

# Cellular Automata for Topology Control in Wireless Sensor Networks Using Matlab

Stavros Athanassopoulos, Christos Kaklamanis,  
Gerasimos Kalfountzos and Evi Papaioannou

**Abstract** We use cellular automata for simulating topology control algorithms in wireless sensor networks (WSNs) using the Matlab programming environment. Our objective has been to provide experimental evidence in order to (i) investigate whether potential researchers who lack sound programming skills but would like to promptly implement their ideas with confidence should use cellular automata rules and Matlab in order to experimentally evaluate their works and (ii) evaluate how the selection of a Moore or a Margolus neighborhood in cellular automata models can affect the performance of simulations of topology control algorithms in WSN.

**Keywords** Cellular automata · Simulation · Matlab · Topology control · WSN

---

This work has been partially supported by EU under the ICT-2010-258307 project EULER and by EU and the Greek Ministry of Education, Lifelong Learning and Religious Affairs under the project Digital School (296441).

---

S. Athanassopoulos · C. Kaklamanis · E. Papaioannou (✉)  
Computer Technology Institute & Press “Diophantus”,  
Patras University Campus, 26504 Rion, Greece  
e-mail: papaioan@ceid.upatras.gr

S. Athanassopoulos  
e-mail: athanaso@ceid.upatras.gr

C. Kaklamanis  
e-mail: kakl@ceid.upatras.gr

S. Athanassopoulos · C. Kaklamanis ·  
G. Kalfountzos · E. Papaioannou  
Department of Computer Engineering and Informatics,  
Patras University Campus, 26504 Rion, Greece  
e-mail: kalfount@ceid.upatras.gr

## 1 Introduction

A cellular automaton (CA) [9] is an idealization of a physical system in which space and time are discrete and the physical quantities take only a finite set of values. Informally, a cellular automaton is a lattice of cells, each of which may be in a predetermined number of discrete states. A neighborhood relation is defined over this lattice, indicating for each cell which cells are considered to be its neighbors during state updates. In each time step, every cell updates its state using a transition rule that takes as input the states of all cells in its neighborhood (which usually includes the cell itself). All cells in the cellular automaton are synchronously updated. At time  $t = 0$  the initial state of the cellular automaton must be defined; then, repeated synchronous application of the transition function to all cells in the lattice will lead to the deterministic evolution of the cellular automaton over time. Cellular automata have received extensive academic study into their fundamental characteristics and capabilities and have been applied successfully to the modelling of natural phenomena and complex systems (e.g., [8, 14]). Based on the theoretical concept of universality, researchers have tried to develop simpler and more practical architectures of CA that can be used to model widely divergent application areas, including theoretical biology [2], game theory [10], etc. Cellular automata have successfully been used as a means for modelling and simulation of topology control algorithms in Wireless Sensor Networks (WSN) (e.g., [3, 6, 11, 15]).

WSN are composed of a large number of autonomous sensor nodes geographically scattered on a surface with the ability to monitor an area inside their range and collect data about physical and environmental conditions. WSN have been used in a wide range of applications. The most important performance aspect in WSNs is the need to be energy efficient as sensor nodes have a finite energy reserve offered by a battery. Topology control is a technique used to reduce the initial topology of a WSN in order to save energy, avoid interference and extend the lifetime of the network by discovering a minimum configuration of nodes capable of monitoring a region equivalent to the monitored one for all nodes. Efficient topology control techniques in WSN are very critical and essential: sensors operate on limited energy (i.e., batteries). Evaluation of topology control algorithms requires simulation since setting up a real WSN is very costly. There is a long literature, both theoretical and experimental, on topology control algorithms for WSN [1, 4, 5, 7]. In this work, we focus on a subset of topology control algorithms (duty cycling and scheduling while maintaining connectivity and coverage) and use the cellular automata simulation approach suggested in [3] in order to experimentally investigate which type of neighborhood should be preferred for obtaining efficient simulations for topology control algorithms in WSN.

