Wireless Networks (2020) 26:4411–4429
https://doi.org/10.1007/s11276-020-02350-y

A distributed reinforcement learning based sensor node scheduling algorithm for coverage and connectivity maintenance in wireless sensor network

Anamika Sharma[1] · **Siddhartha Chauhan**[1]

Published online: 5 May 2020

Abstract
The fundamental challenge for randomly deployed resource-constrained wireless sensor network is to enhance the network lifetime without compromising its performance metrics such as coverage rate and network connectivity. One way is to schedule the activities of sensor nodes and form scheduling rounds autonomously in such a way that each spatial point is covered by at least one sensor node and there must be at least one communication path from the sensor nodes to base station. This autonomous activity scheduling of the sensor nodes can be efficiently done with Reinforcement Learning (RL), a technique of machine learning because it does not require prior environment modeling. In this paper, a Nash Q-Learning based node scheduling algorithm for coverage and connectivity maintenance (CCM-RL) is proposed where each node autonomously learns its optimal action (active/hibernate/sleep/customize the sensing range) to maximize the coverage rate and maintain network connectivity. The learning algorithm resides inside each sensor node. The main objective of this algorithm is to enable the sensor nodes to learn their optimal action so that the total number of activated nodes in each scheduling round becomes minimum and preserves the criteria of coverage rate and network connectivity. The comparison of CCM-RL protocol with other protocols proves its accuracy and reliability. The simulative comparison shows that CCM-RL performs better in terms of an average number of active sensor nodes in one scheduling round, coverage rate, and energy consumption.

Keywords Coverage redundancy · Coverage rate · Network connectivity · Reinforcement learning · Nash Q-learning · Node activity scheduling

1 Introduction

In general terms, the main objective of an energy-efficient coverage-connectivity algorithm in wireless sensor network (WSN) is to monitor the area of interest and transmit the sensing results to the base station with a minimum number of sensor nodes [1, 2]. To accomplish this objective three types of solutions have been proposed in the literature: node deployment optimization [3, 4], node scheduling [4–6], and cover-set formation [7, 8]. Node deployment optimization is defined as to control the number of sensor nodes for deployment. WSN comprises of two types of sensor nodes: static and mobile. A WSN comprises of static sensor nodes is inflexible because it is not capable of handling topology change due to sensor node failure. On its contrary, a WSN having mobile sensor nodes are flexible due to its dynamic behavior. There are real-time applications of WSNs such as border surveillance, harsh geographic surveillance, and disaster management where the number of deployed nodes can't be controlled. To handle such situation node activity scheduling protocols have been proposed [4–8]. According to these protocols, the nodes can be in any one of three modes i.e. active, hibernate, and sleep. The active nodes in each scheduling round form a cover set. The objective of

✉ Anamika Sharma
anamika@nith.ac.in

Siddhartha Chauhan
sid@nith.acin

[1] Computer Science and Engineering Department, National Institute of Technology, Hamirpur, Himachal Pradesh 177005, India

each cover set is to provide maximum coverage and connectivity with a minimum number of sensor nodes.

These node scheduling protocols have disadvantages in terms of unwanted computational energy consumption and resource utilization such as: tolerating the overhead of communication among nodes before the actual scheduling phase and storage of the scheduling information so that nodes can be completely in sleep mode otherwise the nodes have to be in hibernate mode. In hibernate mode, the nodes can turn off their processing unit, but cannot turn off its radio (transceiver). Therefore, there is a need for a node scheduling protocol, which can intelligently manage the node's activity scheduling. This problem can be efficiently solved with the help of machine learning algorithms. These algorithms enable the sensor node to learn its action without being explicitly programmed. This is the one application of computational intelligence in WSN. Computational intelligence is a boon for other WSN challenges such as coverage and connectivity maintenance [9–12], data aggregation [13–15], cluster formation [15, 16], energy management [17], routing [18], node duty cycling at MAC layer [19, 20], and security [21]. It provides autonomy, flexibility, and robustness to WSN.

Kulkarni et al. [22] has done an extensive survey of computational intelligence in WSNs and presented the work of various researchers in this field in a concise manner. At first, the challenges in WSNs such as wireless ad hoc nature, energy limitation, physical distribution, localization, quality of service management, and security are very well explained. A brief overview of various models of machine learning algorithms such as neural networks, fuzzy logic, evolutionary algorithms, swarm intelligence, artificial neural networks (ANN), and reinforcement learning is presented. This paper also explains how computational intelligence has been hybridized to mitigate the various challenges of WSNs. To the best of our knowledge, this paper is the best survey paper for the beginners, who want to pursue their research in the field of computational intelligence in WSNs.

