Rebooting Our Computing Models

Cadareanu, P.; Reddy C, N.; Almudever, C.G.; Khanna, A.; Raychowdhury, A.; Bertels, Koen; Narayanan, V.; Di Ventra, M; Gaillardon, P.-E.

DOI
10.23919/DATE.2019.8715167

Publication date
2019

Document Version
Final published version

Published in
2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)

Rebooting Our Computing Models

P. Cadareanu[1], N. Reddy C[2], C. G. Almudever[3], A. Khanna[4], A. Raychowdhury [5],
S. Datta[4], K. Bertels[3], V. Narayanan[2], M. Di Ventra[6], P.-E. Gaillardon[1]
(1) The University of Utah, USA, (2) Penn State, USA, (3) TU Delft, NL (4) University of Notre Dame, USA,
(5) Georgia Tech., USA, (6) University of California, San Diego, USA

***Abstract*—Innovative and new computing paradigms must be considered as we reach the limits of von Neumann computing caused by the growth in necessary data processing. This paper provides an introduction to three emerging computing models that have established themselves as likely post-CMOS and post-von Neumann solutions. The first of these ideas is quantum computing, for which we discuss the challenges and potential of quantum computer architectures. Next, a computational system using intrinsic oscillators is introduced and an example is provided which shows its superiority in comparison to a typical von Neumann computational system. Finally, digital memcomputing using self-organizing logic gates is explained and then discussed as a method for optimization problems and machine learning.**

I. Introduction

Computers have been evolving at an extremely rapid pace to keep up with the product benefits they support. These benefits include constant innovation in fields such as healthcare as well as personal devices as evidenced by the *Internet of Things* (IoT) phenomenon and the amount of data processing required for its maintenance. This evolution necessitates continual increase in device complexity as exemplified by Moore's law: the idea that every two years the number of *Complementary Metal-Oxide-Semiconductor* (CMOS) transistors in an *Integrated Circuit* (IC) should be doubled. But as we reach and surpass the 5 nm technology node, Moore's "law" becomes impossible to maintain through device scaling alone, and standard von Neumann computing architectures are struggling more than ever to sustain the increase of computing needs. This is because von Neumann architectures work on the assumptions that computation is fast and the amount of data being processed is not large, and the latter assumption has quickly become false [1]. The sense of urgency to power the evolving technological engine that is so integrated to our societal needs is persistent and requires a new approach.

Alternative design approaches such as the use of optimized accelerators or advanced power management techniques are successfully employed in contemporary designs, but these are not enough to keep up with the ever-increasing gap between on-chip and off-chip memory data rates. This trend, known as the von Neumann bottleneck, is the limitation on processor speed due to data transfer, and is the main restraint to advancing both system performance and energy scaling. The quest towards more energy-efficiency then requires solutions that disrupt the von Neumann paradigm, and this in turn necessitates a rethinking of the entirety of computer architecture. This overhaul includes considering both new software system solutions and new physics for developing computer hardware and is the basis for the IEEE Rebooting Computing Initiative *(https://rebootingcomputing.ieee.org)*.

Post-CMOS hardware solutions are quickly emerging from the elementary device units to their integration into nanosystems which consider system organization and architecture. These novel nanotechnologies bring new logic devices and new computational paradigms, requiring support from design tools and methodologies such as Electronic Design Automation (EDA). Device level innovations, including novel geometries and materials, introduce new logic devices [2; 3], such as carbon nanotubes [4], 2D materials (e.g., graphene [5], molybdenum disulfide (MoS_2) [6]), spintronics [7] or devices with improved functionalities with regards to traditional transistors [8; 9].

New computational paradigms are exemplified by a number of revolutionary ideas which aim to change the approach they take to computing. For instance, neuromorphic computing is an innovative computing paradigm inspired by the feedback system present in brain function [10; 11; 12; 13; 14]. The resulting neural networks are computationally substantially more efficient than von Neumann-style computing models. A branch of neuromorphic computing is chemical computation which models itself after biological cellular networks capable of learning and adapting in the same way non-synthetic biological systems do [15]. Other common applications using neuromorphic computation systems are the modeling of spiking neural networks [16; 17; 18] and spintronic-based neural network systems [19; 20], two applications which are also examples of *in-memory* computing. In-memory computation is enabled by both novel memory cells such as *Resistive Random Access Memory* (ReRAM) [21; 22; 23], *Magnetic Random Access Memory* (MRAM) [24], *Ferroelectric Field Effect Transistors* (FeFET) [25; 26], as well as standard memory such as *Static Random Access Memory* (SRAM) [27; 28], and this computation style effectively eliminates the von Neumann bottleneck. Similar to in-memory computing is memcomputing which uses memory to store *and* process information [29; 30]. Memcomputing machines have been shown to be efficient for some combinatorial optimization problems [29; 31; 32; 33]. Alternately, quantum computing [34] and adiabatic computation [35] are some of the better known emerging computing technologies which use quantum mechanical properties to resolve classical problems with increased computing power and decreased energy dissipation [36].

Novel computing models such as these allow for the potential to change the way information is processed and are the most substantial work-arounds of the von Neumann bottleneck. In this paper, we consider three alternatives to von Neumann computing chosen for their tight links with novel device technologies. In particular, quantum computing and its definition as a heterogenous architecture, as well as its challenges and potential applications are addressed. Also of interest is the idea of using coupled oscillators as an alternative to current CMOS-based devices. Memcomputing is the final approach discussed and refers to the idea of using memory

978-3-9819263-2-3/DATE19/©2019 EDAA

cells for processing so that there is no delay due to the transfer between the two, thus eliminating the von Neumann bottleneck completely.