Existing implementations of cellular automata have been developed using Java and C/C++ or C-based special-purpose simulating tools like COOJA, OMNeT++, CAsim tool, etc. These require advanced programming skills on behalf of the user/developer. Matlab, although primarily intended for numerical computing, has been widely used in academia/education and industry by users coming from various

backgrounds of engineering, science and economics. In our work, we use Matlab for implementing corresponding cellular automata. From our experience, we encourage potential researchers who lack sound programming skills but would like to promptly implement their ideas with confidence to use cellular automata rules and Matlab in order to experimentally evaluate their works.

In [Sect. 2](#) we present in detail algorithms and corresponding cellular automata as well as simulation results. We conclude in [Sect. 3](#).

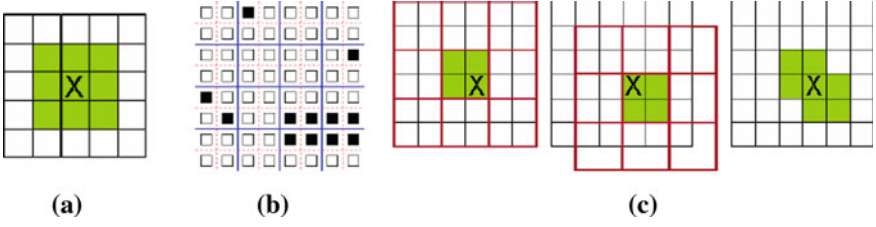
## 2 Topology Control Algorithms: Description and Experimental Results

We have used cellular automata and for simulating and evaluating three topology control algorithms in WSN, exploiting the properties of Moore and Margolus neighborhoods in the corresponding cellular automata. In this section, we first provide an implementation in Matlab of algorithm Topology Control (TC) [3]. For simulating algorithm TC, we have used a cellular automaton with a *Moore neighborhood*. A Moore neighborhood [2] is composed of the eight cells surrounding a central cell on a two-dimensional square lattice including the cell itself (Fig. 1a), i.e., defining an area of nine cells in total.

Then, we present two new topology control algorithms, Margolus neighborhood—Topology Control (MTC) and Weighted Margolus neighborhood—Topology Control (WMTC). These are variations of algorithm TC resulting from the use of a *Margolus neighborhood* in the cellular automata used for simulation. Informally, when Margolus neighborhood [12, 13] is used, the lattice is divided into disjoint blocks of size  $2 \times 2$ ; each block moves down and to the right with the next generation, and then moves back (Fig. 1b, c). This means that the neighborhood of a cell is defined as an area of 4 cells in each time step; in two consecutive time steps (i.e., odd-even successive time steps), the Margolus neighborhood of cell is composed of eight cells in total.

The type of neighborhood adopted can affect the performance of the simulation since it can impose limitations on the number of the active sensors used to cover an area. In a Moore neighborhood, every sensor has one unique neighborhood of 9 nodes (Fig. 1a). In a Margolus neighborhood, every sensor has two different neighborhoods composed of 4 nodes forming a square of size  $2 \times 2$  depending on odd/even time steps. In Fig. 1b, an instance of Margolus neighborhood is shown: dotted lines form the neighborhoods created in the odd time steps and regular lines show the neighborhoods formed in the even time steps. So, in a Margolus neighborhood, every sensor can monitor a total area of seven nodes in two consecutive time steps (Fig. 1c).

Simulations have been developed in Matlab Version 7.0.0.19920 (R14) and executed on an Intel Core i3 530 processor at 2.93 GHz with 6144 MBytes DDR3 RAM running Windows 7 operating system. We have evaluated our algorithms using metrics commonly used in WSNs: (i) number of the active sensors in the



