Self-organization and solution of shortest-path optimization problems with memristive networks

Yuriy V. Pershin[1,*] and Massimiliano Di Ventra[2,†]

[1]*Department of Physics and Astronomy, University of South Carolina, Columbia, South Carolina 29208, USA*

[2]*Department of Physics, University of California, San Diego, California 92093-0319, USA*

We show that memristive networks, namely networks of resistors with memory, can efficiently solve shortest-path optimization problems. Indeed, the presence of memory (time nonlocality) promotes self organization of the network into the shortest possible path(s). We introduce a network entropy function to characterize the self-organized evolution, show the solution of the shortest-path problem and demonstrate the healing property of the solution path. Finally, we provide an algorithm to solve the traveling salesman problem. Similar considerations apply to networks of memcapacitors and meminductors, and networks with memory in various dimensions.

I. INTRODUCTION

Currently, much attention is being devoted to memory circuit elements (memelements) [1,2] and their applications in different domains of electronics [3]. In particular, computing with memelements [4,5]—memcomputing—has received much attention because of the ability of network architectures built out of memelements to store and process information on the same physical platform [4–6]. This is a major conceptual and practical departure from the present day architectures based on the von Neumann machine concept [7]. The strength of memcomputing is essentially based on the analog massively parallel dynamics of many, if not all, memelements in the network and the ability to recover the result(s) of the computation from the same computing units, much like the brain is thought to operate [5,8]. Examples of approaches to compute with memelements include neuromorphic computing with memristive synapses [4,9–14], massively parallel computing with memristive networks [8,15], logic with memory circuit elements [4,6,16,17], and memristive cellular automata [18].

In order to realize this memcomputing paradigm some criteria need to be satisfied. We have introduced and expended on these criteria in Ref. [5]. These are

  (i) scalable massively-parallel architecture with combined information processing and storage,

  (ii) sufficiently long information storage times,

  (iii) the ability to initialize memory states,

  (iv) mechanism(s) of collective dynamics, strong memory content,

  (v) the ability to read the final result (from relevant memelements), and

  (vi) robustness against small imperfections and noise.

One can show that the memcomputing schemes mentioned above satisfy all or the majority of these criteria [5]. We also note that although memristive devices [19] are the main focus of present efforts, memcapacitive and meminductive devices [1] are also of great promise because of their potential to perform calculations at a minimal energy cost.

Previously, we demonstrated a maze problem solution using memristive networks [8]. Clearly, all of the above requirements

are satisfied within this approach [5]. In this paper we consider a more complex case of the shortest-path optimization: the shortest path optimization on a two-dimensional plane. In fact, our approach and results can be easily extended to any hyperplane in an N-dimensional space, with $N > 1$. The difficulty inherent in this problem stems from the fact that any two points of the plane are connected by multiple often degenerate paths, which complicates the finding of a single optimal solution. We show that random networks, without any predefined symmetry directions, are more promising in this regard. In addition, we suggest an algorithm that provides a possible solution to the NP-hard traveling salesman problem. A further development of the suggested algorithm may lead to some quite interesting results on the long way [20] toward finding more efficient solutions of this problem.

The paper is organized as follows. In Sec. II, we discuss essential details of memcomputing schemes utilizing memristive networks. Section III presents our studies of memcomputing with symmetric (regular) networks. In particular, we show that regular networks can find a unique solution if a pair of specified nodes is located along a symmetry direction. In a more general case, however, a degenerate solution is obtained. Moreover, we demonstrate the healing property of the solution path (Sec. III C). Section IV studies memcomputing with random networks suitable to solve the shortest-path problem in any direction (Sec. IV A). We also suggest an algorithm that provides an approximate solution to the traveling salesman problem (Sec. IV B). Finally, we conclude in Sec. V.

II. MEMCOMPUTING WITH MEMRISTIVE NETWORKS

A. Memristive processor

A memristive processor (network) consists of a collection of grid points on a hyperplane of N-dimensional space ($N > 1$) connected by basic units involving memristive elements and traditional elements, such as switches (see Fig. 1). Below, we consider different organizations of the grid points including a regular square array such as that shown in Fig. 1 and a random one on a (two-dimensional) 2D plane. Although not strictly necessary, in the examples we have chosen the design of each basic unit to be symmetric (involving two bipolar memristive elements) so as to conveniently provide independency of the circuit operation on the sign of the applied

---

*pershin@physics.sc.edu

†diventra@physics.ucsd.edu
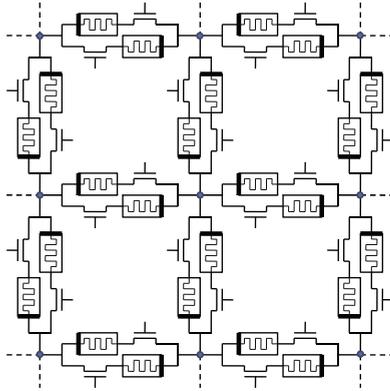
FIG. 1. An example of a memristive processor consisting of a network of memristive elements in which each grid point is attached to several basic units. Each basic unit involves two memristive devices connected symmetrically (in parallel) and two switches (field-effect transistors). The switches provide access to individual memristive devices, while in-parallel connection symmetrizes the response of bipolar memristive elements.

voltage. We also introduce switches for two main functions: (i) to provide independent access to each individual memristive device, and (ii) to define the network topology, if needed (see Ref. [8] for an example). Thus, the architecture of Fig. 1 provides access to each individual memristive device for the purpose of initialization and reading of the calculation result. The calculation consists in the evolution of the network state, defined as the collection of states of all memristive devices, when an appropriate pulse sequence is applied to a subset of grid points.