Wireless sensor network comprises of a set of sensor nodes, where each node is considered as an agent for intelligent computations. Each sensor node has a learning ability [10, 22]. Therefore, WSN is known as a multiagent system. Reinforcement learning is the only machine learning algorithm possible in such a scenario because it does not require any environment model. Reinforcement learning (RL) is a type of machine learning where an agent learns and takes actions during their learning process to gain long-term reward [23]. A reward can be either a positive reward or a negative reward. Reward function optimizes the correctness of action selection in a given state. The identification of relevant actors in a particular state is only possible after executing all states-actions combinations. Out of n number of executions only m combinations are considered, which can maximize the global reward. The execution of these combinations is known as a Markov Decision Process (MDP). MDP is a discrete-time stochastic control process that provides mathematical modeling for decision making, where outcomes are completely or partially random in nature [24]. The environment of RL is modeled as a Markov Decision Process.

Reinforcement learning applies to both single and multiagent systems. It has been stated earlier that WSN is multiagent system because of the autonomous distribution of sensor nodes. Each sensor node acts as an agent. The potential of extensive learning and the disintegration of a complex problem for better rewards are the key advantages of multiagent reinforcement learning [25]. However, in a multiagent system, the agents have less knowledge about their neighboring agents. At first, the agents learn about each other and then behave appropriately. Reinforcement learning has been tailored in various algorithms of WSNs such as data aggregation, routing, dynamic channel allocation, and resource management [26]. Yau et al. [27] have briefly summarized the applications of reinforcement learning in wireless sensor networks. This paper presents an implementation of one of the reinforcement learning algorithms i.e. Nash Q-learning to solve the problem of coverage and connectivity. The overall objective of reinforcement learning in WSN is to take the right decision at the right time.

1.1 Contribution

This paper focuses on the problem of energy efficient coverage and connectivity maintenance. From the literature, it has been concluded that a very few amount of research work have been done on coverage and connectivity maintenance using computational intelligence. In the present scenario, a quality amount of research is going on to embed intelligence into diverse issues and applications of WSN. Keeping in view of these objectives, this paper has proposed a coverage and connectivity maintenance protocol based on reinforcement learning (CCM-RL). The main contributions of this paper are listed below.

1. The sensor nodes are deployed to achieve the coverage up to threshold level and there must be at least one communication path among the sensor nodes. This protocol enables the sensor nodes to take right decision at right time.
2. The learning ability resides inside each sensor node learns their best action to maximize the coverage rate and maintain the connectivity among active node. The purpose behind this is to activate only a subset of

sensor nodes so that energy consumption can be minimized. To fulfill this objective, the sensor nodes do not need to communicate through notification messages. These nodes autonomously learn their best action using Nash Q-learning algorithm.

3. This paper provides a mechanism to eliminate the coverage redundancy among nodes by customizing the sensing range of overlapped sensor nodes. The sensor nodes customized their sensing range after learning their best action. The advantage of this work is to preserve the resources of WSN such has battery power and memory by mitigating the redundant packet generation for same spatial points, avoiding network congestion and contention.
4. CCM-RL provides a solution to preserve the desired coverage rate, connectivity among sensor nodes, and minimizes the total energy consumption.

The rest of the paper is classified as follows. Section 2 discusses concise review of the algorithms proposed in the literature. The WSN model and related assumptions are explained in Sect. 3. Section 4 presents a concise overview of reinforcement learning, Nash Q-learning and definitions related to reinforcement learning in WSN. The proposed protocol Coverage Connectivity Maintenance based on Reinforcement Learning (CCM-RL) is explained in Sect. 5. Section 6 presents the performance evaluation of CCM-RL and its comparison to existing protocols respectively. The conclusions are discussed in Sect. 7.

