Wireless Pers Commun
DOI 10.1007/s11277-014-1626-1

An Energy Efficient Barrier Coverage Algorithm for Wireless Sensor Networks

Habib Mostafaei · Mohammad Reza Meybodi

Abstract Intrusion detection is one of the most important applications of wireless sensor networks. When mobile objects are entering into the boundary of a sensor field or are moving cross the sensor field, they should be detected by the scattered sensor nodes before they pierce through the field of sensor (barrier coverage). In this paper, we propose an energy efficient scheduling method based on learning automata, in which each node is equipped with a learning automaton, which helps the node to select best node to guarantee barrier coverage, at any given time. To apply our method, we used coverage graph of deployed networks and learning automata of each node operates based on nodes that located in adjacency of current node. Our algorithm tries to select minimum number of required nodes to monitor barriers in deployed network. To investigate the efficiency of the proposed barrier coverage algorithm several computer simulation experiments are conducted. Numerical results show the superiority of the proposed method over the existing methods in term of the network lifetime and our proposed algorithm can operate very close to optimal method.

Keywords Barrier coverage · Sensor networks · Energy efficient scheduling · Learning automata (LA)

1 Introduction

Wireless sensor networks have many applications in different situations, for example; national security, surveillance, health care and environmental monitoring. Sensor nodes are small devices that can sense environment, process monitored data, save sensed data, and send monitored data to a base station in the network [1].

H. Mostafaei (✉)
Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, Iran
e-mail: h.mostafaei@iaurmia.ac.ir

M. R. Meybodi
Department of Computer Engineering, AmirKabir University of Technology, Tehran, Iran
e-mail: mmeybodi@aut.ac.ir

In wireless sensor networks (WSNs), one of the most important design challenges is to increase network lifetime. In this case, energy efficiency is a critical issue in WSNs applications when battery change is not applicable. In recent years, most research has been done on the energy use of battery to prolong network lifetime. One of the common methods to improve lifetime is node activity scheduling and in some case it is called energy efficient scheduling.

Movement detection is one of the several important applications of wireless sensors such as when they deployed along international borders to detect illegal intrusion, around forests to detect the spread of forest, around a chemical factory to detect the spread of lethal chemicals, on both sides of a gas pipeline to detect potential sabotage, etc. In wireless sensor networks, barrier coverage guarantees to detect every movement crossing a barrier of sensors and it is known to be an appropriate model of coverage for such applications [2]. This model of coverage has several advantages over the full coverage model that we require to monitor all points in network field. First, barrier coverage needs much fewer sensor nodes than full coverage. If the width of the deployment region is three times the sensing range, full coverage requires more than twice the density of barrier coverage. Second, the sleep-wakeup problem, that determines a sleeping schedule for sensors to maximize the network lifetime, is polynomial-time solvable for barrier coverage even when sensor lifetimes are not equal [3]. For the full coverage model, on the other hand, the sleep-wakeup problem is NP-Hard even if sensor lifetimes are assumed to be identical [4]. Several researches have done in the field of full coverage to maximize network lifetime and in this paper we just focus on increasing life time of barrier coverage problem in wireless sensor networks.

Essentially, barrier coverage can be categorized into two classification: weak barrier coverage and strong barrier coverage [5]. In weak barrier coverage, we only need to detect intruders moving along congruent crossing paths; and in strong barrier coverage, we need to detect intruders with arbitrary moving paths. Most of studies in coverage problem of wireless sensor networks, sensors are supposed to have an omni-directional sensing model, in which the sensing range of a sensor is mostly used a disk model and an object can be covered or detected by a sensor if it is within the sensing range of the sensor [6]. In [7] authors investigated strong barrier coverage using directional sensor networks in which sensor nodes have arbitrarily tunable orientations to provide good coverage. They used directional barrier graph (DBG) to model this barrier coverage problem and employed DBG to solve this problem quickly.

In weak barrier coverage, we require that to cover every crossing path covered by deployed sensor nodes in networks and it is not important to us that how long it is. Figure 1 show that the sensor nodes deployment in a border not providing global barrier coverage due to existence of uncovered crossing path (which is more than 999 km long). A path is a crossing path if it crosses from top to bottom [3]. In real implementation of wireless sensor networks, intruders are highly unlikely to follow such paths; it is more likely that a short path across the belt region is taken.

In this paper, we propose a learning automaton based algorithm to guarantee strong barrier coverage of wireless sensor networks. We equip each node in network by learning automaton which helps each node to find best node to do coverage operation in every time of simulation in network. To assure barrier coverage, it is enough to find a path from the left hand of network to the right hand in network. In the proposed algorithm, whatever the number of nodes that selected by our approach be low, we can gain much network lifetime. So, the proposed method tries to find a shortest path between left hand (which presented with S) and right hand (which denoted with T) by learning automata. In the next section, we first state model, definition, and some earlier results that have been gained recently.

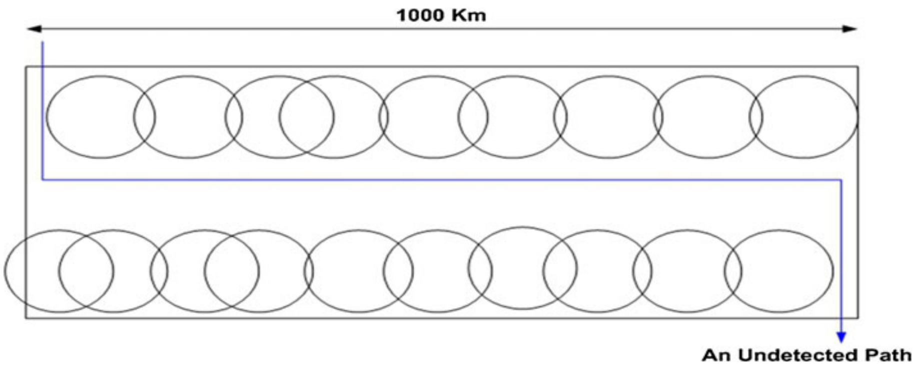

Fig. 1 A belt is not 1-barrier covered because of the existence of a long uncovered crossing path

The main contributions of this paper are as following:

- We use coverage graph of deployed sensor network as a base model to propose our approach. Coverage graph with edge length plays the role of environment for our algorithm.
- We propose a learning automaton based method to find near optimal solution for barrier coverage problem.