The remainder of this paper is as follows: First, Section II considers the future of quantum computing. Section III addresses coupled oscillators and their use in a physical computational system. Section IV of this paper is a mathematical and application-driven explanation of memcomputing. Finally, Section V concludes this paper.

II. A Quantum Computer as an Accelerator

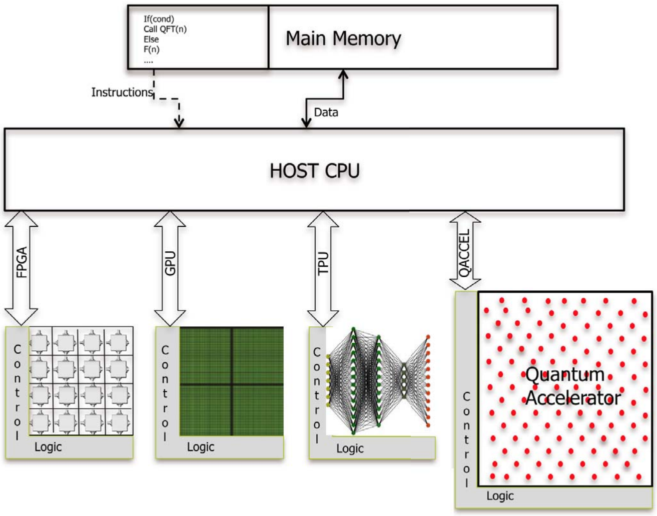
Designing a computer involves much more than the design of the processor chip. A fully-functioning computer as we know it involves using, storing, and computing data. Therefore, besides a processor, a computer requires memory to store instructions and data, interconnects so that this information can be transported to the processor and back again to the memory for storage after computation; ultimately, the results are stored more permanently on a hard disk. We need keyboards and screens to understand that the computer is operating correctly and producing the expected result. The semiconductor industry and subsequently the entire IT-industry has evolved from single-core processor architectures to homogeneous multi-core processors to finally the heterogeneous multi-core architectures widely used today, which include *Digital Signal Processing* (DSPs) and *Graphical Processing Units* (GPUs).

A. Quantum Computing as a Heterogeneous Architecture

The latest innovation in these heterogeneous architectures is the use of *Field-Programmable Gate Arrays* (FPGAs); their integration into conventional technologies took more than 15 years before becoming standard with big players like Intel and IBM. The shift to heterogeneity in computer architecture is the core change in the way computer engineers work to develop and evolve the next generation of computer architectures. If we define the quantum computer as an accelerator technology, it becomes consistent with the heterogeneous multi-core philosophy. In this sense, work on quantum computing seems to be a logical next step to substantially improve the computing power of high-performance computers. To understand this line of research, we must start by defining the steps required in developing a quantum computer and addressing how such a computer can be connected to classical computers.

One main reason for adopting a quantum system view is the current movement to cloud-based computer architectures where powerful computer servers exist in data centers and are accessed by the end-user through the Internet. This change in processing implies also a change in applications to comply to this new technical standard. This helps to solve the issue that most fundamental quantum phenomena such as superposition and entanglement work at very low temperatures, often at the mK level. For instance, the use of superconducting quantum processors requires (in most cases) an operational temperature for the quantum chip of around 20 mK. Such low temperatures require the use of expensive vacuum and control systems that are difficult and costly to implement in a distributed and decentralised way, but could easily be accessed through a cloud-based system.

As shown in Figure 1, a heterogeneous multi-core system architecture is defined in which GPUs, FPGAs, *Tensor Processing Units* (TPUs) and now also quantum accelerators can all be used. This means that end-user application developers are capable of programming their source code to be compiled and executed on the quantum device.



```
If(cond)
Call QFT(n)
Else
F(n)
....
```

Fig. 1: System architecture with heterogeneous accelerators.

B. Current Challenges in Quantum Computing

R. Feynman's 1982 paper on quantum computing [37] instigated a world-wide research dive into quantum computing. The focus has been on overcoming significant low-level challenges which has led to the development of, for instance, the superconducting qubits. The design of proof-of-concept quantum algorithms and their study with respect to their theoretical complexity show computational improvements over classical algorithms, and this capacity has received significant attention. However, we still need substantial progress in both the low-level design and programming domains. Qubits with sufficiently long coherence times combined with true quantum-killer applications are crucial requirements that have not yet been met by the community. They are vital to demonstrating the exponential performance increase of quantum over conventional computers *in practice* and are urgently needed to convince quantum skeptics about the usefulness of quantum computing. These developments are necessary for quantum computing to become a mainstream technology within the coming 10 to 15 years. However, much more work is needed before a fully-functioning computational device which connects the algorithmic level to the physical chip can be created. The requirements of such a device include: a compiler, run-time support, and most importantly a micro-architecture that executes a well-defined set of quantum instructions. So the development of any quantum computer will be limited, at least in the next 15 years, to the development of the layers that the Quantum Accelerator will need to have. Figure 2 shows the full system stack that any quantum accelerator should have and be actively executing the quantum part of the big system application and thus interacting with the controlling classical processor.

