Solving mazes with memristors: a massively-parallel approach

Yuriy V. Pershin[1] & Massimiliano Di Ventra[2]

[1]Department of Physics and Astronomy and USC Nanocenter, University of South Carolina, Columbia, South Carolina 29208, USA.

[2]Department of Physics, University of California San Diego, La Jolla, California 92093-0319, USA.

Solving mazes is not just a fun pastime. Mazes are prototype models in graph theory, topology, robotics, traffic optimization, psychology, and in many other areas of science and technology [1-6]. However, when maze complexity increases their solution becomes cumbersome and very time consuming. Here, we show that a network of memristors - resistors with memory [7,8] - can solve such a non-trivial problem quite easily. In particular, maze solving by the network of memristors occurs in a massively parallel fashion since all memristors in the network participate simultaneously in the calculation. The result of the calculation is then recorded into the memristors' states, and can be used and/or recovered at a later time. Furthermore, the network of memristors finds all possible solutions in multiple-solution mazes, and sorts out the solution paths according to their length. Our results demonstrate not only the first application of memristive networks to the field of massively-parallel computing, but also a novel algorithm to solve mazes which could find applications in different research fields.

Mazes are a class of graphical puzzles in which, given an entrance point, one has to find the exit via an intricate succession of paths, with the majority leading to a dead end, and only one, or few, correctly "solving" the puzzle. Mazes - sometimes also called labyrinths - have been known since ancient times, the oldest presumably being the one created by Daedalus in Crete, as passed on by Greek mythology more than 4000 years ago. Algorithms to solve mazes vary from the simplest - and extremely slow - "random mouse" to mathematical search algorithms that operate

on a sequential fashion to find the exit. However, all these methods suffer from very slow solution times when the complexity of the maze increases, with solution times sometimes increasing dramatically with increasing local connectivity of the network. This has led some researchers to look for physical, chemical and biological systems to solve mazes [5,9]. For instance, in Ref. [5] a maze is solved by an amoeba that finds the minimum-length path between two points in a labyrinth connecting separate food sources. However, also these methods - while being interesting from a fundamental point of view - result in slow searches since they all require sequential computation for a successful solution of the maze.

Here, we suggest and demonstrate a new strategy for solving mazes that is instead based on massively-parallel computation as afforded by a network of memristors (short for memory resistors) [7]. Memristors are resistors whose resistance depends on the state history of the system, and can therefore record their past dynamics. These systems, which belong to the larger class of memory circuit elements – that includes also memcapacitors and meminductors [8] – are attracting considerable attention due to their usefulness in diverse research areas [10] ranging from memories [11-12] to neuromorphic computing and learning [13-15]. It is indeed worth noting here that we could perform maze solving also with a network of meminductors or memcapacitors. We choose to work with memristors since they are the most studied so far, and can be easily realized experimentally, thus allowing a practical implementation of our algorithm.

Let us then start by mapping a given maze into a network of memristors. This is shown in Figure 1. First of all, we superimpose a periodical array of vertical and horizontal lines on the maze. The period of this array corresponds to the intrinsic period of the maze (for non-periodic mazes, the period of line arrays should be selected in such a way to take into account all important maze features). The crossing points of vertical and horizontal lines define grid points of the memristor network. The network consists of basic units (memristors plus switches) connecting grid points. Since the direction of current flow is not known *a priori*, the polarity of adjacent memristors (indicated by the black thick line in the memristor symbol in Figure 1) is

chosen to be alternating. It is assumed that external signals can be applied to any grid points for the purpose of initializing the memristors' states as well as to read the calculation results. The externally-controlled switches are used to define the topology of the maze: a maze wall is modelled by a switch in the "not-connected" state.

The calculation performed by this memristor processor is initiated by application of a constant voltage V across the two grid points corresponding to the entrance and exit points of the maze as shown in Figure 1. In this case, the current flows only along those memristors that connect the entrance and exit points. The state of these memristors is changed by the current, thus the maze is solved in a *massively parallel* way, since all memristors in the network participate simultaneously in the calculation. Specifically, assuming that at the initial moment of time all memristors are in the high resistance ("OFF") state, as time passes, every other memristor along the solution path changes its resistance, eventually switching into the low-resistance ("ON") state. Therefore, the chain of memristors in the "ON" state connecting the entrance and exit points represents the maze's solution. Indeed, the state of the memristors connecting the entrance and exit points represents a solution at any given time after the initial one. If there are multiple paths, then the shortest one would contain less memristors and thus offers less resistance than the longest one, with all intermediate paths (in terms of length) offering a proportionate resistance. Therefore, since current flows in proportion to the resistance of a path, at any given time, the change of state of a given path is proportional to the current. The different paths of the maze can then be identified by the different state their memristors have at any given time.