The rest of the paper is organized as follows. In Sect. 2, we illustrate model and definitions in the filed of barrier coverage. In Sect. 3, we present related works in the field of energy efficiency barrier coverage problem. Learning automata as a basic learning strategy used in the proposed method will be discussed in Sect. 4. In Sect. 5, the proposed algorithm is presented. Section 6 presents the simulation results and Sect. 7 concludes the paper.

2 Network Model and Problem Description

In this section, we introduce assumption, model, and defintions that are used in our proposed approach.

Assumption 2.1 We consider a rectangular two dimension belt region with size L $\times$ W and N sensor nodes are randomly deployed in this belt region with a high enough density to build a sensor barrier path.

Assumption 2.2 We assume a binary sensing and communication model, ie. two sensor nodes are able to communicate if their distance is less than R_{tx} which R_{tx} is transmission range of each node and each sensor is able to monitor a circular area of radius R_s and also, we assume $R_{tx} \geq 2R_s$.

Assumption 2.3 Intrusion to network is assumed to occur from top to bottom of region in deployed network.

Definition 2.1 A coverage graph of a sensor network with N sensor nodes is constructed as follows. Let G(N) = (V;E): The set V consists of a vertex corresponding to each sensor node. An edge exists between two sensor nodes if their sensing regions overlap in the deployment region R. In addition, it has two virtual nodes, s and t to correspond to the left and right boundaries. An edge exists between s or t if the sensing region of node overlaps with the left boundary (or right boundary) of the region.

If a sensor is placed within the sensing range R from the left (or right) boundary, it has an edge to a sink node and considered to have a virtual arc. An example of the disk graph is shown in Fig. 2a.

Definition 2.2 (*Probability of link failure* (λ):) We also define λ parameter be probability of link failure between nodes in network. The value parameter of this parameter shows amount of responses that not received accurate from other nodes. This is due to in wireless sensor network; we have different type of applications that require different communication reliability between nodes‘ links. In the case of barrier coverage problem, we desire to monitor barrier without loss information and also reliable transfer monitored data to base station in network.

A barrier path should satisfy the following requirements;

Requirement 2.1 A barrier path should cover the sensing area of network in which any intruders cannot peneterate to network.

Requirement 2.2 Sensor nodes in a barrier path should be able to transfer monitored data via sensor networks to outside by mutli-hop communications network.

An example of the disk graph is shown in Fig. 2a. A barrier path could be depicted on coverage graph G(N). If a path from virtual nodes s to t could be found which satisfies requirements 1 and 2 and also if multiple path exists between s and t they don't share any sensor nodes and they also satisfy the requirement 4. This constraint is called non-disjoint barrier path.

Theorem 2.1 *A network N provides k-barrier coverage iff there exist k node-disjoint paths between the two virtual nodes s and t in* $G(N)$ [8].

Intrusion detection by a barrier path should be transfers via multi-hop communications as soon as possible. Hence a barrier path is a shortest path in network from s to t.

Requirement 2.3 A barrier path is a shortest path in coverage graph G(N).

Requirement 2.4 Each barrier path operates independently and a sensor node should belong to only one barrier path.

Figure 2b shows the coverage graph of deployed network in Fig. 2a. Our algorithm subject is to find shortest barrier path in coverage graph with learning automata.

The network lifetime is an elapsed working time from the system starts to the time when the object detection quality cannot be met for the first time.

3 Related Work

Coverage problem deals with the ability of a network to monitor a certain area or events. This problem in wireless sensor networks is classified into three main different types [9]; area coverage, point coverage, and barrier coverage. The main objective of area coverage problem monitor the whole area of the network with respect to different performance criteria such as coverage ratio, minimum number of sensors providing desired coverage level during the maximum lifetime of the network. The node sleeping scheduling algorithms mostly are

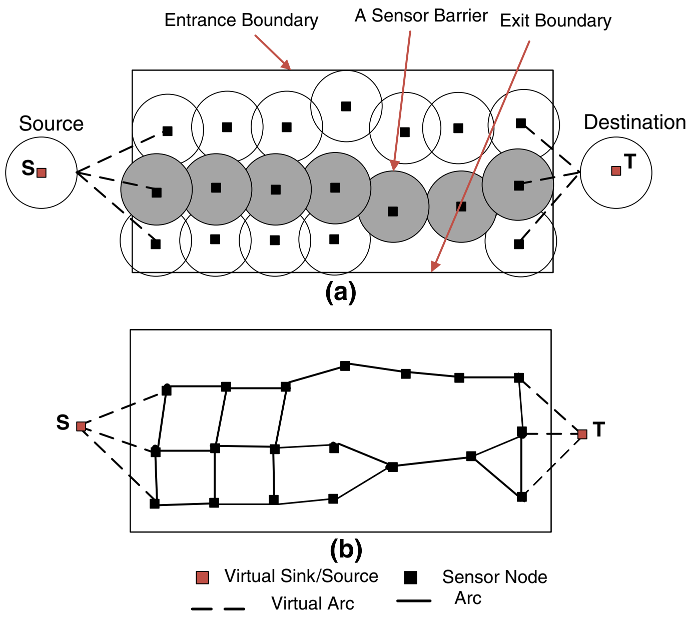

Fig. 2 An example of disk graph and coverage graph

used to maximize network's lifetime. Point coverage (target): the objective of point coverage problem is to cover a set of stationary or moving points. Scheduling sensor nodes into cover set is mostly used in different approach is used to solve this problem.

Barrier coverage was first introduced in robotic system [10]. It can be considered as the coverage with the goal of minimizing the probability of undetected penetration through the barrier (sensor network). This type of coverage problem needs less number of sensors than full coverage problem. The notation of barrier coverage in sensor networks defined in [8]. Authors proposed a centralized approach to find k-barrier in network. The centralized algorithm could cause high-energy consumption cost and communication overhead. It also may not suitable for manually deployed sensor nodes in networks. Authors [2] proposed Randomized Independent Sleeping (RIS) algorithm that used for computing a sleeping schedule for barrier coverage. Their algorithm neither guarantees barrier coverage nor provides any guarantee on performance.