B. Algorithms

Computation algorithms for memristive processors involve three standard stages: initialization, calculation, and result reading. In this work, we focus on solving the shortest path and traveling salesman problems. The corresponding algorithms are formulated below.

1. The shortest-path problem

In the shortest-path problem, we want to find the shortest path between two specified (input and output) nodes [such as, for example, those shown by dark red (dark gray) arrows in Figs. 2(a) or 2(b)] in the network. The algorithm used for this calculation is the following:

Initialization stage. All memristive devices are pre-initialized into the off (high-resistance) state.

Calculation stage. A voltage pulse of a suitable amplitude and duration is applied to the pair of specified nodes.

Calculation reading. The shortest path is given by a subset of basic units in their on (low-resistance) state [21].

2. Traveling salesman problem

In the traveling salesman problem [20], a salesman is given a set of cities to visit. The objective of the salesman is to find a path, which minimizes the round-trip distance: each of the cities should be visited only once. In order to solve this problem, we will consider a two-dimensional memristive network and select a set of nodes representing all the cities in the set. The actual geographic locations of the cities will be taken into account for the selection of nodes. The basic algorithm for the traveling salesman problem includes the following steps.

Initialization stage. All memristive devices are preinitialized into the off (high-resistance) state.

Calculation stage. A sequence of square voltage pulses of random polarity is applied to randomly selected pairs of nodes representing the cities. During this stage, low-resistance paths in the network will start to develop. Importantly, the memristive network will retain memory on the previously applied pulses, and the subsequent network evolution will be largely based on the topology of the already developed low-memristance paths. In this way, an optimized problem solution emerges. Moreover, we anticipate that the path formation in the network occurs *hierarchically*: at shorter times, the low memristance paths between closely spaced cities are formed and, at longer times, this different group of cities becomes connected. As the electric current tends to flow through the shortest/least resistive path, we expect that the proposed algorithm works out the traveling salesman problem, at least in some cases.

Calculation reading. The shortest path is given by a subset of basic units in their on (low-resistance) state.

III. SELF-ORGANIZATION AND HEALING IN REGULAR NETWORKS

A. Self-organization

We now provide an explicit example of computing with memelements, specifically with regular memristive networks, in order to exemplify even further the criteria given above, and the possibilities offered by this paradigm. Details of the specific memristive systems used and the simulation details can be found in Appendix.

Figure 2(a) shows the initial state of the network when all memristive devices are in their OFF states. At the initial moment of time $t = 0$, a single constant amplitude voltage pulse is applied to the input/output nodes shown by dark red (dark gray) arrows in Fig. 2(a). The final state of the memristive network (the calculation result) is presented in Fig. 2(b). Clearly, two specified nodes are connected by a chain of memristive devices in the ON state giving the shortest-path problem solution. Note that Figs. 2(a) and 2(b) depict the memristance of each basic unit consisting of two memristive devices.