C. Real-World Quantum Computing Applications

As we mentioned, one of the challenges is to find a killer application. The highest level in the system stack for Figure

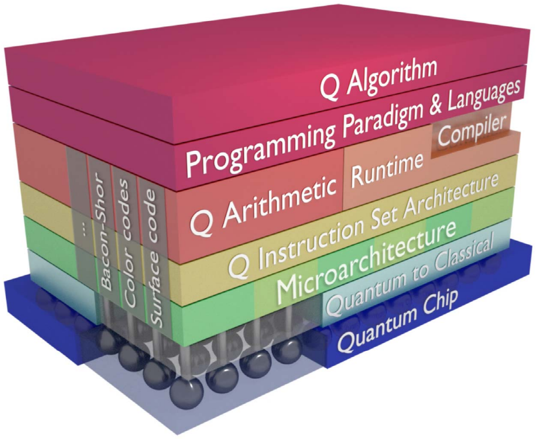


Fig. 2: The required layers of a Quantum Computer.

2 is the application layer where the potential end-user of the quantum computing system will provide the instructions which include computational needs. Quantum computing promises to become a computational game changer, allowing for much faster calculation of various algorithms (in some cases, exponentially faster) than their classical counterparts. Applications with large sets of data are particularly suitable for processing by quantum computers. The cryptography domain is a clear candidate for quantum computing as algorithms such as Shor's factorization have shown that a quantum computer has the potential to break any RSA-based encryption by finding the prime factors of the public key [38] and then, based on that, easily calculating the private key. In anticipation of this, the cryptography domain has already established a post-quantum cryptography research domain.

Another potential application area for quantum computing is the biological domain specific to chemistry and pharmacology. As an example, consider the contemporary focus on genome sequencing. A quantum computer would be necessary to compute the DNA-profile of every human being in the world; this takes currently one week on a large network of extremely powerful servers for a single person's DNA. With enough qubit capacity, the entire inputted data-set can be encoded simultaneously as a superposition of a single wave function.[1] This particular property makes it possible to perform the computation of the entire data-set in parallel. This kind of computational acceleration provides a promising approach to addressing the computational challenges of DNA-analysis algorithms where both character-based and sequence-based correlation analyses are required. Regarding genome sequencing, we have to investigate whether the quantum approach can be used to calculate the similarity between two different DNA sequences.

III. Intrinsic Computing using weakly coupled oscillators

Currently, most digital information processing is performed by conventional CMOS-based devices; these consume significantly more power than mixed-signal systems based on emerging devices and architectures. This power consumption is the main hindrance for deploying highly parallel ICs in small form factor and their embedded application domains, and this is due to limited available power budgets and insufficient heat-sinking mechanisms. For this reason, emerging devices are preferred to CMOS devices for information processing.

Certain computer vision algorithms such as corner detection and pattern matching consist of a high number of multiply-accumulate and fractional norm operations which require significant computational resources [39]. Most of these algorithms exhibit a high degree of parallelism and are also latency-critical in real-time applications. Leveraging the parallelism to speed up the computation in real-time applications requires either stacking CMOS transistors or an efficient pipeline mechanism with intermediate staging or buffering storage requirements, both of which suffer from large gate counts and increased power consumption. In recent years, nano-oscillator based computing paradigms have garnered significant interest owing to their inherent system dynamics which can be used to solve computationally hard problems in computer vision, optimization and neuromorphic applications.

In [39], an array of weakly coupled oscillators is shown to synchronize when coupled together with close initial states.These synchronized oscillatory systems can be leveraged to perform several associative functions in a wide variety of computer vision applications [40; 41]. The efficiency of a coupled oscillator-based system in terms of power and area has been shown in computer vision problems such as vertex coloring of graphs [42] and morphological image processing [43]. Most of these applications use the coupled oscillator-based systems to compute a variety of fractional norms or l_k norms, where the l_k norm of x is defined as $\|x\|_p = \sqrt[k]{\sum_i |x_i|^k}$.

Recently, in [44] a coupled oscillator-based co-processor has been proposed to accelerate computations like sorting, degree of matching, etc. for use in applications such as pattern recognition, clustering, and text recognition. A computation model based on vanadium dioxide (VO_2)-based coupled oscillators is demonstrated in this section and used to illustrate the efficiency of a coupled oscillator-based implementation of a corner detection algorithm compared to a CMOS-based implementation.

A. A VO_2-based Coupled Oscillator System

VO_2 undergoes a volatile and sharp *Insulator-to-Metal Phase Trransition* (IMT) with an applied electrical bias. When a resistor is connected in series with the VO_2 such that the load line passes through the unstable regions of the hysteretic I-V curve, it enables continuous relaxation oscillations in a compact *one-transistor and one-resistor* (1T1R) configuration [40]. The replacement of the series resistor with a transistor allows control of the frequency of oscillation through the transistor gate voltage which adjusts the effective series resistance seen by the IMT device.

Electrical coupling between two oscillators is achieved through simple resistive and capacitive elements. Individually, each oscillator can operate at a range of frequencies as controlled by the series transistor's gate voltage (V_{gs}), and when the frequencies of two coupled oscillators are sufficiently close to each other the coupling elements facilitate *frequency locking*, as seen in Figure 3. The phase difference between two synchronized waveforms is governed by the frequency difference of the uncoupled oscillators as well as the strength

[1] Though not an exact calculation, given the size of the genome, the number of qubits would have to be at least in the millions.