He and Shi [11] proposed a distributed algorithm to find maximum number of barrier coverage in wireless sensor networks. Their method works for any size of sensor nodes and any shape of field in networks. Their proposed approach complexity is in order of O(n^2), if n be the number of deployment nodes in network.

Yang and Qiao [12] proposed an energy efficient method based on collaborations and information fusion between neighboring sensors to decrease the number of active nodes. Saipulla and et al. [13] studied barrier coverage with line based deployment in wireless sensor networks. They compared their proposed approach with Poisson model.

In [14], Silvestri proposed a distributed and asynchronous algorithm for k-barrier coverage in sensor networks with mobile nodes. In [7,13] authors provided barrier coverage with mobile sensor nodes. Huan Ma et al. [7] used the energy cost and reliability as objectives respectively to study two problems of k-barrier coverage: the minimum energy cost k-barrier coverage problem in static wireless sensor networks and the maximum k-barrier coverage problem in limited mobile wireless sensor networks. They proposed wo heuristic algorithm based on linear programming relaxation. In [13], authors presented a barrier coverage algorithm for line-based sensor deployment strategy and explore how to use sensor node mobility to improve barrier coverage. They proposed an efficient algorithm to relocate sensor nodes after randomly deployment to develop barrier coverage.

In [11] authors build barrier sensor with minimum cost in sensor networks. They provided a distributed algorithm to solve minimum-cost barrier coverage problem in asynchronous wireless sensor networks and cost is defined any consumed energy by sensor nodes. Their approach just used form neighbors nodes information to monitor barrier.

A localized algorithm to discover every crossing path confined to a slice of belt region proposed in [15,16]. However, a crossing path that stretches over the length of the slice may not be detected by the localized barriers. They proved that their localized approach guarantees local barrier coverage.

Authors [17] concentrated on the problem how to relocate mobile sensors to construct k sensor barriers with minimum energy consumption. They modeled the problem as integer linear programming and proved this problem is NP-Hard. Then proposed an approximation algorithm AHGB to construct one energy-efficient sensor barrier. Based on AHGB, a Divide-and-Conquer algorithm was proposed to achieve k-barrier coverage for large sensor networks. Ssu et al. [18] studied the k-barrier coverage problem in directional sensing model and developed a distributed algorithm to construct k-barrier coverage in randomly deployed directional sensor fields.

4 Learning Automata

A learning automaton is an adaptive decision-making tool that operates in unknown random environments. It has a finite set of actions to choose at each state and choose an action based on action probability vector. For each action that is chosen by learning automaton, the environment gives a signal based on probability distribution. The automaton update its action probability based on reinforcement signal that environment gives to random selected action. The main objective of learning automaton is to find optimal action among action set. It tries to minimize average penalty that received from the environment. Figure 3 illustrates relationship between automaton and environment.

Environment can be described by the triple $E = \{\alpha, \beta, c\}$ where $\alpha = \{\alpha_1, \ldots, \alpha_r\}$ denotes finite input set, $\beta = \{\beta_1, \ldots, \beta_r\}$ represents the output set that can be given by reinforcement signals, and $c = \{c_1, c_2, \ldots c_r\}$ is a set of penalty probabilities, where each element c_i of c corresponds to one input of action α_i. The environment can be classified into: P-model, Q-model, and S-model based reinforcement signal. The environment in which β can take only two binary values 0 or 1 is referred to P-model environment. Another class of the environment allows a finite number of the values in the interval [0, 1] can be taken by

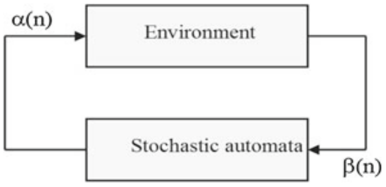

Fig. 3 The relationship between the learning automaton and its random environment

the reinforcement signal. Such an environment is referred to as Q-model environment. In S-model environments, the reinforcement signal lies in the interval [a, b]. Learning automata are classified into fixed-structure stochastic, and variable-structure stochastic. In the following, we consider only variable-structure automata [19].

A learning algorithm can be defined as follows $p(n+1) = T[p(n), \alpha(n), \beta(n)]$. Let $\alpha(k)$ and p(k) denote the action chosen at instant k and the action probability vector on which the chosen action is based, respectively. The repetition equation shown by 1 and 2 is a linear learning algorithm by which the action probability vector p is updated. Let $\alpha_1(k)$ be the action chosen by the automaton at instant k.

$$\begin{aligned} p_i(n+1) &= p_i(n) + a(1 - p_i(n)) \\ p_j(n+1) &= (1-a)p_j(n) \quad \forall j, j \neq i \end{aligned} \tag{1}$$

when the taken action is rewarded by the environment (i.e.,) and

$$\begin{aligned} p_i(n+1) &= (1-b)p_i(n)) \\ p_j(n+1) &= \frac{b}{r-1}(1-b)p_j(n) \quad \forall j, j \neq i \end{aligned} \tag{2}$$

when the selected action is penalized by the environment (i.e., $\beta(n) = 1$), r is the number of actions that can be a and b denote the reward and penalty parameters and determine the amount of increases and decreases of the action probabilities, respectively. In these two equations, a and b are reward and penalty parameters respectively. For $a = b$, learning algorithm is called Linear Reward–Inaction(L_{R-I}) algorithm, for $b \ll a$, it is called Linear Reward epsilon Penalty ($L_{R-\varepsilon P}$) algorithm, and for $b = 0$, it is called linear reward–penalty(L_{R-P}) algorithm [20]. In [21–24] some usage of learning automata for coverage problem and deployment strategy in sensor networks are introduced.