Experimentally, the suggested network could be fabricated using, e.g., CMOL (Cmos+MOLecular-scale devices) architecture [16] combining a single memristor layer with a conventional CMOS (complementary metal oxide semiconductor) layer. Here instead, we have performed numerical simulations of the network dynamics. This approach can be easily implemented and offers a practical computational algorithm for maze solving. For this purpose,

we have used the model of an ideal memristor [17] whose memristance (memory resistance) is given by

$$R_{ij}^{M} = R_{ON} x_{ij} + R_{OFF}\left(1 - x_{ij}\right), \qquad (1)$$

where R_{ON} and R_{OFF} are minimal and maximal values of memristance, x_{ij} is the dimensionless internal state variable bound to the region $0 \le x_{i,j} \le 1$, and (i,j) are grid indexes of a memristor to identify its location in the network. For simplicity, we choose the dynamics of $x_{i,j}$ to be given by

$$\mathrm{d}x_{i,j} / \mathrm{d}t = \alpha\, I_{i,j}(t), \qquad (2)$$

where α is a constant and $I_{i,j}(t)$ is the current flowing through the memristor (i,j). 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 memristors' states was computed using Eq. (2).

All numerical results reported in this Report were obtained using model parameters $R_{ON} = 10\,\mathrm{Ohms}$, $R_{OFF} = 100$ Ohms, $R_{ij}^{M}(t=0) = 91\,\mathrm{Ohms}$, $x(t=0) = 0.1$ and $\alpha = 10^{4}\left(\mathrm{s}\cdot\mathrm{A}\right)^{-1}$ for all memristors. The applied voltage was selected to be equal to 50V during the first 0.1 s time interval, and -50V at $t > 0.1\,\mathrm{s}$. The sign of the applied voltage was changed in order to better represent the maze solution as discussed below. Two types of mazes were considered: a single-path maze, and a multiple- (two-) path maze.

Figure 2 presents results of solution of a single-path maze (see also Supplementary Movie 1). We have found that with these parameters it takes approximately 0.06 s to switch every other memristor along the solution path into the low-resistance "ON" state. The resulting sequence of memristors in the low-resistance state represents the maze solution as shown in Figure 2a. This maze solution can be better seen if we change the sign of the applied voltage and wait some time. Then, the resistance of memristors - along the solution path - that are in the "ON" state will increase, and the resistance of memristors in the "OFF" state will decrease. Figure 2b captures a specific moment of time when these memristances are equal. At this moment of time, every

memristor along the solution path is in the same intermediate state and thus the solution of the maze is better visible.

A two-path maze whose solution is given in Figure 3 was obtained from the maze shown in Figure 2 by removing a single segment of the wall. In this maze, the current flowing through a common segment (red dots in Figure 3) splits between two possible paths according to their resistances. Therefore, memristors in the common segment change their resistance faster than those in the two possible paths (see Figure 3 and also Supplementary Movie 2). Moreover, comparing memristors from two possible paths, memristors in the shorter path change their resistance faster than those in the longer path. As discussed previously, this allows sorting all possible solutions according to their length in such a way that the resistances of the memristors along shorter paths are smaller.

In conclusion we have shown how a network of memristors - a memristor processor - can solve mazes in a massively-parallel way. This approach can be realized experimentally with available systems and devices, or simply implemented on a computer. The resulting algorithm is superior to any existing maze solving methods and therefore it is ideal when the complexity of the maze increases with increasing local connectivity of the graph. We thus envision its use in a large set of applications in both basic science and technology.

References

1. Pellow, S., Chopin, P., File, S. E. & Briley M. Validation of open-closed arm entries in an elevated plus-maze as a measure of anxiety in the rat. *J. Neurosci. Methods* **114**, 149-167 (1985).