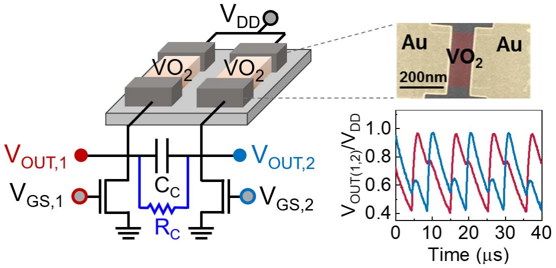


Fig. 3: An IMT device coupled through its Resistive and Capacitive elements showing frequency locking.

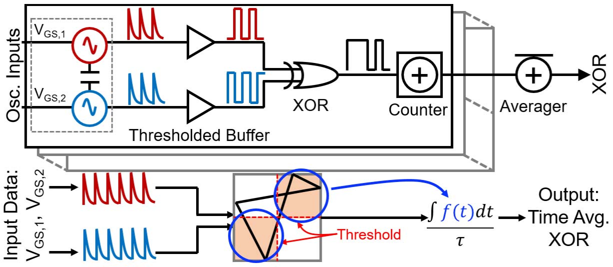


Fig. 4: Readout scheme for pairwise coupled oscillators based on a thresholded and time-averaged XOR.

of the coupling network, i.e. the values of the resistive and capacitive elements.

One pathway towards non-traditional computing with coupled oscillators involves encoding information (input values) in the gate voltages (V$_{gs1}$, V$_{gs2}$) of the series transistors to manipulate the phase-difference which can be quantified through a readout circuit. To demonstrate this, we designed an XOR-based readout, seen in Figure 4, that takes synchronized waveforms as its input and performs a threshold-XOR operation to be time-averaged over a certain number of cycles to provide a stable output value. For a large range of coupling strengths, two nearly-identical oscillators always have the [1-Avg(XOR)] measure minima near the point where Δ V$_{gs}$ = (V$_{gs1}$-V$_{gs2}$) is zero. For increasing coupling strengths, (that is, decreasing R$_C$), the shape of the curves around the minima point follow increasing l_k norms ((V$_{gs1}$ -V$_{gs2}$)^k) ranging from almost $(k \sim 1.6)$ to parabolic $(k \sim 2.0)$ to extremely nonlinear $(k \sim 3.4)$ as shown in Figure 5. Further away from the minima the curves take on a fractional norm shape $(k < 1)$ for a short range before becoming irregular near the edge of the locking range.

B. Corner Detection Using the Coupled Oscillator-based Distance Norm

Corner detection is the process of identifying the important features or interest points in an image which can then be used to infer its contents. It is one of the fundamental preprocessing steps in computer vision and is widely used in applications such as object detection, motion recognition and tracking, and 3D modeling and scene construction. *Features from Accelerated Segment Tests* (FAST) [45] is one of the most common corner detection algorithms owing to its computational efficiency, as it is both faster and uses less computational resources than other similar algorithms. The FAST corner detection algorithm compares a pixel with its surrounding 16 pixels on a Bresenham circle of radius 3. If the pixel is either darker or brighter than the N contiguous pixels by a certain threshold on the circle, it will be marked as a corner. The FAST algorithm involves a series of systematic and parallel comparison operations, and these operations are performed using coupled oscillator distance norms for a demonstration of the efficiency of this computation method.

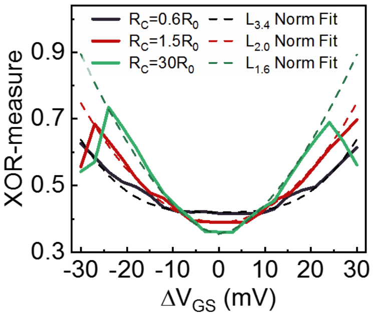


Fig. 5: XOR-measure showing various l_k norms achieved with coupled oscillators.

FAST corner detection using coupled oscillator distance norms involves two processing steps. First, a pixel under test is compared with its 16 surrounding pixels and then results of the norm operation are checked against a threshold to identify the darker/brighter pixels. The intensities of the pixels under comparison are then fed as voltages to the coupled oscillator distance metric computation primitive for the comparison operation. The distance metric gives an approximation of absolute difference between the two voltages, but the direction of the difference (that is, whether a pixel is brighter or darker than the other pixel) is not known and does not matter. The FAST algorithm requires a pixel to be either brighter or darker than N contiguous surrounding pixels. To avoid false positives, if the first step outputs any contiguous pixels which meet the threshold conditions, we compare the adjacent pixels in the result set with each other to check if they are similar. If any of the difference values are greater than two times the threshold, then we can classify the result set as a false positive. The entire system data flow is depicted in Figure 6.

Even though the coupled oscillator-based distance norms provide low power hardware primitives for comparison, tuning across computational layers might be needed to incorporate them. For this case, we must do two comparison steps instead of the one required for the baseline software algorithm. Unlike CMOS-based accelerators, our design can be closely integrated with analog image acquisition hardware, as the inputs to the comparison hardware are variable voltage signals. This is significant because it benefits on-the-fly computer vision applications by reducing the amount of on-chip storage required for processing the entire image. This is because it requires only the corner information to be stored in on-chip memory for further processing. The power consumption of the coupled oscillator-based block designed in this example to identify corners is 0.936 mW (including the XOR readout), whereas the power consumption of the corresponding CMOS implementation at the 32 nm process node is 3 mW, which shows the powerful benefit of using a computational model based on coupled oscillators.