2 Literature review

For the past two decades, a quality amount of research work has been done to improve the performance of wireless sensor networks in terms of various parameters such as: coverage [1–8, 28], connectivity [1, 2, 28], network congestion [29], packet delivery rate [20], energy consumption [30], and security [31]. This paper emphasizes on coverage, connectivity, and energy consumption parameter. The coverage is categorized as target coverage, area coverage, and barrier coverage [32]. The network connectivity parameter is considered with only a few coverage protocols, whereas energy consumption is computed for each protocol. This is done so far because sensor nodes are tiny and resource-constrained devices with limited memory and battery power. Energy consumed by the wireless sensor nodes for sensing, computational processing, and communication has a direct and indirect impact on the WSN's lifetime [33]. However, it has been identified by many researchers that random deployment, coverage redundancy, redundant communications, and idle listening are responsible for unwanted energy consumption and reduce the network's lifetime. Sensor node activity scheduling is the prime solution, where a subset of sensor node is active and ensures the coverage rate up to a threshold level and connectivity among sensor nodes. The rest of the sensor nodes are in sleep mode to prevent unwanted energy consumption. A lot of researchers have contributed towards the solution of this problem. More et al. [34] have reviewed the key coverage protocol and also illustrated various open challenges such as heterogeneous nodes with obstacles, node failure probability, limited node mobility, coverage degree, and optimization of wake-up rate of sleeping nodes for energy-efficient coverage protocol. Cardei et al. [35] have proposed a solution to solve the problem of coverage redundancy in target coverage and stated this problem as an Adjustable Range Set Cover (AR-SC) problem. According to them, there still exists coverage redundancy after successful node activity scheduling.

Nowadays, to design smart networks and communications with resource optimization and energy management, intelligence has been embedded in the sensor nodes. Although sensor nodes are resource constrained devices, artificial intelligence and data mining techniques have given them the potential to reach a new level of computation, learning and reasoning [20]. The first step of a learning algorithm requires information about the environment and internal structure. This information is provided by the sensor nodes. For a succinct understanding of how to hybridize reinforcement learning systems in wireless sensor network, a lot of researchers published very good research papers [22, 26, 27, 36]. Seah et al. [9] have considered two main parameters i.e. coverage and energy consumption for a large sensor network. This is the first paper where the authors have solved coverage and energy consumption issues using a reinforcement learning algorithm. The authors have proposed a Q-learning based coordinated algorithm (COORD) to identify the coordination among sensor nodes so that area coverage can be maximized as well as total energy expenditure can be minimized. The deployed sensor nodes are multiagent. Each autonomous agent has its reward function and thus aims at maximizing its discounted reward. The learning steps are given reward based on action taken by the sensor node to cover the maximum grid points.

The Probabilistic Coverage Protocol [37] has been proposed for node activity scheduling where the sensor nodes are considered to be localized at a distance of s from each other and have a probabilistic sensing range. The main advantage of PCP is that it can perform coverage computation on both binary and probabilistic sensing models. The Scheduling Algorithm based on Learning Automata (SALA) [10] has been proposed to cover the periodic events and dynamic target points with a minimum number of sensor nodes. The periodic event distribution is based on the Poisson distribution. In SALA, each sensor node is embedded with the set of learning automata. The learning automata enables the sensor node to learn maximum sleep time without compromising the dynamic target detection rate. The learning of sleep time has been dependent on the target trajectory. The key emphasis of SALA is to successfully cover the dynamic event and target points with a minimum number of sensor nodes. These nodes maximize the detection rate to reduce the energy consumption rate. SALA has not shown that whether the active sensor nodes in each scheduling round have connectivity among themselves and there is no network partitioning.

Mohamadi et al. [38] have proposed a scheduling algorithm based on learning automata where the sensor nodes have learnt the appropriate sensing range to cover all target points in each cover set and maximize the network lifetime. They have stated this problem as Maximum Network Lifetime with Adjustable Sensing Range (MNLAR). The sensor nodes are equipped with learning automata based on pruning rules to learn their appropriate sensing range to form a cover set in each scheduling round. Sleep scheduling with Predictive Coverage Redundancy Check (SPRC) [39] has been proposed to solve the problem of coverage redundancy cause due to the random deployment of sensor nodes in abundance to achieve maximum coverage. In reality, it caused wastage of network resources and also increase the network cost. The scheduling algorithms have to perform repeated coverage redundancy checks. Therefore, SPRC proposed an analytical model to predict the need for coverage redundancy check before each scheduling round to minimize the computational scheduling energy consumption. Reinforcement Learning based Sleep Scheduling for Coverage (RLSSC) [11] has been proposed to optimize the selected action of the sensor nodes in each scheduling round using Q-learning. The conventional approach has been applied to schedule the activities of the sensor nodes. RLSSC is a two-step scheduling algorithm. In the first step, it has identified the coverage redundancy among sensor nodes. Secondly, sensor nodes have learnt their best action based on Q-learning. These algorithms entirely emphasize on maximizing the coverage and minimizing the energy consumption, however, the connectivity among nodes is completely ignored for successful communication among nodes.