4.1 Action Probability Updating

In this section, we restate the variable structure stochastic automata (VSSA). VSSA are the ones in which the state transition probabilities are not fixed. In such automata, the state transitions or the action probabilities themselves are updated at every time instant using a suitable scheme. The transition probabilities and the output function in the corresponding Markov chain vary with time, and the action probabilities are updated on the basis of the input. VSSA depend on random number generators for their implementation. The action chosen is dependent on the action probability distribution vector, which is, in turn, updated based on the reward/penalty input that the automaton A variable-structure automaton is defined by the quadruple $\{\alpha, \beta, P, T\}$ in which $\alpha = \{\alpha_1, \ldots, \alpha_n\}$ represents the action set of the automata, $\beta = \{\beta_1, \ldots, \beta_n\}$ represents the input set, $P = \{P_1, \ldots P_n\}$ represents the action probability set, and finally $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ represents the learning algorithm. This automaton operates as follows. Based on the action probability set p, automaton randomly selects an action, and performs it on the environment. After receiving the environment's reinforcement signal, automaton updates its action probability set based on Eq. (1) for favorable responses, and Eq. (2) for unfavorable ones.

5 Proposed Method

In the proposed algorithm network operations are divide into different rounds and the proposed learning automata based scheduling algorithm consists of 3 major phases; Initialization, learning, and monitoring. Each round starts with *initialization* phase and continues with *learning* and *monitoring* phase. During the *initialization* phase, coverage graph of deployed sensor nodes build and each sensor node in the network finds its outgoing neighborhood nodes. At the end of this phase all node in network know its outgoing neighbors and each outgoing edge of a node is one of the actions of this learning automaton. Selecting nodes to do barrier coverage is done in the *learning* phase and each node with help of learning automata tries to find best outgoing nodes to build barrier path. At the end of this phase the action probability vector of each node which is selected by node S (source node) to node T(destination node) is valued. Finally, scheduling the active times of sensor nodes is performed during the *monitoring* phase. At the end of this phase each node operates based on best action which means to be in barrier path or not. We describe these three phases in more details in the subsequent sections.

5.1 Initialization

1. LABC algorithm obtains a snapshot of the directed coverage graph with each edge having a length. To do this, each node broadcasts an *Initial* packet to its neighborhood. The node then listens to receive *Initial* packets from its neighbors. The algorithm maintains an action probability vector, $P = \{p_1(n), p_2(n) \ldots p_r(n)\}$, for *each* sensor node of the coverage graph that contains the probability values for choosing different actions, $\alpha = \{\alpha_1, \ldots, \alpha_r\}$, offered by the random environment which are the edges leaving the node. These are modeled as the actions. $p_i(n)$ represents the probability of choosing action α_i at the nth time instant. All elements of the action probability vector of a particular sensor node are initialized to have a value equal to one divided by the outdegree of that sensor node. For example, if a node has four outgoing edges (four possible actions) and the action probability vector for that node initially is {0.25, 0.25, 0.25, 0.25}, we use our assumptions in Sect. 2 to build coverage graph. A higher probability value demonstrates a superior action. In this case, each node has a number of possible outgoing edges in coverage graph. Each possible outgoing edge corresponds to a probable action that can be selected for calculating the shortest barrier path tree. Based on the chosen action, the system performs shortest barrier path computations, whence it finds whether the random environment inputs the automaton with $\beta = \{0, 1\}$, where 0 represents a reward and 1 a penalty. Let α_i denote the action corresponding to choosing the outgoing edge (i; j) from node i to node j, and let dist(i) be the shortest barrier path distance of i from the source, $\ell\{i,j\}$ and the weight of edge (i; j).

$$\beta = 0 \Rightarrow dist(i) + \ell\{\mathrm{i,j}\} \times \lambda < dist(j) \tag{3}$$

$$\beta = 1 \Rightarrow dist(i) + \ell\{\mathrm{i,j}\} \times \lambda \geq dist(j) \tag{4}$$

2. After obtaining snapshot from network in first step, Dijkstra's Algorithm is run once to determine the shortest path edges on the coverage graph snapshot. Based on this, the action probability vector of each node is updated such that the outgoing edge from a node which is determined to belong to the shortest barrier path edge has an increased probability than before the update. Each learning automata updates action probability vector by using L_{R_I} learning algorithm as described over variable action learning automata in Sect. 4.1.

5.2 Learning

3. After the *Initialization phase*, then learning automaton of a sensor node is randomly chosen from the current coverage graph. For this sensor node, based on the action probability vector, an edge is chosen as follows. For example, if a sensor node has four outgoing edges (four

possible actions) and the action probability vector for that node is {0.05, 0.15, 0.2, 0.6}, the edge (action) chosen is selected based on this distribution.

4. Based on selected edge in step3, the environment is requested for the correct edge-weight of this edge which is randomly selected above. Since the edge-weights are real numbers, the edge-weight should have increased/decreased. Thus, the current shortest barrier path's tree is calculated.

5. In this step, the action probability vector for the sensor node whose edge was just selected is updated such that the edge that could now potentially belong to the shortest barrier path's tree has a greater likelihood of being selected than before the update.

6. Steps 3–5 above are repeated a large number of times until the algorithm converges.

5.3 Monitoring

7. In barrier monitoring phase, nodes for which, the probability of selecting action of their LA passes this threshold, becomes an active node. Active nodes will be active to monitor the barrier until end of current barrier monitoring phase. Other nodes switch to low consumed energy state. Each active node in this phase builds the nodes in barrier path. The next round will be started when the current monitoring phase is over. These phase will continue until atleast barrier path can be built.

The formal reasoning of why the LABC algorithm works probably relies on the *ε-optimal* property of the scheme, the shortest-path property of Dijkstra's algorithm. We provide below an intuitive reasoning (without a formal proof) in support of our above statement.

A close inspection of the LABC algorithm shows that the only place it calculates the shortest barrier paths are when executing Dijkstra's algorithm on the coverage graph's snapshot taken at the initialization step from deployed sensor networks. Dijkstra's algorithm is well-known to find the optimal shortest paths on a given graph topology.