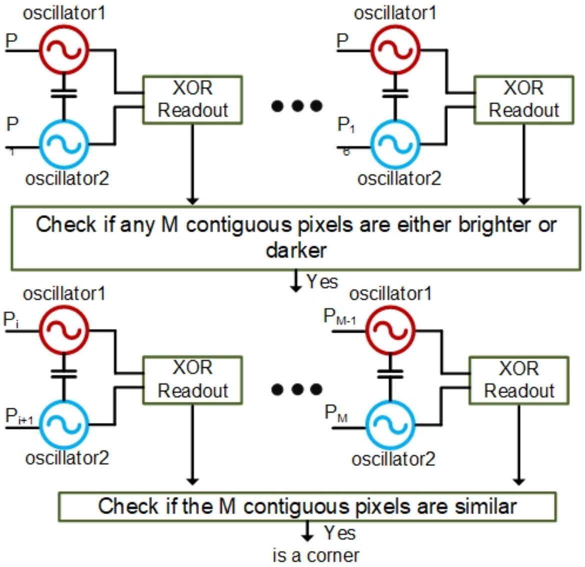


Fig. 6: FAST corner detection using coupled oscillator distance norms.

IV. The memcomputing paradigm

In this section we discuss yet another computing paradigm that was recently suggested: memcomputing [30; 46; 47], which stands for computing *in* and *with* memory (time non-locality) [46]. Memcomputing is based on the premise that the processing of information is accomplished by appropriate memory systems, without the need to transfer information from a processing unit to a memory unit, thus overcoming the von Neumann bottleneck completely. The physical memory units (memprocessors) that perform the computation can be realized in practice using a variety of systems and materials [30]. Irrespective, the memory components comprise both *active* elements (e.g., transistors) and *passive* elements with memory (e.g., resistive memories) and without memory (e.g., standard resistors, capacitors, etc.). The active elements are fundamental to this computing paradigm since they provide the necessary *feedback* to guide the machine towards the solution of the problem it is meant to solve.

Although memcomputing machines can be defined as both digital and analog [30], the digital ones are those that are easily scalable [47]. This is due to the fact that in digital computing the input and the output can be read/written with finite precision, independent of the size of the machine. In order to solve a specific combinatorial optimization problem, *Digital Memcomputing Machines* (DMMs) are then designed as follows. The problem is first written in Boolean form (or in algebraic form if the problem is an integer linear programming one, as seen in [48]). The corresponding Boolean circuit is not even unique, in view of the freedom available in choosing different logic gates as the basis of our Boolean logic [49]. The gates of the circuit are then replaced by *Self-Organizing Logic Gates* (SOLGs) [47] (or self-organizing algebraic gates for algebraic problems [48]), whose only requirement is to self-organize into the correct logical proposition of the given gate *irrespective* of whether the signal comes from the traditional inputs or the traditional outputs. In other words, SOLGs are *terminal agnostic*, although not necessarily invertible in a one-to-one sense.

As previously mentioned, physical realizations of these gates employ a combination of circuit elements with and without memory. When assembled together to form the full Boolean circuit representing a given problem, these gates then define a physical electronic circuit that can be built using available technology. In fact, resistive memory components can be emulated by an appropriate combination of active elements [50], thus avoiding the need to integrate circuit components outside of the traditional CMOS technology. The original problem is then solved by applying the appropriate signals at specific input terminals, and then letting the circuit reach a steady-state. The signals at the appropriate output terminals then represent the solution to the original problem.

Although a hardware implementation of DMMs would be ideal, these machines are non-quantum systems. Therefore, it is natural to ask if the equations of motion describing their circuits offer any advantage when simulated on our standard computers.

To address this, we first note that the physical DMMs are described by non-linear ordinary differential equations whose point attractors represent the solution to the original problem. These equations may assume different forms [47] so long as they represent the correct properties of SOLGs. One tidy form for a given gate terminal i can be written as

$$\dot{v}_i = \Delta g_M x \Delta V_M + g_R \Delta V_R, \tag{1}$$

$$\dot{x} = h(\Delta V_M, x), \quad x \in [0, 1], \tag{2}$$

for the voltage v_i representing the variable i and the memory state variable x. The first and second terms on the RHS of Eq. 1 represent the contributions from resistors with memory and standard resistors, respectively. Δg_M (ΔV_M) and g_R (ΔV_R) are the respective conductances (voltage drops) and h is a function of the memory state variable.

In order to represent valid DMMs, the above equations must correspond to *point-dissipative systems* [51]. This guarantees that trajectories are bounded and will always converge to an invariant set that is uniformly asymptotically stable, and that if the solution to the original problem exists, then the system will find it [47], and no periodic orbits [52] or chaos [53] can coexist. The approach to equilibrium of these dynamical systems would then solve the original problem.

Recent work has shown that simulations of DMMs perform much better than traditional algorithmic approaches on a wide variety of combinatorial optimization problems [32; 48; 54; 55; 56]. For instance, in [54] it was shown that these simulations outperform specialized software specifically designed to tackle maximum satisfiability problems. In [55] simulations of DMMs were employed to the training of *Restricted Boltzmann Machines* (RBMs) that are difficult to pre-train. The results have shown that by simulating DMMs one can accelerate (in number of iterations) the pre-training of RBMs as much as the reported hardware application of the quantum annealing method implemented by the D-Wave machine on the same network and data set [57]. However, the memcomputing approach is found to perform far better than the D-Wave machine in terms of *training-quality* [55]. In addition, the memcomputing approach has been shown to maintain a quality advantage ($> 1\%$ in accuracy, corresponding to a 20% reduction in error rate) over state-of-the-art supervised-learning training.