Mostafaei et al. [12] have proposed a Partial Coverage with Learning Automata (PCLA) protocol, which has focused on issues such as partial coverage and continuous monitoring of the target point. The main objective of PCLA is to minimize the active number of sensor nodes for desired area coverage and connectivity among sensor nodes. PCLA provides a probabilistic framework to select the most eligible sensor nodes for desired area coverage and network connectivity. Though PCLA has been designed for large scalable wireless sensor networks, but it does not present any solution for handling coverage redundancy.

Coverage Contribution Area (CCA) [5] is a centralized and distributed coverage protocol that has been proposed to solve the k-coverage problem in random deployment of sensor nodes. The residual energy and the spatial density of the sensor nodes is the eligibility criteria for the sensor nodes to become active in a scheduling round. However, CCA neither incorporated any learning mechanism for node activity scheduling nor provided any solution to preserve connectivity among sensor nodes. A brief parametric comparison of existed protocols based on machine learning and CCM-RL to solve the problem of coverage and connectivity is presented in Table 1.

3 Network model

In this paper, the sensor nodes are deployed to cover a hostile region up to a threshold level and maintain network connectivity. Figure 1 represents the network model of WSN. The sensor nodes are randomly deployed inside remote hostile region. At the time of deployment, the sensor nodes have uniform sensing and communication range. The hostile region also comprises obstacles which can disrupt the sensing and communication power of the sensor nodes. The nodes nearer to the base station communicates with the base station. This network model can be applicable in providing barrier coverage where scheduling algorithm divided the sensing task among the sensor nodes [40].

This paper proposes a node activity scheduling protocol where each sensor node autonomously learns its best action through reinforcement learning in each scheduling round. To do so, each sensor node is equipped with learning ability. In Algorithm 1, for a scheduling round the sensing range of the sensor nodes is customized according to the requirement and the resultant sensing range is marked as R_{final}. To achieve this objective a network is designed with the following assumptions.

1. A wireless sensor network is built to perform monitoring and surveillance task in remote hostile region. It comprises of a set of N static sensor nodes that are randomly deployed inside the sensing region of area A_s with high density.
2. Each sensor node S_i (i = 1,2,3,…,N) is well aware of its location using predefined localization algorithm [41].

Table 1 Parametric comparisons between protocols

Algorithm	Deployment type	Coverage redundancy check	Sensing range customization	Node activity scheduling	Coverage maintenance	Connectivity maintenance	Energy modelling	Machine learning technique
COORD [9]	Deterministic	✗	✗	✗	✔	✗	✔	Reinforcement learning (Q-learning)
PCP [37]	Uniform random	✔	✗	✔	✔	✗	✔	NA*
SALA [10]	Random	✗	✗	✔	✔	✗	✔	Learning automata
MNLAR [38]	Deterministic	✗	✔	✔	✔	✗	✔	Learning automata (pruning rule)
SPRC [39]	Random	✔	✗	✔	✔	✗	✔	NA
RLSSC [11]	Random	✔	✗	✔	✔	✗	✔	Reinforcement learning (Q-learning)
PCLA [12]	Random	✗	✗	✔	✔	✔	✔	Learning automata
CCA [5]	Random	✗	✗	✔	✔	✗	✔	NA
CCM-RL	Random	✔	✔	✔	✔	✔	✔	Reinforcement learning (Nash Q-learning)

*NA** not applicable

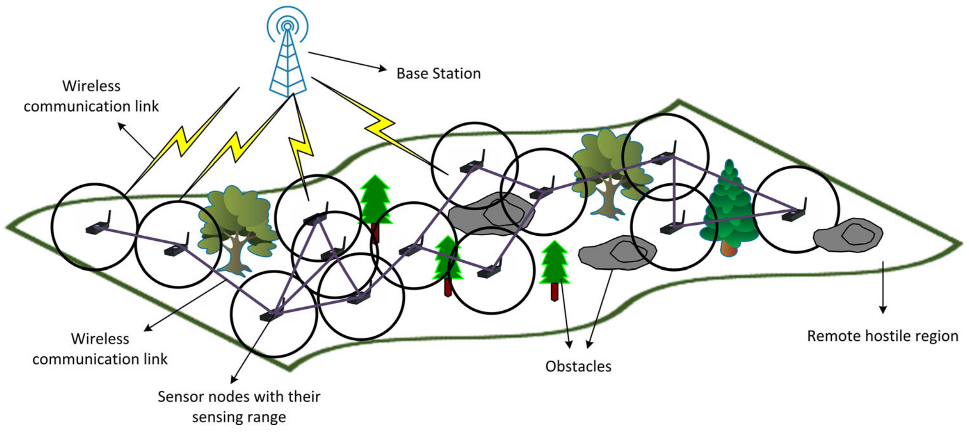
Base Station
Wireless communication link
Wireless communication link
Sensor nodes with their sensing range
Obstacles
Remote hostile region