Since Dijkstra's algorithm is initially executed on the static coverage graph snapshot, it necessarily calculates the *initial* optimal shortest paths. The proof that the shortest barrier path is found (for each automaton) with a probability as close to unity as desired, probably follows because of the fact that the action probability updating scheme used to select the edges for a given node, the L_{R_I}, is *ε-optimal* (this result would be certainly true if the choices of, and responses to the individual automata are independent). Thus, given the former, the probability of the scheme choosing the most pertinent edge (for every node) will be arbitrarily close to unity. Therefore, it is plausible that as the number of iterations tends toward infinity, the probability of choosing the optimal action (edge) for each node tends toward unity, and the overall barrier path costs will tend to the shortest barrier path costs of the mean edge weights—which has also been confirmed experimentally.

6 Simulation Results

To evaluate the performance of the proposed method several experiments have been conducted. All the experiments are implemented in [25] and run on a core i5 CPU 2.5-GHz machine with 3-G RAM. In these simulations, a fixed sensor network is considered, in which all sensor nodes are randomly deployed within a 2 km × 100 m area. Sensing ranges of all sensor nodes are equal. Parameters of the conducted simulations are as follows:

- r, the sensing range. We vary the sensing range of sensor nodes in the range 30–60 m.

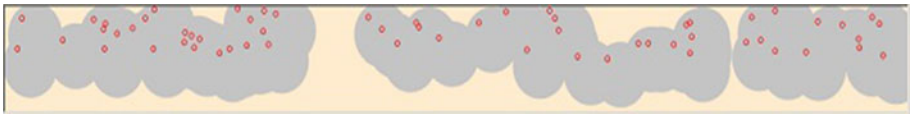

Fig. 4 Not supported barrier coverage with our simulator

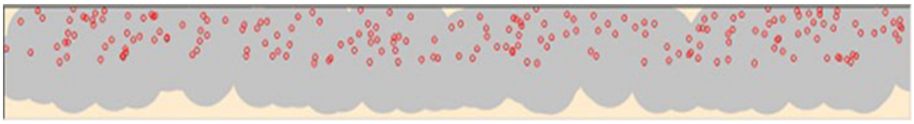

Fig. 5 Supported barrier coverage with our simulator

- N, the number of sensor nodes. We vary the number of sensor nodes in the range [400, 1,400] to study the effect of node density on the performance of LABC.
- initial energy for each sensor is supposed 10 weeks.
- Probability of link failure(λ). We vary the probability of link failure in the range [0.05–0.25].

Energy consumption of sensor nodes for communication tasks follows the first order energy model given in [26]. Energy required to switch a node from sleep to active mode is assumed to be negligible. Simulations are performed in wireless sensor network simulator given in [25]. Results are averaged over 15 runs.

Figure 4 shows the network that cannot support barrier coverage with our scattered sensor nodes by our simulator and also Fig. 5 shows the network can support barrier coverage of wireless networks by our deployed nodes.

6.1 Experiment 1

We first study performance of the LABC algorithm in term of network lifetime and investigate how much longer lifetime can gain by incresing sensor nodes in deployed network. For this experiment, we suppose the simulation parameters as follows: sensing range is set to 30(m) and the network size varies from 400 to 1,400 step by 200 sensor nodes. Figure 6 illustrates network lifetime of each algorithm. It can be seen from the figure that in proportion as we increase number of sensor nodes, the network lifetime increase and network lifetime considerably increase with increasing number of nodes.

6.2 Experiment 2

In this experiment, we study the effect of the sensing ranges of the sensor nodes on the lifetime of the network in the proposed scheduling mechanism. We set the number of the sensors vary between 400 and 1,400. The results of this experiment, which are given in Fig. 7, indicate that the network lifetime increases as the sensing ranges of the sensor nodes increase. With increasing the sensing radius of each node in network, fewer sensors will be needed to monitor the barrier path, and hence, more sensor nodes in the network can switch to the sleep state, saving their energies for future rounds. In Fig. 8, we perform the same simulation with network size 500 sensor nodes to study the effect of the distribution of sensor nodes with different sensing ranges on the performance of the proposed algorithm. We compared

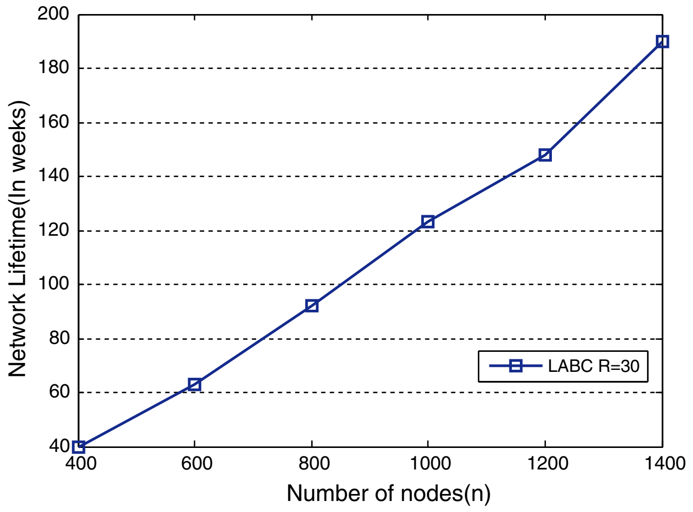

Fig. 6 Network lifetime with our proposed algorithm for R = 30 m

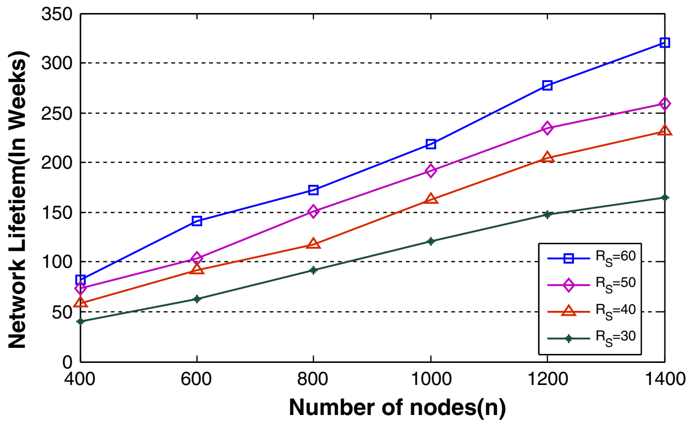

Fig. 7 A network lifetime with different sensing range and nodes

Fig. 7 with Fig. 8 and observed that even the network lifetime of each curve very close to each other.