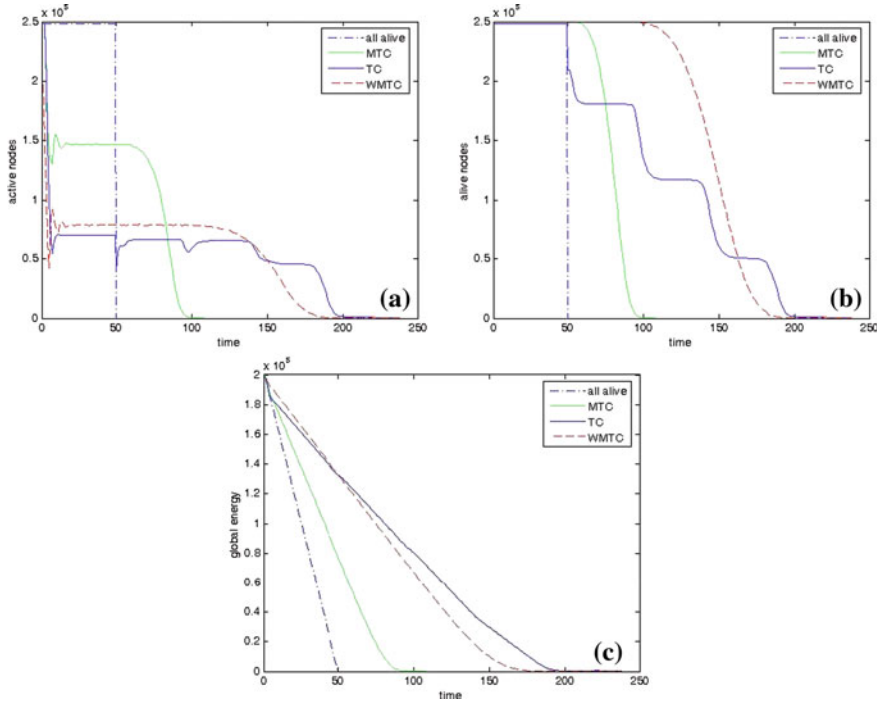
**Fig. 1** Moore neighborhood used in the CA for algorithm TC (a), Margolus Neighborhood used in the CA for algorithms MTC and WMTC (b, c)

network at each time step: increasing the number of active sensors improves network performance, (ii) number of the alive sensors: alive sensors include both the active and the stand-by nodes; their number reflects the global energy of the WSN, i.e., the sum of the remaining energy of the batteries of all the alive nodes: if the global energy is slowly reduced, the network lifetime is extended and (iii) coverage and connectivity: coverage reflects the percentage of active sensors in the network and its value shows the degree to which the network is covered by active nodes; connectivity reflects the ability of network nodes to communicate and increases as the number of active nodes there in a neighborhood increases.

## 2.1 CA with a Moore Neighborhood: Algorithm TC

In a WSN, it is usual that an area is covered by redundant sensors due to their random deployment. When many redundant sensors remain active simultaneously, the global energy of the network is rapidly reduced and the network lifetime is being shortened. The objective of algorithm TC [3] is to minimize the number of redundant active sensors covering a particular area by deactivating a subset of them for a period of time. Deactivation of a node turns it to stand-by mode, i.e., although the sensor has not run out of battery, it does not monitor its area probably because there is another neighboring sensor covering it. Having network nodes in stand-by mode periodically implies less energy consumption, thus, longer network lifetime. The main idea of algorithm TC can be outlined as follows: network nodes form an  $n \times n$  grid. Initially, all network nodes are active. There is a timer assigned to each node, which randomly receives integer values in [1, 5] and decreases by one in each time step. When the value of the timer of a node becomes zero, the node checks its neighborhood (which includes the node itself) for active nodes. If there are at least two active nodes, the node is/remains deactivated (i.e., in stand-by mode); otherwise, the node becomes/remains active. The node re-initializes its timer and repeats the same procedure (until it runs out of energy).

The cellular automaton we used for simulation uses a  $(n \times n)$  grid cell-space and a Moore neighborhood (Fig. 1a). Every cell  $c_{i,j}$  corresponds to a network node