Let us now consider the network evolution, the dynamics of the calculation stage, in more detail. First of all, we would like to mention a similarity between the process of computing as performed by the memristive processor and the ant colony optimization algorithm [22,23]. The latter is an adaptable algorithm inspired by the observation that ants, upon finding food, mark their trails by pheromones thus attracting other ants in order to reinforce the trail closest to their nest. A similar type of reinforcement is observed in the memristive network dynamics. Figure 2(c) shows the current distribution in the network at the initial moment of time when all the memristive devices are in the OFF state. In this case, the current flows
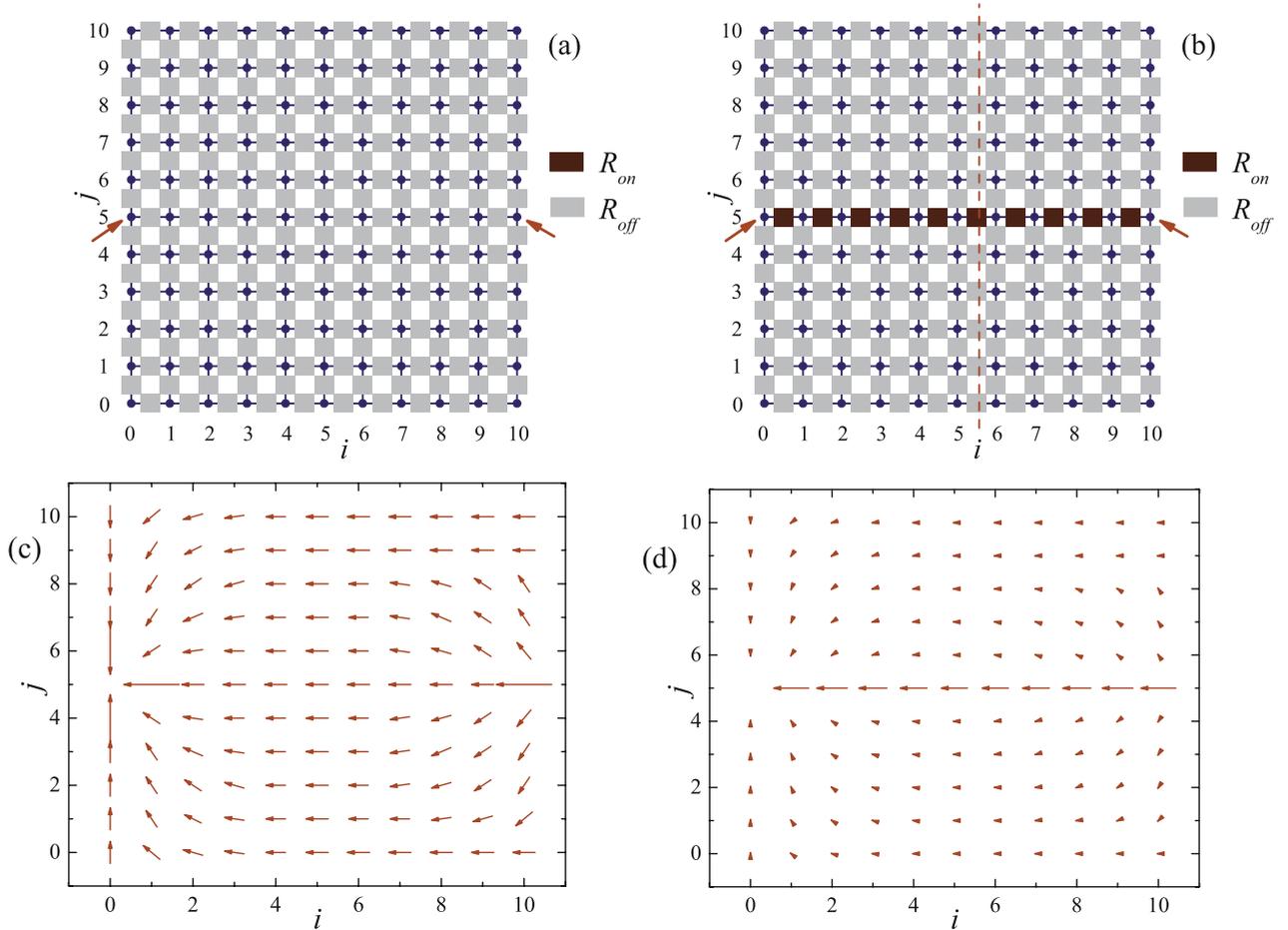
FIG. 2. (Color online) Solution of the shortest path problem for the pair of nodes indicated by dark red (dark gray) arrows in (a) and (b) in a 11×11 memristive network. (a) Initial and (b) final states of the memristive network. Here, the memristance of each basic unit (involving two memristive devices) is represented by a color. The vertical dark red (dark gray) dashed line shows the network cross section used in entropy calculations. Distributions of electron current corresponding to (a) and (b) are shown in (c) and (d), respectively. See Appendix for details of all calculations.

in multiple paths. However, since the rate of memristance change is proportional to the current, the memristance of the least resistive path will decrease faster attracting more and more current. Therefore, the current flowing through the least resistive path will reinforce this path, similarly to the trail reinforcement of the ant colony, see Fig. 2(d).

Moreover, it is interesting to note that the problem solution develops gradually, starting from *both* the specified nodes. This is clearly seen in Fig. 3(a) that presents the rate of change of the memristances along the solution path as a function of time. Note also that the reinforcement of the solution is not supervised, thus implying that the shortest path arises spontaneously as a *self-organized process*. This is an important fundamental result: it is the presence of memory, namely time nonlocality, in the system that leads to self-organization of the dynamics: no memory, no self-organization.

B. Network entropy

In order to quantify the system evolution even further, we define a *network entropy* with respect to currents in the network (similarly, we can use memristances for this purpose).

For example, by considering currents through a vertical cross section of the network at its center [see Fig. 2(b)], we can define the network cross section entropy as [24]

$$\sigma_i(t) = - \sum_{j=0}^{N-1} \tilde{I}_{ij}(t)\ln(\tilde{I}_{ij}(t)), \qquad (1)$$