6.3 Experiment 3

In this experiment, we assess effect of λ parameter on network lifetime. This experiment has done with the same parameters that used in first experiment. As indicated in Fig. 9, when the value of λ parameter is big, we cannot gain the long lifetime because in this case most of the network nodes are fail to give correct response. On the other hand, when the value of this parameter is low, we can obtain more network lifetime. We did this experiment for fixed number of sensor nodes to investigate the effect of λ values on the performance of proposed approach. and also observed the same trend. Figure 10 demonstrate the results of this simulation.

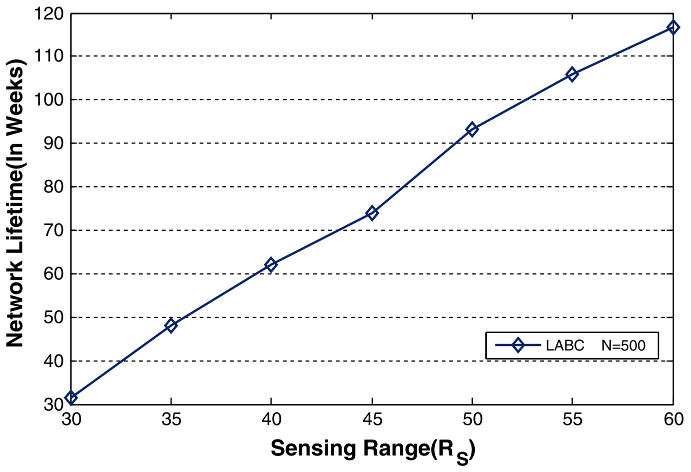

Fig. 8 Network lifetime with N = 500 and different sensing range

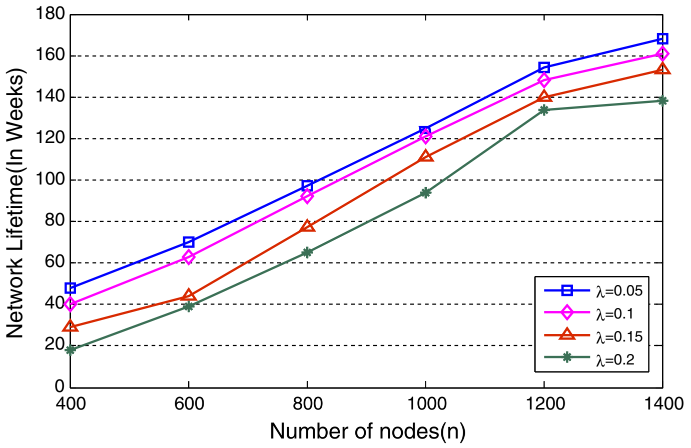

Fig. 9 Effect of λ parameter on network lifetime with different nodes

6.4 Experiment 4

In this experiment, we study the impact of the learning rate, used in the proposed algorithm, on the network lifetime. We consider the following learning rates: 0.01, 0.1, 0.2, and 0.4. Additional simulation parameters are as follows: sensing range is set to 30(m) and the network size varies from 400 to 1,400 step by 200 sensor nodes. The result of this experiment, which is given in Fig. 11, shows that by decreasing the learning rate, the network lifetime also increases. In other words, increasing the (computational and communicational) complexity of the learning phase of the proposed algorithm (by decreasing the learning rate) is not a waste of resources, since this results in more better scheduling of the activity states of the sensor nodes, which consequently results in the network lifetime to increase. In this experiment we also compared effect of learning rate on fixed deployed sensor nodes to study the effect of

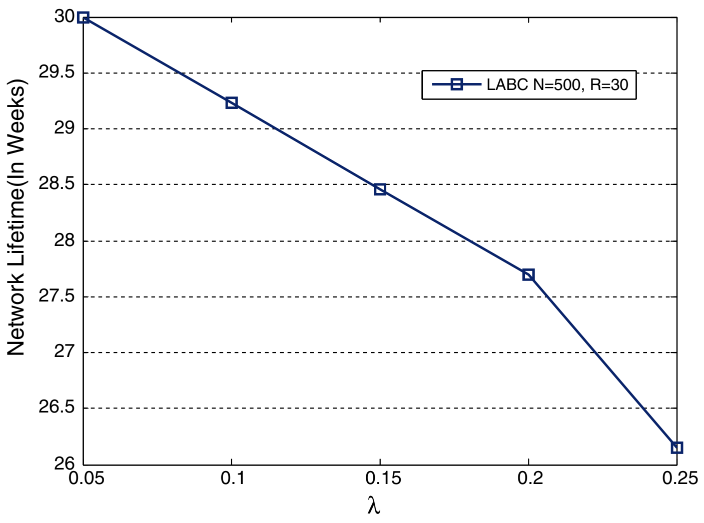

Fig. 10 Effect of λ parameter on network lifetime with fixed nodes

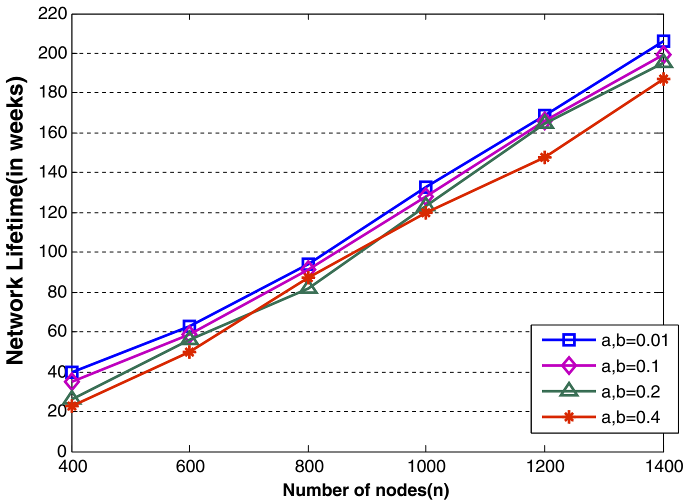

Fig. 11 Impact of learning rate on network lifetime

the learning rate on the performance of LABC. Figure 12 shows the same trend in network lifetime.