To better understand the physical reason behind this efficiency, [58] shows both numerically and analytically that the transient dynamics of DMMs proceeds via a succession of classical trajectories (instantons) that connect critical points (namely the zero solutions of Eqs. 1 and 2) in the phase space with different stability (indexes).

The existence of these instantonic trajectories is the reason for the *Dynamical Long-Range Order* (DLRO) in the system. DLRO means that distant parts of the machine can correlate during dynamics, thus allowing the efficient computation of complex problems. Another advantage of these transient dynamics is that the critical points are topological objects, so that their number and stability cannot be easily changed by perturbations and noise unless the topology of the phase space itself is changed. This would require changing the physical circuit itself.

As a consequence, the solution search of DMMs is very robust to external perturbations, a fact that has also been shown explicitly by adding noise to Eqs. 1 and 2 [59]. Recently, this DLRO was more clearly demonstrated in the solution of a difficult problem that is particularly important in physics: the problem of the frustrated-loop using spin glass [56]. In this case, it was shown that DMMs allow for the *collective* flipping of clusters of spins spanning the entire lattice, as if the system underwent a continuous phase transition.

All these studies show that memcomputing, and in particular its digital (hence scalable) version, is a promising alternative to the present computing paradigm capable of tackling a variety of problems of interest to science and technology.

V. Conclusions

This paper addressed the rising need for disruptive computing models beyond our current von Neumann computing model to continue to sustain growing computational needs. Beyond a brief survey of existing emerging computing models, three computing solutions were presented and the benefits provided by these suggestions were thoroughly discussed. Quantum computing was considered from its most basic physics to the practicalities of making a useful computer, including the necessary micro architecture (qubits) and programming capabilities. The challenges of developing such a system were outlined, along with some hopeful applications. Also presented were novel ways of computing using intrinsic oscillators. The physics of this idea has been around for over 40 years but computing applications have not been considered until recently. In particular, this section suggested a physical computation model based on coupled oscillators made from VO_2 and showed the efficiency of such a model when compared to a CMOS-based computation. Finally, memcomputing, which uses self-organizing logic gates to solve complex computing problems efficiently, was discussed. The digital form of this model was considered specifically in terms of training RBMs and other optimization problems.

Acknowledgements

The authors would also like to acknowledge for their support the NSF Career Award number 1751064, the SRC Contract 2018-IN-2834, the Nanoelectronics Research Corporation (NERC), a wholly-owned subsidiary of the Semiconductor Research Corporation (SRC), through Extremely Energy Efficient Collective Electronics (EXCEL), an SRC- NRI Nanoelectronics Research Initiative under Research Task IDs 2698.001, and 2698.004, the Center for Memory Recording Research at UCSD and MemComputing, Inc. *(http://memcpu.com/)*.

Bibliography

[1] M. Soeken, P. Gaillardon, S. Shirinzadeh, R. Drechsler, and G. D. Micheli, “A plim computer for the internet of things,” *Computer*, vol. 50, no. 6, pp. 35–40, 2017.

[2] K. Bernstein, R. K. Cavin, W. Porod, A. Seabaugh, and J. Welser, “Device and architecture outlook for beyond cmos switches,” *Proceedings of the IEEE*, vol. 9, no. 12, pp. 2169–2184, 2010.

[3] D. E. Nikonov and I. A. Young, “Benchmarking of beyond-cmos exploratory devices for logic integrated circuits,” *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 1, pp. 3–11, 2015.

[4] A. D. Franklin, M. Luisier, S.-J. Han, G. Tulevski, C. M. Breslin, L. Gignac, M. S. Lundstrom, and W. Haensch, “Sub-10 nm carbon nanotube transistor,” *Nano Letters*, vol. 12, no. 2, pp. 758–762, 2012.

[5] K. S. Novoselov, A. K. Geim, S. V. Morozov, D. Jiang, Y. Zhang, S. V. Dubonos, I. V. Grigorieva, and A. A. Firsov, “Electric field effect in atomically thin carbon films,” *Science*, vol. 306, no. 5696, pp. 666–669, 2004.

[6] B. Radisavljevic, A. Radenovic, J. Brivio, V. Giacometti, and A. Kis, “Single-layer mos2 transistors,” *Nature Nanotechnology*, vol. 6, no. 3, pp. 147–150, 2011.

[7] S. A. Wolf, D. D. Awschalom, R. A. Buhrman, J. M. Daughton, S. V. Molnar, M. L. Roukes, A. Y. Chtchelkanova, and D. M. Treger, “Spintronics: a spin-based electronics vision for the future,” *Science*, vol. 294, no. 5546, pp. 1488–1495, 2001.

[8] P.-E. Gaillardon, L. Amaru, J. Zhang, and G. D. Micheli, “Advanced system on a chip design based on controllable-polarity fets,” *DATE Tech. Dig.*, 2014.

[9] J. Zhang, X. Tang, P. Gaillardon, and G. D. Micheli, “Configurable circuits featuring dual-threshold-voltage design with three-independent-gate silicon nanowire fets,” *IEEE TCAS I*, vol. 61, no. 10, pp. 2851–2861, 2014.

[10] F. M. Ham and I. Kostanic, “Principles of neurocomputing for science and engineering,” *McGraw-Hill Higher Education*, 2000.