where N is the number of basic units connected in the horizontal direction [crossed by the dark red (dark gray) dashed line in Fig. 2(b)], j is the index of the row of horizontal basic units crossed by the cross section, $\tilde{I}_{ij} = I_{ij}/I_{tot}$ is the normalized current through a horizontally connected basic unit, and $I_{tot} = \sum_{j=0}^{N-1} I_{ij}$. Figure 3(b) demonstrates (with $N = 11$ and $i = 6$) that the network entropy decreases in time as the computation proceeds. In statistical physics, the entropy is related to the number of states available to the system. Here, its decrease can be thus interpreted as due to the decrease in the number of paths available for the current, with a more pronounced decrease the larger the memory content in the system (as represented by the ratio R_{off}/R_{on}). Alternatively, one can consider an entropy defined for the complete network, namely, $\sigma(t) = - \sum_{ij} \tilde{I}_{ij}(t)\ln(\tilde{I}_{ij}(t))$, where the summation is
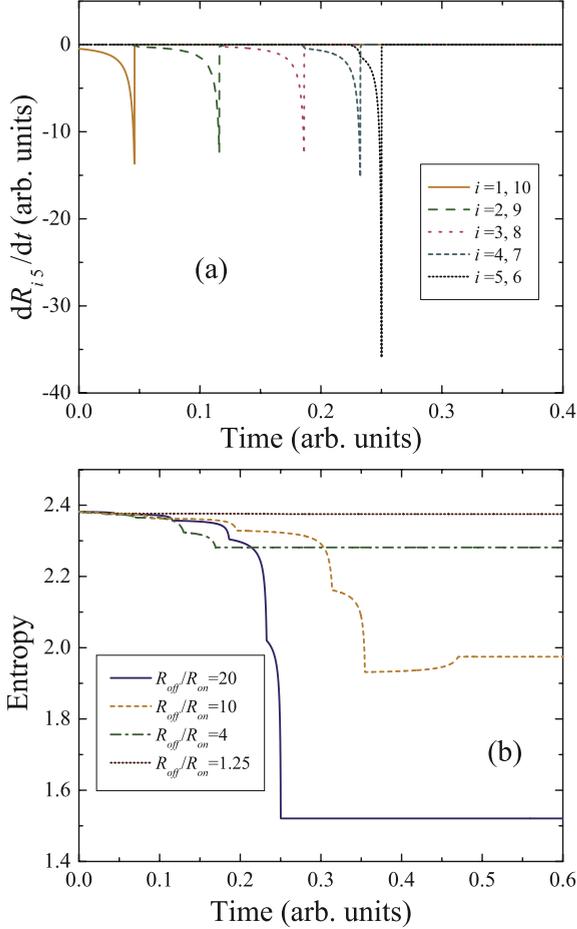
FIG. 3. (Color online) (a) Dynamics of resistance switching within the calculation stage corresponding to the shortest-path problem solution presented in Fig. 2. This plot shows that the solution emerges from both sides and propagates to the center. (b) Network entropy as a function of time from Eq. (1) for networks of different memory content. A slight increase of the entropy for the $R_{\rm off}^M/R_{\rm on}^M = 10$ curve at the time of about 0.4 (arb. units) is due to a delayed partial switching of four vertical units directly connected to the input/output nodes.

performed over all edges in the network. Qualitatively, its time dependence is expected to be close to that shown in Fig. 3(b) for the cross section entropy.

In order to show that networks with higher memory content provide a better solution, we perform a calculation similar to that presented in Fig. 2 assuming, however, a much smaller difference between $R_{\rm on}^M$ and $R_{\rm off}^M$ of each memristive device ($R_{\rm off}^M/R_{\rm on}^M = 1.25$ compared to $R_{\rm off}^M/R_{\rm on}^M = 20$ used in the previous calculation). Figure 4 demonstrates that now the solution of the shortest-path problem can not be found exactly at the given bias. In fact, in addition to the switching of memristive devices directly connecting the input and output nodes, many other memristive devices are also switched into the ON state (see Fig. 4). This example demonstrates the importance of the strong memory content requirement [criterion (iv) discussed in the Introduction, see also Ref. [5]].

In the general case, however, when the selected nodes are located along an arbitrary direction (different from the network symmetry directions), regular networks fail to uniquely find
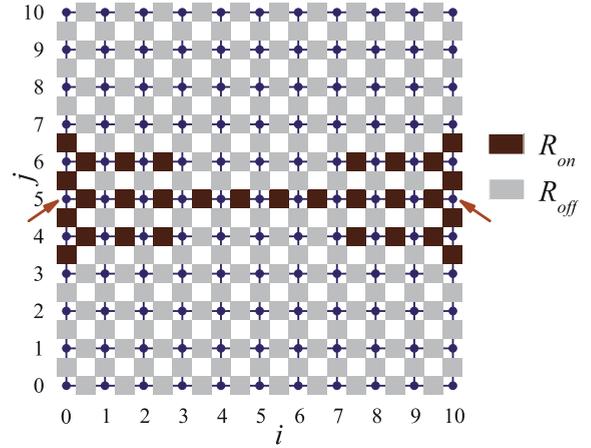
FIG. 4. (Color online) Solution of the shortest-path problem by a network of low memory content, $R_{\rm off}^M/R_{\rm on}^M = 1.25$.

the problem solution. We demonstrate this feature in Fig. 5 showing a shortest-path problem solution, which is doubly degenerate and developed along the symmetry directions of the network. This is a direct consequence of the network symmetry. Therefore, we expect that networks without any local or global symmetry (random networks) or ordered networks but not periodic, such as quasiperiodic networks [25], are best candidates to solve shortest-path optimization problems on the hyperplane. In Sec. IV below we indeed show that random networks can find the proper solution for arbitrary oriented selected nodes.