**Fig. 2** Active (a) and alive (b) nodes and global energy (c) for algorithms TC, MTC and WMTC

and contains information about its position in the network [determined by its coordinates  $(i, j)$ ], its remaining energy, its state  $Sc_{i,j} \in \{0, 1\}$  and a timer  $Tc_{i,j}$ . Cells can be in one of the following two states: a cell  $c_{i,j}$  is in state 1 ( $Sc_{i,j} = 1$ ) when the corresponding network node contains an *active* sensor;  $c_{i,j}$  is in state 0 ( $Sc_{i,j} = 0$ ), when the corresponding network node contains a *stand-by* sensor. Initially, cells are in state 1 (i.e., active). The timer  $Tc_{i,j}$  is initialized with an integer value in  $[1, 5]$  and decreases by one in each step. When  $Tc_{i,j} = 0$ , the state of cell  $c_{i,j}$  is updated according to the following rule and  $Tc_{i,j}$  is re-initialized:

$$S_{c_{i,j}}(t+1) = \begin{cases} 1, & \text{if } \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} S_{c_{k,l}}(t) \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

**Simulation results.** Figure 2 shows active (a) and alive (b) sensors and the global energy (c) of a WSN when algorithm TC is applied. The network lifetime is significantly extended (by a factor of 4) with respect to the case where no topology control algorithm is used. This is reasonable and expected: the number of active sensors decreases since several sensors remain in stand-by mode; the energy savings due to stand-by sensors extend the network lifetime.

Figure 3a shows coverage and connectivity when algorithm TC is applied. Algorithm TC obtains almost 100 % coverage in every time step of the network

lifetime. This is because algorithm TC periodically deactivates redundant sensors which do not increase the network coverage rate, since the area is already covered by some other sensor. On the other hand (Fig. 3b), when algorithm TC is used, connectivity is too low, since only active nodes can increase connectivity.

These observations match the experimental evaluation obtained for algorithm TC using the CAsim tool [3], a special purpose, ANSI C-based simulator for cellular automata models.

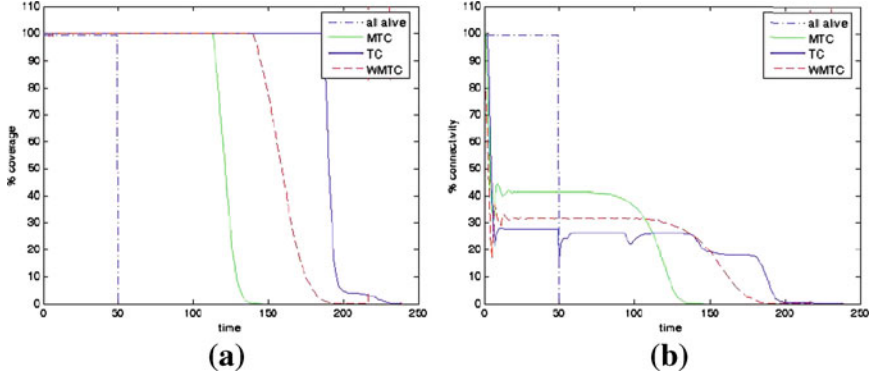
## 2.2 CA with a Margolus Neighborhood: Algorithms MTC and WMTC

Next, we present two alternative topology control algorithms, namely MTC and WMTC. Both algorithms work on a  $n \times n$  grid WSN. Their main idea follows similar line to these of algorithm TC and can be outlined as follows: initially, all network nodes are active. There is a timer assigned to each node, which randomly receives integer values in [1, 5] and decreases by one in each time step. When the value of the timer of a node value becomes zero, the node checks its current neighborhood (which includes the node itself) for active nodes. If there is at least one active node, the node is/remains deactivated (i.e., in stand-by mode); otherwise, the node becomes/remains active. The node re-initializes its timer and repeats the same procedure (until it runs out of energy). This implies that in each time step, in each four-node neighborhood, there is at least one active node.

The difference between these two algorithms and TC lies in the type of neighborhood used in the corresponding cellular automata models: now, a Margolus neighborhood (Fig. 1c) replaces the Moore neighborhood used in the cellular automaton for algorithm TC. Our cellular automata use a  $(n \times n)$  grid cell space.