[11] J. Seo, B. Lin, M. Kim, P.-Y. Chen, D. Kadetotad, A. M. Z. Xu, S. Vrudhula, S. Yu, J. Ye, and Y. Cao, “On-chip sparse learning acceleration with cmos and resistive synaptic devices,” *IEEE Transactions on Nanotechnology, Special Issue on Cognitive and Natural Computing with Nanotechnology*, vol. 14, no. 6, pp. 969–979, 2015.

[12] T. F. Wu, H. Li, P.-C. Huang, A. Rahimi, J. M. Rabaey, H.-S. P. Wong, M. M. Shulaker, and S. Mitra, “Brain-inspired computing exploiting carbon nanotube fets and resistive ram: Hyperdimensional computing case study,” *Digest of Technical Papers - IEEE International Solid-State Circuits Conference*, vol. 61, pp. 492–494, 2018.

[13] J. Seo, B. Lin, M. Kim, P.-Y. Chen, D. Kadetotad, Z. Xu, A. Mohanty, S. Vrudhula, S. Yu, J. Ye, and Y. Cao, “Nbox based oscillation neuron for neuromorphic computing,” *Applied Physics Letters*, vol. 111, 2017.

[14] J. Seo, B. Lin, M. Kim, P.-Y. Chen, D. Kadetotad, Z. Xu, A. Mohanty, S. Vrudhula, S. Yu, J. Ye, and Y. Cao,

"Neuro-inspired computing with emerging non-volatile memory," *Proceedings of IEEE*, vol. 106, no. 2, pp. 260–285, 2018.

[15] D. Blount, P. Banda, C. Teuscher, and D. Stefanovic, "Feedforward chemical neural network: An in silico chemical system that learns xor," *Artificial Life*, vol. 23, no. 3, pp. 295–317, 2017.

[16] D. R. B. Ly, A. Grossi, C. Fenouillet-Beranger, E. Nowak, D. Querlioz, and E. Vianello, "Role of synaptic variability in resistive memory-based spiking neural networks with unsupervised learning," *Journal of Physics D: Applied Physics*, vol. 51, no. 44, 2018.

[17] G. Indiveri and T. Horiuchi, "Frontiers in neuromorphic engineering," *Frontiers in Neuroscience*, vol. 5, p. 118, 2011.

[18] G. Indiveri, B. Linares-Barranco, T. Hamilton, A. van Schaik, R. Etienne-Cummings, T. Delbruck, S.-C. Liu, P. Dudek, P. Hfliger, S. Renaud, J. Schemmel, G. Cauwenberghs, J. Arthur, K. Hynna, F. Folowosele, S. SAGHI, T. Serrano-Gotarredona, J. Wijekoon, Y. Wang, and K. Boahen, "Neuromorphic silicon neuron circuits," *Frontiers in Neuroscience*, vol. 5, p. 73, 2011.

[19] D. Vodenicarevic, A. Mizrahi, N. Locatelli, J. Grollier, and D. Querlioz, "Spintronic nanoscillators for unconventional circuits," *European Conference on Circuit Theory and Design*, vol. 61, no. 10, pp. 2851–2861, 2017.

[20] D. Vodenicarevic, N. Locatelli, and D. Querlioz, "A neural network based on synchronized pairs of nano-oscillators," *Proc. of the 17th IEEE International Conference on Nanotechnology*, 2017.

[21] P. Gaillardon, L. Amar, A. Siemon, E. Linn, R. Waser, A. Chattopadhyay, and G. D. Micheli, "The programmable logic-in-memory (plim) computer," *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 427–432, March 2016.

[22] A. Haj-Ali, R. Ben-Hur, N. Wald, R. Ronen, and S. Kvatinsky, "Imaging: In-memory algorithms for image processing," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 12, pp. 4258–4271, 2018.

[23] C. Teuscher, "The weird, the small, and the uncontrollable: Redefining the frontiers of computing," *Computer*, vol. 50, no. 8, pp. 52–58, 2017.

[24] S. Jain, A. Ranjan, K. Roy, and A. Raghunathan, "Computing in memory with spin-transfer torque magnetic ram," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 3, pp. 470–83, 2018.

[25] A. Aziz, E. T. Breyer, A. Chen, X. Chen, S. Datta, S. K. Gupta, M. Hoffmann, X. S. Hu, A. Ionescu, M. Jerry, T. Mikolajick, H. Mulaosmanovic, K. Ni, M. Niemier, I. O'Connor, A. Saha, S. Slesazeck, S. K. Thirumala, and X. Yin, "Computing with ferroelectric fets: Devices, models, systems, and applications," in *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1289–1298, 2018.

[26] J. Hoffman, X. Pan, J. W. Reiner, F. J. Walker, J. P. Han, C. H. Ahn, and T. P. Ma, "Ferroelectric field effect transistors for memory applications," *Advanced Materials*, vol. 22, pp. 2957–2961, 2010.

[27] A. Agrawal, A. Jaiswal, and K. Roy, "X-SRAM: enabling in-memory boolean computations in CMOS static random access memories," *CoRR*, vol. abs/1712.05096, 2017.

[28] J. Zhang, Z. Wang, and N. Verma, "In-memory computation of a machine-learning classifier in a standard 6t sram array," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 4, pp. 915–924, 2017.

[29] H. Manukian, F. L. Traversa, and M. D. Ventra, "Memcomputing numerical inversion with self-organizing logic gates," *CoRR*, vol. abs/1612.04316, 2016.