2. Reiners, P. D. "Robots, Mazes, and Subsumption Architecture," in *IBM Developer Works*, (2007). Available at: http://www.ibm.com/developerworks/java/library/j-robots

3. Modesti, P. & Sciomachen, A. A utility measure for finding multiobjective shortest paths in urban multimodal transportation networks. *Eur. J. Oper. Res.* **111**, 495-508 (1998).

4. Teroa, A., Kobayashia, R. & Nakagaki, T. Physarum solver: A biologically inspired method of road-network navigation. *Physica A* **363**, 115-119 (2006).

5. Nakagaki, T., Yamada, H., & Toth, A. Intelligence: Maze-solving by an amoeboid organism. *Nature* **407**, 470 (200).

6. Crowe, D.A., Averbeck, B.B., Chafee, M.V., Anderson, J.H. & Georgopoulos, A.P. Mental Maze Solving. *J Cogn. Neurosci.* **12**, 813-827 (2000).

7. Chua, L. O. Memristor - the missing circuit element. *IEEE Trans. Circuit Theory* **18**, 507–519 (1971).

8. Di Ventra, M., Pershin, Y.V. & Chua, L.O. Circuit elements with memory: memristors, memcapacitors and meminductors Proc. IEEE **97**, 1717 (2009).

9. Lagzi,I., Soh, S., Wesson, P. J., Browne, K. P., & Grzybowski, B. A. Maze Solving by Chemotactic Droplets. J. Am. Chem. Soc. **132**, 1198-1199 (2010).

10. Pershin, Y. V. & Di Ventra, M. Memory effects in complex materials and nanoscale systems. *Advances in Physics* (in press). Preprint at http://arxiv.org/abs/1011.3053 (2010).

11. Burr, G.W., Kurdi, B.N., Scott, J.C., Lam, C.H., Gopalakrishnan, K. & Shenoy, R.S. Overview of candidate device technologies for storage-class memory, *IBM J. Res. Dev.* **52**, 449–464 (2008).

12. Dietrich, S., Angerbauer, M., Ivanov, M., Gogl, D., Hoenigschmid, H., Kund, M., Liaw, C. & Markert, M. A Nonvolatile 2-Mbit CBRAM Memory Core Featuring Advanced Read and Program Control, *IEEE J. Sol.-State Circ.* **42**, 839 (2007).

13. Pershin, Y. V., La Fontaine, S. & Di Ventra, M. Memristive model of amoeba's learning. *Phys. Rev. E* **80**, 021926 (2009).

14. Pershin, Y. V. & Di Ventra, M. Experimental demonstration of associative memory with memristive neural networks. *Neural Networks* **23**, 881 (2010).

15. Jo, S. H., Chang, T., Ebong, I., Bhadviya, B. B., Mazumder, P. & Lu, W. Nanoscale Memristor Device as Synapse in Neuromorphic Systems. *Nano Lett.* 10, 1297–1301 (2010).

16. Likharev, K.K. & Strukov, D.B. CMOL: Devices, Circuits, and Architectures, in Introducing Molecular Electronics, G.F. G. Cuniberti and K. Richter, eds., Springer, 2005, pp. 447–477.

17. Strukov, D. B., Snider, G. S., Stewart, D. R. & Williams, R. S. The missing memristor found. *Nature* **453**, 80-83 (2008).

Supplementary Information

Supplementary Movie 1

This movie shows dynamics of maze solving for a single-path maze. The time is in seconds, resistance in Ohms, voltage in Volts, and current in Amperes. The red line links network grid points connected to memristors whose $R_{ij}^{M} < 80$ Ohms and not separated by maze walls.

Supplementary Movie 2

This movie shows dynamics of maze solving for a multiple-path maze. The time is in seconds, resistance in Ohms, voltage in Volts, and current in Amperes. The red line links network grid points connected to memristors whose $R_{ij}^{M} < 80$ Ohms and not separated by maze walls.

The movies can be found at http://www.physics.sc.edu/~pershin/memdevices.htm.

Acknowledgements

M.D. acknowledges partial support from the National Science Foundation.

Author Information

Correspondence and requests for materials should be addressed to Y.V.P. (pershin@physics.sc.edu) or M.D. (diventra@physics.ucsd.edu).