C. Healing of the damaged solution

In order to analyze criterion (vi) discussed in the Introduction in more detail, let us now damage the shortest-path problem solution shown in Fig. 2(b) (as well as the network) by removing three horizontal basic units in the central part of the network as shown in Fig. 6(a). We note that, due to memory, the memristive network has a remarkable ability to *repair* damaged solutions—the healing ability we have mentioned
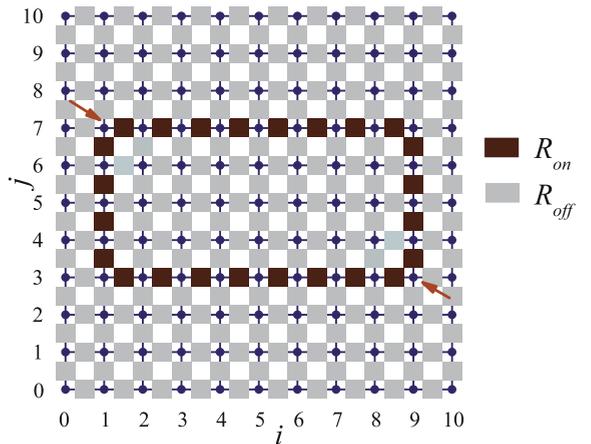
FIG. 5. (Color online) Solution of the shortest-path problem by a network of high memory content, $R_{\rm off}^M/R_{\rm on}^M = 10$, when the input and output nodes (shown by arrows) are selected not along a symmetry direction. Two possible solutions are found.
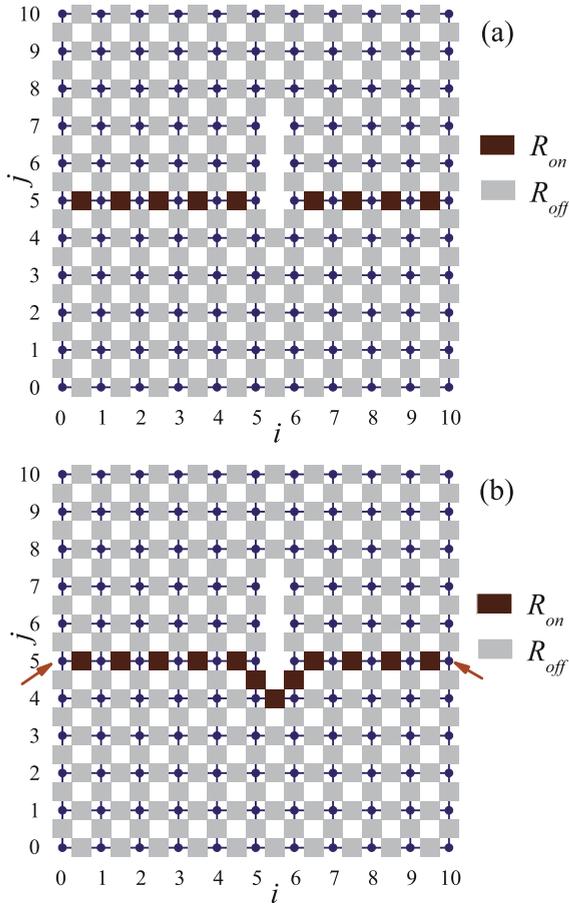
FIG. 6. (Color online) Healing (b) of a damaged (a) solution. To heal the solution damage in (a), a single square pulse of appropriate width and duration is applied to the input and output nodes shown by the dark red (dark gray) arrows in (b).

above. Indeed, this property is close to the *self-healing* ability that can be ascribed to systems or processes, which by nature or by design tend to correct any disturbances.

The healing of the damaged solution is performed by applying a single pulse of a certain amplitude and duration to the input and output nodes. The result presented in Fig. 6(b) shows that a new path connecting two pieces of the initial shortest-path solution develops below the damaged region. The three missing connections have been removed intentionally in an asymmetric fashion in order to show that the healing occurs along the shortest possible path around the damaged region.

It is easy to understand the origin of this healing process: as soon as we switch on the pulse between the input and the output nodes, the current will flow through all possible paths. However, the shortest one is again the one that is mostly affected, and thus reinforced during dynamics.

The stability of the shortest-path problem solution to small imperfections of the system (e.g., finite width distributions of threshold current, limiting values of memristance, etc.) is evident, and therefore does not deserve a closer inspection.

IV. RANDOM NETWORKS

Let us now discuss the use of networks with no local/global symmetry in the solution of optimization problems. Such

random networks are generated using the following algorithm. The nodes are placed randomly in a $Na \times Na$ square area where N is a number and a is a unit length with the constraint that no two nodes can be closer than $0.9a$. After that, any two nodes located closer than $1.5a$ are connected by a memristive edge. We note that the random networks obtained in this way may involve some local intersections of edges that do not play any significant role in the network dynamics, but in principle could be realized on a chip. Therefore, we do not bother with resolving the crossings. Moreover, we assume that each edge is of a similar structure to that shown in Fig. 1 and, on the hardware level, all operations such as initialization, dynamics, and reading of the final results are performed as described in Sec. II. Therefore, the only difference between regular and random networks is in the network topology.