[30] F. L. Traversa and M. Di Ventra, "Universal memcomputing machines," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, p. 2702, 2015.

[31] Y. R. Pei, F. L. Traversa, and M. D. Ventra, "On the universality of memcomputing machines," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11, 2018.

[32] M. Di Ventra and F. L. Traversa, "Memcomputing: Leveraging memory and physics to compute efficiently," *J. Appl. Phys.*, vol. 123, p. 180901, 2018.

[33] M. D. Ventra and F. L. Traversa, "Memcomputing: An efficient topological computing paradigm," *2017 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS)*, pp. 1–2, 2017.

[34] D. Deutsch, "Quantum theory, the church-turing principle and the universal quantum computer," *Proceedings of the Royal Society of London - A. Mathematical and Physical Sciences*, vol. 400, no. 1818, pp. 19–117, 1985.

[35] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Quantum computation by adiabatic evolution," *arXiv preprint quant-ph/0001106*, 2000.

[36] I. L. Markov, "Limits on fundamental limits to computation," *Nature*, vol. 512, 2014.

[37] R. P. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, no. 6-7, pp. 467–488, 1982.

[38] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pp. 124–134, Nov 1994.

[39] T. Shibata, R. Zhang, S. P. Levitan, D. E. Nikonov, and G. I. Bourianoff, "Cmos supporting circuitries for nano-oscillator-based associative memories," *2012 13th International Workshop on Cellular Nanoscale Networks and Their Applications (CNNA)*, pp. 1–5, 2012.

[40] N. Shukla, A. Parihar, M. Cotter, M. Barth, X. Li, N. Chandramoorthy, H. Paik, D. G. Schlom, V. Narayanan, A. Raychowdhury, and S. Datta, "Pairwise coupled hybrid vanadium dioxide-mosfet (hvfet) oscillators for non-boolean associative computing," *2014 IEEE International Electron Devices Meeting*, pp. 28.7.1–28.7.4, 2014.

[41] M. J. Cotter, Y. Fang, S. P. Levitan, D. M. Chiarulli, and V. Narayanan, "Computational architectures based on coupled oscillators," *2014 IEEE Computer Society Annual Symposium on VLSI*, pp. 130–135, 2014.

[42] A. Parihar, N. Shukla, M. Jerry, S. Datta, and A. Raychowdhury, "Vertex coloring of graphs via phase dynamics of coupled oscillatory networks," *Scientific Reports*, vol. 7, no. 911, 2017.

[43] N. Shukla, W. . Tsai, M. Jerry, M. Barth, V. Narayanan, and S. Datta, “Ultra low power coupled oscillator arrays for computer vision applications,” *2016 IEEE Symposium on VLSI Technology*, pp. 1–2, 2016.

[44] N. Gala, S. Krithivasan, W.-Y. Tsai, X. Li, V. Narayanan, and V. Kamakoti, “An accuracy tunable non-boolean co-processor using coupled nano-oscillators,” *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 14, no. 1, 2018.

[45] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” *Computer Vision – ECCV 2006*, pp. 430–443, 2006.

[46] M. Di Ventra and Y. V. Pershin, “The parallel approach,” *Nature Physics*, vol. 9, pp. 200–202, 2013.

[47] F. L. Traversa and M. Di Ventra, “Polynomial-time solution of prime factorization and np-complete problems with digital memcomputing machines,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, p. 023107, 2017.

[48] F. L. Traversa and M. Di Ventra, “Memcomputing integer linear programming,” *arXiv:1808.09999*, 2018.

[49] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[50] Y. V. Pershin and M. Di Ventra, “Experimental demonstration of associative memory with memristive neural networks,” *Neural Networks*, vol. 23, no. 7, pp. 881–886, 2010.

[51] J. Hale, *Asymptotic Behavior of Dissipative Systems*, vol. 25 of *Mathematical Surveys and Monographs*. Providence, Rhode Island: American Mathematical Society, 2nd ed., 2010.

[52] M. Di Ventra and F. L. Traversa, “Absence of periodic orbits in digital memcomputing machines with solutions,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, p. 101101, 2017.

[53] M. Di Ventra and F. L. Traversa, “Absence of chaos in digital memcomputing machines with solutions,” *Phys. Lett. A*, vol. 381, p. 3255, 2017.

[54] F. L. Traversa, P. Cicotti, F. Sheldon, and M. Di Ventra, “Evidence of exponential speed-up in the solution of hard optimization problems,” *Complexity*, vol. 2018, p. 7982851, 2018.

[55] H. Manukian, F. L. Traversa, and M. Di Ventra, “Accelerating deep learning with memcomputing,” *Neural Networks*, vol. 10, p. 1, 2018.

[56] F. Sheldon, F. L. Traversa, and M. Di Ventra, “Taming a non-convex landscape with long-range order: Memcomputing the ising spin glass,” *arXiv:1810.03712*, 2018.

[57] S. H. Adachi and M. P. Henderson, “Application of quantum annealing to training of deep neural networks,” *arXiv preprint arXiv:1510.06356*, 2015.

[58] M. Di Ventra, F. L. Traversa, and I. V. Ovchinnikov, “Topological field theory and computing with instantons,” *Ann. Phys. (Berlin)*, vol. 529, p. 1700123, 2017.

[59] S. R. B. Bearden, H. Manukian, F. L. Traversa, and M. Di Ventra, “Instantons in self-organizing logic gates,” *Physical Review Applied*, vol. 9, p. 034029, 2018.