6.5 Experiment 5

In this experiment, the LABC algorithm compared with Randomized Independent Sleeping (RIS) [8], which is a localized algorithm, and with the optimal (centralized) algorithm of [27] in term of network lifetime. For this experiment, we suppose the simulation parameters as follows: sensing range is set to 30(m) and the network size varies from

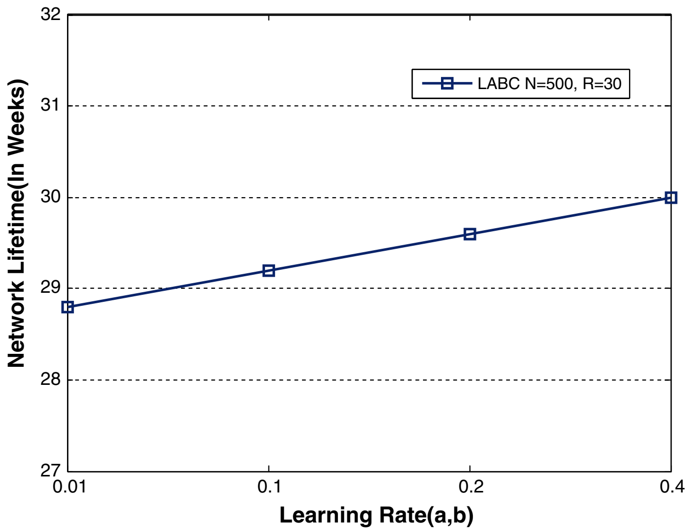

Fig. 12 Impact of learning rate on network lifetime with N = 500 and R = 30

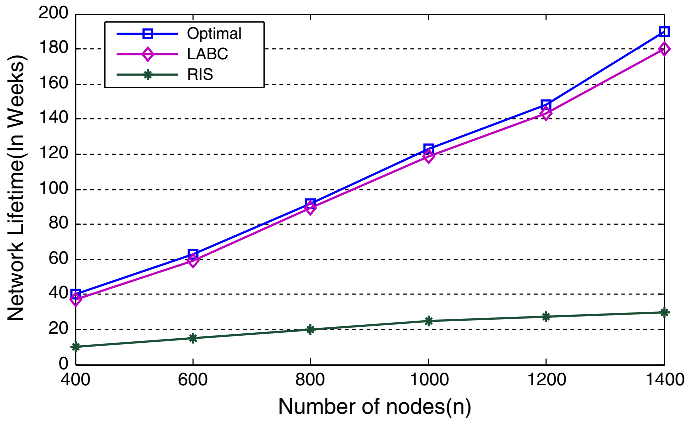

Fig. 13 Network lifetime achieved with LABC in comparison with RIS and Optimal

400 to 1,400 step by 200 sensor nodes. The simulation results are shown in Fig. 13. It can be seen from the figure the proposed algorithm not only outperforms the RIS algorithm by up to 6 times but also the provided network lifetime is very close to optimal method.

In the next simulation, we compare the results our algorithm with RIS and Optimal methods in term of network lifetime with different network width. We used the same simulation parameters as above. Figure 14 shows the results of this simulation. From this figure, we can conclude that with increasing network width, the gained network lifetime increase in our proposed algorithm. This is due to in coverage graph of our algorithm learning automata of each node could find a number of proper neighbor nodes to monitor barrier path.

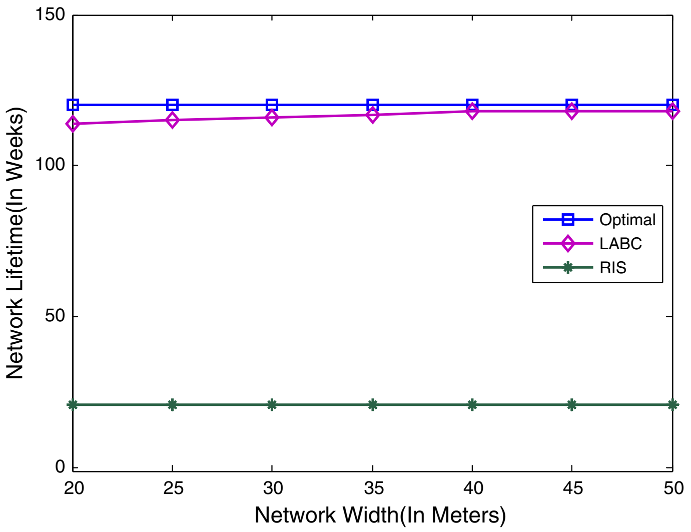

Fig. 14 Network lifetime achieved with LABC as the value of network width is varied when 1,000 nodes is randomly deployed in the network

7 Conclusions

There are many rigorous resource limitations on sensor networks such as battery power, memory size, computational capacity, and etc. scheduling sensor nodes into proper state in any time in network can preserve these critical resource. In this paper, we proposed a novel method based on learning automata for barrier coverage in wireless sensor networks. In this approach, each node in the deployed network is equipped with a learning automaton and also we used network coverage graph to introduce our approach. Learning automata used to select best nodes to guarantee barrier coverage of wireless sensor networks and residual nodes switch to low consumed energy to save their resources. To determine the effectiveness of our learning automata based algorithm, we performed different simulations.