Figure Captions

Figure 1 | Maze mapping into a network of memristors. Right panel. The maze is mapped into an array of vertical and horizontal lines having the periodicity of the maze. **Left panel.** Architecture of the network of memristors in which each crossing between vertical and horizontal lines in the array is represented by a grid point to which several basic units consisting of memristors and switches (field-effect transistors) are linked in series. The maze topology is encoded into the state of the switches such that if the short line segment connecting neighboring crossing points in the array crosses the maze wall then the state of the corresponding switch is "not connected" (shown in red). All other switches are in the "connected" state. The external voltage (V) is applied across the connection points corresponding to the entrance (V) and exit (ground, GND) points of the maze.

Figure 2 | Solution of a single-path maze. a, Network state at $t = 0.075$ s. The chain of memristors in the low-memristance state (shown by red dots) clearly connects the entrance and exit points of the maze (note that memristors in the "OFF" state - when $R_{ij}^{M}(t=0) > 90$ Ohms - are not shown). Here, every other memristor along the solution path is in the low-memristance state. **b,** Network state at $t = 0.12$ s. Note that at $t = 0.1$ s the sign of the applied voltage has been changed. At $t = 0.12$ s, each memristor along the solution path shows the maze solution. The resistance is in Ohms, the voltage in Volts, and the current in Amperes.

Figure 3 | Solution of a multiple-path maze. Network state at $t = 0.047$ s. The maze solution contains two common segments (red dots), and two alternative segments of different lengths close to the left bottom corner (blue and green dots). The memristance in the shorter segment (blue dots) is smaller than that in the longer segment (green dots) since the current through the shorter segment is larger and, consequently, the change of the memristors' state along this segment is larger. The resistance is in Ohms, the voltage in Volts and the current in Amperes.