A. The shortest-path problem

Figure 7 shows solutions of the shortest-path problem found with random networks for two different choices of specified nodes. In both cases, the solution is very close to the shortest geometrical solution. However, in some cases, small deviations from the shortest solution are possible such as a small bump in Fig. 7(b) starting at the second node of the solution path bypassing an area of closely spaced nodes (follow the solution path from the left bottom specified node). We associate such deviations with local inevitable fluctuations of network resistivity. Since, initially, each edge in the network is of the same resistance, longer edges are more likely to be
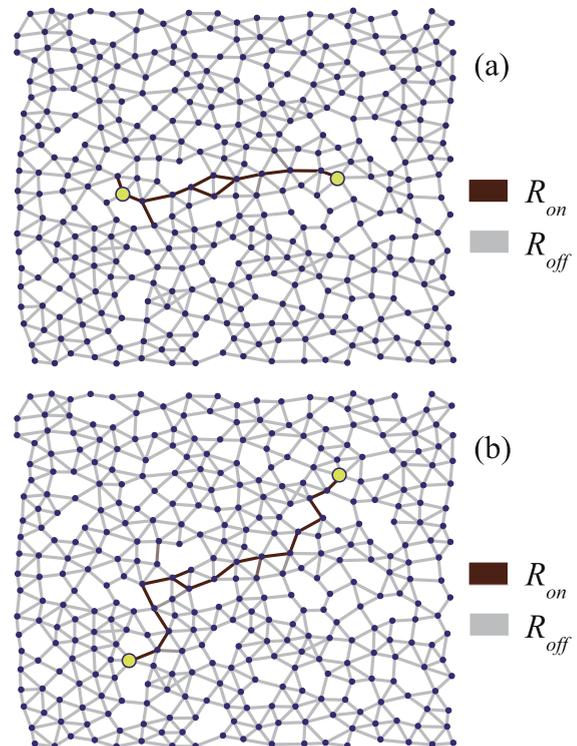
FIG. 7. (Color online) Solution of the shortest-path problem by a random $20a \times 20a$ network for a horizontal (a) and diagonal (b) choice of the specified nodes (large circles). These solutions are obtained for $R_{off}^M / R_{on}^M = 100$.
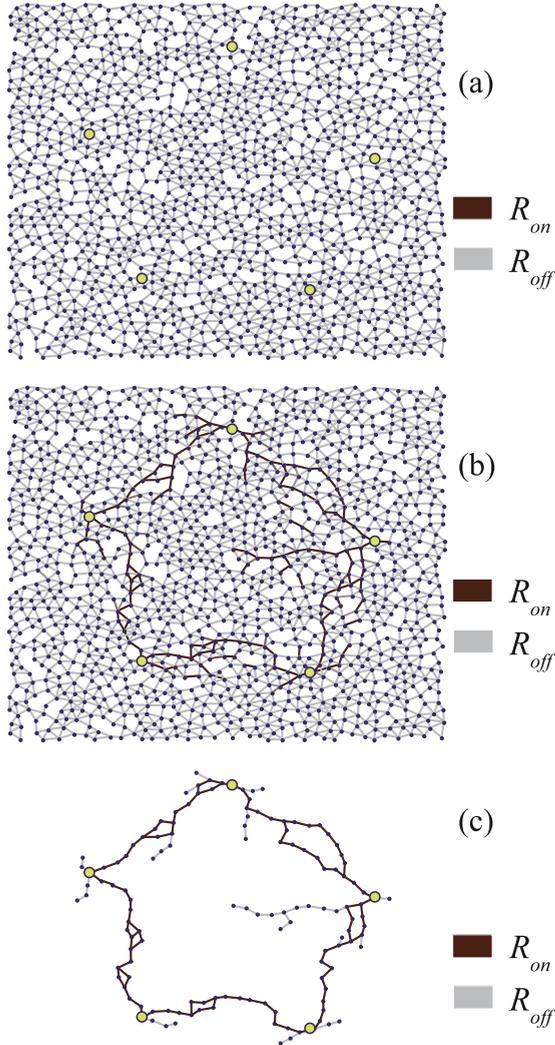
FIG. 8. (Color online) Solving the traveling salesman problem using a random $40a \times 40a$ network: (a) initial state of the network with cities denoted by large circles, (b) initial solution, (c) improved solution [dark red (dark gray) lines after post processing where the dead ends (light gray lines) have been eliminated. This solution is obtained for a network with $R_{off}^M/R_{on}^M = 100$. The final solution of the traveling salesman problem is represented by edges in R_{on} state in (c).

involved in the solution path as they offer less resistance to the current flow.

Clearly, a solution degeneracy or near degeneracy is also possible with random networks. For example, Fig. 7(a) demonstrates a solution splitting in its central part. In some cases, the solution near-degeneracy (for example, two parallel segments of different length) can be removed by a post-processing procedure in which the first problem solution (the set of edges in the R_{on} state) is used as a new network for the shortest-path problem that is subsequently solved following exactly the same algorithm as outlined in Sec. II B 1. Additionally, such post-processing procedure (that actually can be applied several times) removes all possible dead ends as well as segments disconnected from the solution path that occasionally are

switched into R_{on} state [an example of a dead end is the edge to above the left specified node in Fig. 7(a)].