Every cell  $c_{i,j}$  of the cellular automaton corresponds to a network node and contains information about its position in the network [determined by its coordinates  $(i, j)$ ], its remaining energy, its state  $Sc_{i,j} \in \{0, 1\}$  and a timer  $Tc_{i,j}$ . Cells can be in one of the following two states: a cell  $c_{i,j}$  is in state 1 ( $Sc_{i,j} = 1$ ) when the corresponding network node contains an *active* sensor;  $c_{i,j}$  is in state 0 ( $Sc_{i,j} = 0$ ), when the corresponding network node contains a *stand-by* sensor. Initially, cells are in state 1 (i.e., active). The timer  $Tc_{i,j}$  is initialized with an integer value in [1, 5] and decreases by one in each step.

The difference between MTC and WMTC lies on whether cells states are updated using information of the neighborhood in previous steps. In particular, according to algorithm MTC, each cell adopts a static view of its Margolus neighborhood, i.e., every cell takes into account only its current neighborhood for implementing the transition function induced by MTC. When algorithm WMTC is used, a dynamic approach to the Margolus neighborhood of each cell is adopted: every cell, in order to implement the transition function induced by WMTC, takes into account not only its current neighborhood but also its Margolus neighborhood during the previous step. This means that when  $Tc_{i,j} = 0$ , each cell  $c_{i,j}$  updates its



**Fig. 3** Coverage (a) and connectivity (b) for algorithms TC, MTC and WMTC

state according to the following rules, for MTC and WMTC, respectively, and re-initializes  $Tc_{i,j}$ :

**Algorithm MTC**

$$S_{c_{i,j}}(t+1) = \begin{cases} 1, & \text{if } \sum_{k=i-1}^i \sum_{l=j-1}^j S_{c_{k,l}}(t) \leq 1, \quad \text{for even } t \\ 1, & \text{if } \sum_{k=i}^{i+1} \sum_{l=j}^{j+1} S_{c_{k,l}}(t) \leq 1, \quad \text{for odd } t \\ 0, & \text{otherwise} \end{cases}$$

**Algorithm WMTC**

$$S_{c_{i,j}}(t+1) = \begin{cases} 0, & \text{if } W_{c_{i,j}}(t) \geq 1 \text{ and } W_{c_{i,j}}(t-1) \geq 1 \\ 1, & \text{otherwise} \end{cases}$$

Notice that the dynamic view of Margolus neighborhood of cells in algorithm WMTC has been implemented (see CA rules for algorithm WMTC) via the assignment of two weight values to each cell  $c_{i,j}$  ( $W_{c_{i,j}}(t)$  equals  $\sum_{k=i-1}^i \sum_{l=j-1}^j S_{c_{k,l}}(t)$

when  $t$  is even and  $\sum_{k=i}^{i+1} \sum_{l=j}^{j+1} S_{c_{k,l}}(t)$  when  $t$  is odd) which actually count the number of active cells in the Margolus neighborhood of each cell in successive time steps:

**Simulation results.** Simulation results are presented in Figs. 2 and 3.

Active and alive nodes, total network energy, coverage and connectivity of algorithms TC, MTC and WMTC are presented in comparison with the case where no topology control algorithm is used. Performance evaluation for TC, MTC and WMTC is based on the network lifetime they obtain as well as on the coverage and connectivity rates they achieve. According to our simulations, algorithm TC, which has been simulated by a cellular automaton using a Moore neighborhood, clearly outperforms algorithms MTC and WMTC, which have been simulated by cellular

automata using a Margolus neighborhood. So, bad news is that, with respect to a Moore neighborhood, the use of a Margolus neighborhood fails to offer more efficient cellular automata for the modelling and simulation of algorithms for topology control in WSN. However, good news is that using a Margolus neighborhood in cellular automata for topology control still results in a “good” performance: it is worth-mentioning that algorithm WMTC obtains a performance close to that of algorithm TC; in particular, in terms of network lifetime, algorithm WMTC reaches a 91.6 % of the performance of algorithm TC (which corresponds to an improvement of 4.4 versus 4.8 % of algorithm TC, against the case where no algorithm is used for topology control). Especially, in terms of connectivity, algorithm WMTC outperforms algorithm TC since it achieves an average connectivity of 25 % versus a corresponding value of 20 % achieved by algorithm TC.