Figure 1

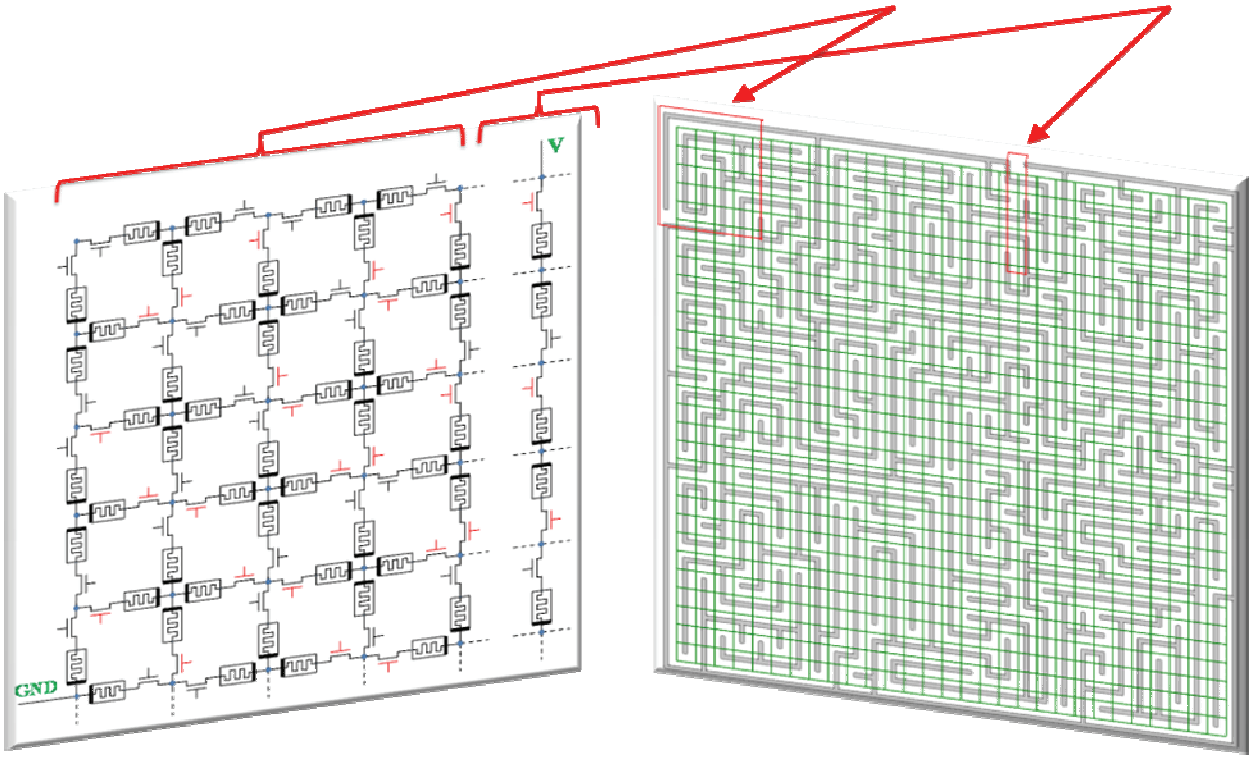
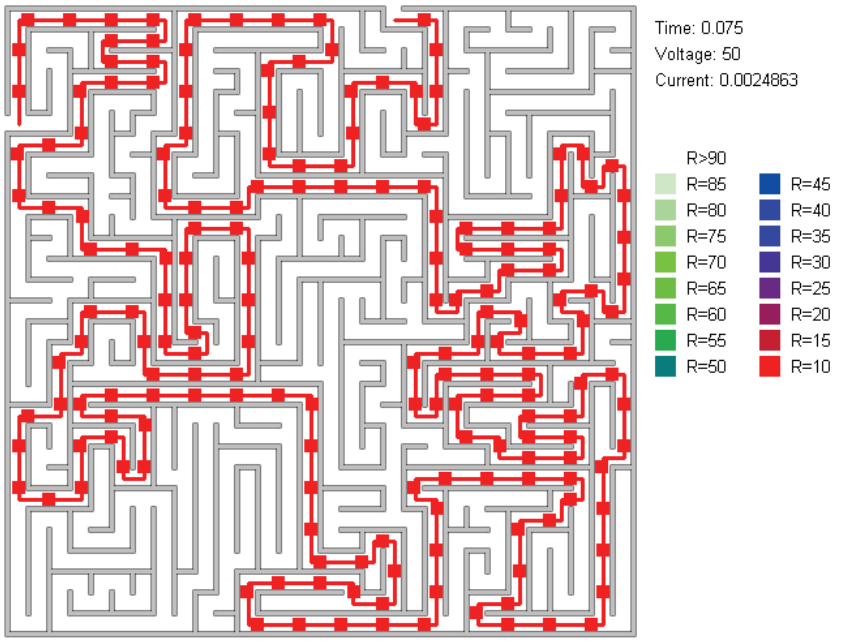
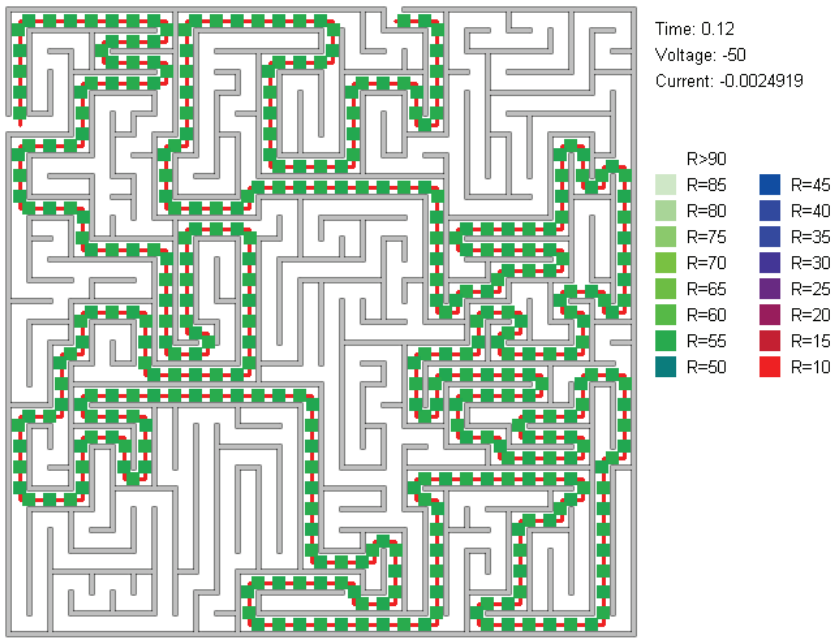


Figure 2



a.



b.

Figure 3