B. Traveling salesman problem

Let us now consider the traveling salesman problem. For this purpose we use the algorithm described in Sec. II B 2, and we apply it twice. First of all, we apply this algorithm to the original network [Fig. 8(a)] and find the solution (that we call as the initial solution S1) presented in Fig. 8(b). This solution, however, contains some disconnected edges in the R_{on} states and dead ends (paths with single connection). In order to improve on the initial solution, we use a post-processing procedure consisting in the following. We form a reduced network consisting of all edges of S1 that are in the R_{on} state and nodes connected by these edges. Then, the traveling salesmen algorithm (Sec. II B 2) is applied to the reduced network. The total number of post-processing operations depends on the network complexity. Networks presented in this paper normally require one to two operations of this type. The final solution is represented in Fig. 8(c) by edges in R_{on} state [connected by dark red (dark gray) lines in the same figure]. This is the correct solution of the traveling salesman problem for the given configuration of cities.

Currently, however, it is difficult to identify strict mathematical conditions when the traveling salesman problem can be correctly solved by our approach and when it can not be solved. But, even in the cases when the complete solution can not be found, it is quite possible that the suggested algorithm will provide partial solutions and thus tremendously reduce the number of possibilities to verify. Understanding these limitations will be a subject of a future study.

V. CONCLUSION

In conclusion, we have studied the possibility to solve shortest-path optimization problems with memristive networks. Such approach is related to the concept of memcomputing: storing and processing of information on the same physical platform [5]. There are six main criteria that need to be satisfied in order to realize such a paradigm and it is clear that memristive networks considered in this paper satisfy all of them. Unlike other promising but more speculative proposals, like quantum computing, memcomputing is already a practical reality, at least in regard to some applications, such as digital logic [4,6,16,17]. It bypasses several of the bottlenecks of present-day computing architectures and its constitutive units—memristors, memcapacitors, and meminductors—are already widely available. Indeed, these elements emerge quite naturally with increasing miniaturization of electronic devices. The computational possibilities offered by this paradigm are varied, and due to its tantalizing similarities both with some features of the brain as well as with the collective properties of colonies of living organisms, it promises to open new directions in neuromorphic architectures and biological studies.

In this work we have suggested and studied the implementation of the shortest-path and traveling salesmen problem algorithms utilizing memristive networks. Intuitively, the shortest-path calculation can be understood as follows.

Typically, the shortest path has the lowest resistance. Therefore, memristive devices along the shortest path switch into their low memristance state faster than all other devices in the network (since the switching rate is proportional to the current). This process reinforces the current along the shortest path finally resulting in a chain of memristive devices in the low memristance state along the shortest path only.

Regarding the traveling salesman problem, the constraint of passing through all the cities is satisfied *a priori* in our algorithm as we apply multiple pulses between randomly selected pairs of cities. In this way, the position of each city is taken into account in the problem solution. This occurs, however, in a nontrivial way whose explanation is hidden in corresponding equations of circuit dynamics. However, we understand intuitively that the problem solution is found hierarchically. At shorter times, the low memristance paths are formed between closely spaced cities. At longer times, these clusters of cities become connected. As the electric current tends to flow through the shortest/least resistive path, it is expected that the proposed algorithm works out approximately the traveling salesman problem.

The main advantage of our approach is based on the analog parallel dynamics of many memristive devices. We have reported certain limitations of periodic networks that in some cases provide degenerate solutions. Random and quasiperiodic networks without any (or a few) symmetries are more promising in this regard. We have considered the network evolution that can be conveniently described by a network entropy. We anticipate that networks with memory have a wide range of applications beyond the shortest-path problems. In particular, such networks can be used to simulate problem solving abilities of the human brain, the effect and progress of mental illness, etc.

ACKNOWLEDGMENT

APPENDIX: SIMULATIONS

The numerical results presented in this paper have been obtained for a network of current-controlled bipolar memristive devices with threshold. Each memristive device is described by

$$V_M = R(x) I_M, \qquad (A1)$$

and

$$\frac{dx}{dt} = \begin{cases} 0 & \text{for} \quad |I_M| < I_t \qquad (A2) \\ \text{sgn}(I_M)\gamma(|I_M| - I_t) & \text{for} \quad |I_M| \geqslant I_t \qquad (A3) \end{cases}$$

where V_M and $I_M = \dot{q}(t)$ denote the (time-dependent) voltage and current across the device, respectively, $R(x) \equiv x$ is the memristance that changes between two limiting values R_{on}^M and R_{off}^M, x is the internal state variable, γ is a constant describing the rate of change of memristance when the magnitude of the electric current I_M exceeds the threshold current I_t; and sgn is the sign function. We note that a current-controlled threshold-type memristive device model was used to describe switching in bipolar memristive devices [26]. Moreover, many models of voltage-controlled memristive devices can be easily reformulated in the current-controlled form [3]. Equations (A1)–(A3) were directly used in our studies of the shortest-path problem. In the case of the traveling salesman problem, an absolute value of voltage was applied to each basic unit (edge) in order to avoid back-and-forth switchings when random polarity voltage is applied. Such an approach can be easily implemented in electronics.

