (2005)
51. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (eds.) Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada, pp. 3104–3112 (2014). URL http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks
52. Turing, A.M.: On computable numbers, with an application to the entscheidungsproblem. Proceedings of the London Mathematical Society **42**(2), 230–265 (1936)
53. Wasshuber, C.: Computational Single-Electronics. Springer-Verlag (2001)
54. Wasshuber, C.: Single-Electronics – How it works. How it's used. How it's simulated. In: Proceedings of the International Symposium on Quality Electronic Design, pp. 502–507 (2012)
55. Wasshuber, C., Kosina, H., Selberherr, S.: A simulator for single-electron tunnel devices and circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **16**, 937–944 (1997)
56. Wolfram, S.: Universality and complexity in cellular automata. Physica D: Nonlinear Phenomena **10**(1), 1–35 (1984)