### 3 Conclusions

In this paper, we have used cellular automata for simulating and evaluating three topology control algorithms in WSN using Matlab. The motivating question has been how the selection of a Moore or a Margolus neighborhood in cellular automata models can affect the performance of simulation of topology control algorithms in WSN. Our results indicate that a cellular automaton with a Moore neighborhood achieves improved performance. It is interesting to investigate how the use of a Margolus neighborhood can actually exploit additional information about cells and result in more efficient algorithms for topology control in WSN. Regarding the use of Matlab, we believe that potential researchers who lack sound programming skills but would like to promptly implement their ideas with confidence should be encouraged to use cellular automata rules and Matlab in order to experimentally evaluate their works.

### References

1. Burkhart, M., von Rickenbach, P., Wattenhofer, R., Zollinger, A.: Does topology control reduce interference? In: *Proceedings of MobiHoc 2004*, ACM, pp. 9–19 (2004)
2. Chopard, B., Droz, M.: *Cellular automata modeling of physical systems*. Cambridge University Press, Cambridge (1998). ISBN 0-521-46168-5
3. Cunha, R.O., Silva, A.P., Loureiro, A.A.F., Ruiz, L.B.: *Simulating large wireless sensor networks using cellular automata*. Federal University of Minas Gerais, Department of Computer Science, Department of Electrical Engineering (2005)
4. Johansson, T., Carr-Motyckova, L.: Reducing interference in ad hoc networks through topology control. In: *Proceedings of DIALM-POMC 2005*, ACM, pp. 17–23 (2005)
5. Labrador, M.A., Wightman, P.M.: *Topology control in wireless sensor networks: with a companion simulation tool for teaching and research*. Springer, Heidelberg (2009). ISBN: 978-1-4020-9584-9 pbk
6. Li, W., Zomaya, A.Y., Al-Jumaily, A.: Cellular automata based models of wireless sensor networks. In: *Proceedings of MobiWAC 2009*, ACM, pp. 1–6 (2009)



7. Lou, T., Tan, H., Wang, Y., Lau, F.C.M.: Minimizing average interference through topology control. In: *Proceedings of ALGOSENSORS 2011*, LNCS 7111, Springer, Berlin, pp. 115–129 (2012)
8. Mitchell, M., Hrabér, P.T., Crutchfield, J.P.: Revisiting the edge of chaos: evolving cellular automata to perform computations. *Complex Syst.* **7**, 89–130 (1993)
9. Neumann, J.V.: *Theory of self-reproducing automata*. In: Burks, A.W. (ed.). University of Illinois Press, Urbana, IL (1966). 403 pages, ISBN 0598377980 pbk
10. Nowak, M., May, R.: Evolutionary games and spatial chaos. *Nature* **359**(6398), 826–829 (1992)
11. Qela, B., Wainer, G., Mouftah, H.: Simulation of large wireless sensor networks using Cell-Devs. In: *Proceedings of the 2009 Winter Simulation Conference*, IEEE, pp. 3189–3200 (2009)
12. Schiff, J.: 4.2.1 Partitioning cellular automata. *cellular automata: a discrete view of the world*, pp. 115–116. Wiley-Interscience, UK (2008). 280 pages, ISBN 978-0-470-16879-0 pbk
13. Toffoli, T., Margolus, N.: II.12 The Margolus neighborhood. *cellular automata machines: a new environment for modeling*, pp. 119–138. MIT Press, MA (1987). 259 pages, ISBN 0-262-20060-0 pbk
14. Wolfram, S.: *Theory and Applications of Cellular Automata*. World Scientific, Singapore (1986). ISBN 9971-50-124-4 pbk
15. Zhang, W., Zhang, L., Yuan, J., Yu, X., Shan, X.: Demonstration of non-cluster based topology control method for wireless sensor networks. In: *Proceedings of CCNC 2009*, IEEE, pp. 1–2 (2009)

Future Information Technology, Application, and Service

FutureTech 2012 Volume 1

Park, J.J.H.; Leung, V.C.; Wang, C.-L.; Shon, T. (Eds.)

2012, LVIII, 770 p., Hardcover

ISBN: 978-94-007-4515-5