All numerical results were obtained for the following model parameters: $R_{\text{off}}^M = 200\Omega$, $R_{ij}(t = 0) = R_{\text{off}}^M$, $\gamma = 10^6 \Omega/(\text{s } A)$, $I_t = 10$ mA. Figures 2, 3(a), 5, and 6 are obtained with $R_{\text{on}}^M = 10\Omega$ and $V = 6$V of applied voltage; Fig. 3(b) is found using $V = 6, 6.75, 10, 15.25$V for $R_{\text{off}}^M / R_{\text{on}}^M = 20, 10, 4, 1.25$ curves, respectively; Fig. 4 is plotted with $R_{\text{on}}^M = 160\Omega$ and $V = 15.25$V. Note that R_{on}^M and R_{off}^M are related to individual memristive devices, while R_{on} and R_{off} (used in figures) represent limiting values of memristance of the basic unit. While the OFF state of the basic unit is attained when both memristive devices are in their OFF states, the ON state of the basic unit corresponds to the ON, OFF combination of single device states. In our simulations, at each time step, the potential at all grid points is found as a solution of Kirchhoff's current law equations obtained using a sparse matrix technique. The corresponding change in the memristive states was computed using Eq. (A3). The width of the voltage pulse is selected sufficiently long to reach the steady state in each calculation.

[1] M. Di Ventra, Y. V. Pershin, and L. O. Chua, Proc. IEEE **97**, 1717 (2009).

[2] M. Di Ventra, Y. V. Pershin, and L. O. Chua, Proc. IEEE **97**, 1371 (2009).

[3] Y. V. Pershin and M. Di Ventra, Adv. Phys. **60**, 145 (2011).

[4] Y. V. Pershin and M. Di Ventra, Proc. IEEE **100**, 2071 (2012).

[5] M. Di Ventra and Y. V. Pershin, Nat. Phys. **9**, 200 (2013).

[6] J. Borghetti, G. S. Snider, P. J. Kuekes, J. J. Yang, D. R. Stewart, and R. S. Williams, Nature (London) **464**, 873 (2010).

[7] J. Von Neumann, IEEE Ann. Hist. Comput. **15**, 27 (1993).

[8] Y. V. Pershin and M. Di Ventra, Phys. Rev. E **84**, 046703 (2011).

[9] G. S. Snider, SciDAC Review **10**, 58 (2008).

[10] B. Linares-Barranco and T. Serrano-Gotarredona, *9th IEEE Conference on Nanotechnology, 26–30 July 2009, Geneva* (IEEE, Piscataway, 2009), pp. 601–604.

[11] A. Afifi, A. Ayatollahi, and F. Raissi, IEICE El. Express **6**, 148 (2009).

[12] Y. V. Pershin and M. Di Ventra, Neural Network **23**, 881 (2010).

[13] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, Nano Lett. **10**, 1297 (2010).

[14] I. Ebong and P. Mazumder, Proc. IEEE **100**, 2050 (2012).

[15] Z. Ye, S. H. M. Wu, and T. Prodromakis, arXiv:1303.3927.

[16] D. Strukov and K. Likharev, Nanotechn. **16**, 888 (2005).

[17] E. Lehtonen and M. Laiho, in *Proceedings of the 2009 International Symposium on Nanoscale Architectures (NANOARCH'09) 30–31 July 2009, San Francisco* (IEEE, Piscataway, 2009), pp. 33–36.

[18] M. Itoh and L. O. Chua, Int. J. Bif. Chaos **19**, 3605 (2009).

[19] L. O. Chua and S. M. Kang, Proc. IEEE **64**, 209 (1976).

[20] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook, *The Traveling Salesman Problem: A Computational Study* (Princeton University Press, Princeton, 2011).

[21] The memristance of a basic unit is that of two memristive devices connected in parallel. Note that in-parallel connection of two memristive devices can be described as a single higher-order memristive device.

[22] A. A. Colorni, M. M. Dorigo, and V. Maniezzo, in *Actes de la Premiere Conference Europienne sur la Vie Artificielle, Paris, France* (Elsevier Publishing, Amsterdam, 1991), pp. 134–142.

[23] N. Monmarche, F. Guinand, and P. Siarry, *Artificial Ants* (Wiley-ISTE, New York, 2010).

[24] Note that in the absence of capacitive components the total current is independent of the choice of this cross section at any given time, provided such cross section crosses the shortest-path solution and spans the far upper and lower ends of the network without self-intersecting.

[25] D. Levine and P. Steinhardt, Phys. Rev. Lett. **53**, 2477 (1984).

[26] M. D. Pickett, D. B. Strukov, J. L. Borghetti, J. J. Yang, G. S. Snider, D. R. Stewart, and R. S. Williams, J. Appl. Phys. **106**, 074508 (2009).