References

1. Arora, A., Dutta, P., Bapat, S., Kulathumani, V., Zhang, H., Naik, V., et al. (2004). A line in the sand: A wireless sensor network for target detection, classification, and tracking. *Computer Networks*, *46*(5), 605–634.
2. Kumar, S., Lai, T. H., & Balogh, J. (2004). On k-coverage in a mostly sleeping sensor network. In *ACM MobiCom'04* (pp. 144–158).
3. Li, X.-Y., Wan, P.-J., & Frieder, O. (2003). Coverage in wireless ad-hoc sensor networks. *IEEE Transactions on Computers*, *52*(6), 753–763.
4. Meguerdichian, S., Koushanfar, F., Potkonjak, M., & Srivastava, M. (2001). Coverage problems in wireless ad-hoc sensor networks. In *Proceedings of the IEEE InfoCom (InfoCom'01)* (pp. 115–121). Anchorage, AK.
5. Liu, B., Dousse, O., Wang, J., & Saipulla, A. (2008). Strong barrier coverage of wireless sensor networks. In *ACM MobiHoc* (pp. 411–419).
6. Liu, C., & Cao, G. (2011). Spatial-temporal coverage optimization in wireless sensor networks. *IEEE Transaction on Mobile Computing*, *10*(4), 465–478.
7. Ma, H., & Al, E., (2012). Energy efficient k-barrier coverage in limited mobile wireless sensor networks. *Computer Communications*.

8. Kumar, S., Lai, T. H., & Arora, A. (2005). Barrier coverage with wireless sensors. In *Proceedings of ACM MobiCom* (pp. 284–298). Cologne, Germany.
9. Zhu, C., Zheng, C., Shu, L., & Han, G. (2012). A survey on coverage and connectivity issues in wireless sensor networks. *Journal of Network and Computer Applications*, *35*, 619–632.
10. Gage, D. W. (1992). Command control for many-robot systems. *Unmanned Systems Magazine*, 22–24.
11. He, J., & Shi, H. (2012). Constructing sensor barriers with minimum cost in wireless sensor networks. *Journal of Parallel and Distributed Computing*, *71*, 1654–1663.
12. Yang, G., & Qiao, D. Barrier information coverage with wireless sensors. (2009). In *28th IEEE International Conference on Computer Communications, INFOCOM* (pp. 918–926). Rio de Janeiro, Brazil.
13. Saipulla, A., Westphal, C., Liu, B., & Wang, J. (2010). Barrier coverage of line-based deployed wireless sensor networks. *Ad Hoc Network*, 127–135.
14. Silvestri, S. (2011). MobiBar: Barrier coverage with mobile sensors. In *Proceedings of the the IEEE Global Communications Conference (GLOBECOM)* (pp. 1–6).
15. Chen, A., Kumar, S., & Lai, T. (2010). Local barrier coverage in wireless sensor networks. *IEEE Transaction on Mobile Computing*, *9*(4), 491–504.
16. Chen, A., Kumar, S., & Lai, T. H. (2007). Designing localized algorithms for barrier coverage. In *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking, MobiCom'07, ACM* (pp. 63–74). New York, NY, USA.
17. Ban, D., Yang, W., Jiang, J., Wen, J., & Dou, W. (2010). Energy-efficient algorithms for k-barrier coverage in mobile sensor networks. *International Journal of Computers Communications and Control*, *5*, 616–624.
18. Ssu, K., Wang, W., Wu, F., & Wu, T. (2009). K-barrier coverage with a directional sensing model. In *Proceedings of the International Journal on Smart Sensing and Intelligent Systems* (pp. 75–83).
19. Thathachar, M. A. L., & Sastry, P. S. (2002). Varieties of learning automata: An overview. *IEEE Transaction on Systems, Man and Cybernetics-Part B: Cybernetics*, *32*(6), 711–722.
20. Narendra, K. S., & Thathachar, M. A. L. (1989). *Learning automata: An introduction*. Englewood Cliffs: Prentice Hall.
21. Mostafaei, H., & Meybodi, M. R. (2013). Maximizing lifetime of target coverage in wireless sensor networks using learning automata. *Wireless Personal Communications*, *71*(2), 1461–1477. doi:10.1007/s11277-012-0885-y.
22. Mostafaei, H., Meybodi, M. R., & Esnaashari, M. (2010). A learning automata based area coverage algorithm for wireless sensor networks. *Journal of Electronic Science and Technology*, *8*(3), 200–205.
23. Mostafaei, H., Meybodi, M. R., & Esnaashari, M. (2010). EEMLA: Energy efficient monitoring of wireless sensor network with learning automata. In *International conference on signal acquisition and processing* (pp. 107–111). Bangalore, India.
24. Esnaashari, M., & Meybodi, M. R. (2010). A cellular learning automata-based deployment strategy for mobile wirelesssensor networks. *Journal of Parallel and Distribted Computing*, *71*(7), 988–1001. doi:10.1016/j.jpdc.2010.10.015.
25. http://www.djstein.com/projects/index.html.
26. Heinzelman, W., Chandrakasan, A., & Balakrishnan, H. (2000). Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the 33rd Hawaii international conference on system sciences* (pp. 1–10). Hawaii, USA.
27. Kumar, S., Lai, T. H., Posner, M. E., & Sinha, P. (2007). Optimal sleep-wakeup algorithms for barriers of wireless sensors. In *Fourth International Conference on Broadband Communications, Networks, and Systems (IEEE BROADNETS)* (pp. 327–336). Raleigh, NC.

Habib Mostafaei received the B.S. degree from the Islamic Azad University Khoy branch, in 2006 and the M.S. degree from the Islamic Azad University Arak branch, in 2009, both in software engineering. He joined the faculty of the Computer Engineering Department at Urmia Azad University in 2009. He is a reviewer for Journal of Network and Computer Applications (JNCA) and Computer & Electrical Engineering (COMPELECENG) and Wireless Networks(WINET). His research interests include learning systems, sensor networks, soft computing, and cloud computing.

Mohammad Reza Meybodi received his B.S. and M.S. degrees in Economics from Shahid Beheshti University in Iran during 1973 and 1977, respectively. He also received his M.S. and Ph.D. degrees from Oklahoma University, USA during 1980 and 1983, respectively in Computer Science. Presently, he is a Full Professor in Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran. Prior to the present position, he worked from 1983 to 1985 as an Assistant Professor at Western Michigan University, and from 1985 to 1991 as an Associate Professor at Ohio University, USA. His research interests include wireless networks, fault tolerant systems, learning systems, parallel algorithms, soft computing and software development.