Fig. 1 Wireless sensor network model

3. In the beginning of the algorithm, these sensor nodes have uniform sensing range R_s and communication range R_c. R_c is less than or twice to that of the sensing range ($R_c \leq 2R_s$). The coverage area CA_i provided by S_i is represented as a circle.
4. The sensing region of area A_s is divided into a G number of square grids of length l ($l = 2 \times R_s$) where $G = \frac{A_s}{l \times l}$.
5. Initially, the status of all deployed sensor nodes is active and have the same amount of energy E. This paper has incorporated simple energy consumption model [42] to process and communicate b-bit of packets over a distance d, $d \leq R_c$.

4 Reinforcement learning

The objective of reinforcement learning in WSN is to autonomously maintain the coverage and connectivity of randomly deployed sensor nodes. The best combination of state and actions is considered from a defined set of states and actions. Reinforcement learning is applicable because of the lack of trained data and enough expertise about the constraints and issues of WSN. Therefore, the sensor nodes learn what to do and how to form an optimum combination of state and action to maximize the numerical reward. The learning is dependent on two terms i.e. exploration and exploitation. Exploration is about going through the entire environment to capture more information that can help in reward maximization. On the other hand, exploitation is to utilize already known information to maximize the rewards. In WSN, the network designers have the crystal-clear idea about the desired output and RL achieves that desired output in terms of rewards.

In reinforcement learning, the environment gives states to the agents and then agents perform a suitable action to maximize the global reward. Reinforcement learning is modelled as a Markov Decision Process (MDP). The schematic diagram of reinforcement learning is shown in Fig. 2.

4.1 Nash Q-learning

Q-learning is a model-free reinforcement learning algorithm. The purpose of Q value in Q-learning is to learn an optimum policy for an agent to choose its best action that can maximize the overall reward value. The change in the state of the environment is entirely based on agent's current action. Q-learning algorithm has been proven to be a boon for single agent system as it has simple updating approach in which an agent starts with random initial values of $Q(s,a)$. The Q-values in any discrete time step t ($t = 1,2,3\ldots,n$) are updated using Eq. 1 and stored in the form of a matrix.

$$Q_{t+1}(s_t,a_t) = (1-\alpha_t)\,Q_t\,(s_t,a_t) + \left[r_t + \gamma \max_a\, Q_t(s_{t+1},a)\right] \tag{1}$$

In Eq. 1, $s \in S$ is set of states, a $\in$ A is set of actions and r_t is the reward while transiting from state s_t to s_{t+1}. α is learning rate ($0 < \alpha \leq 1$) and γ is discount factor ($0 < \gamma \leq 1$). Learning rate defines the extent of new information overriding the previous information. At $\alpha = 0$, agent learns nothing and $\alpha = 1$ agent considers only recent information. The small value of α ($\alpha > 0$) represents exploration. Therefore, in stochastic environment the values of $\alpha = 0.1$ is considered for exploration. Discount factor defines the importance of future rewards. The larger value of γ represents that agent is going to explore the entire environment.

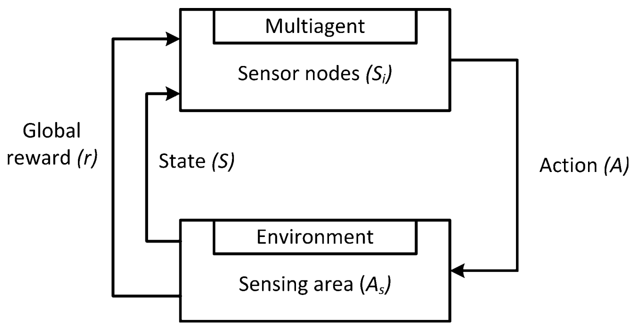

Fig. 2 Reinforcement learning and its environment

In real time there are many complex problems that require multiple agents to achieve global optimality. Therefore, extended Q-learning algorithms such as Distributed-Q learning [43], Friend-or-foe Q-learning [44], Nash-Q learning [23], Correlated-Q learning [45], Minimax-Q learning [46] and Optimal adaptive learning (OAL) [47] have been proposed to tackle the multiagent system. In multiagent reinforcement learning (MARL) the agents learn by interacting dynamically with their environment as shown in Fig. 2. The global state of the environment is governed by the actions of all agents. There is one challenge in MARL related to the behavior of agents. The agents can behave as cooperatively, competitively or neutrally [25].

Hu et al. [23] have extended the single agent Q-learning to multiagent systems for stochastic environment. The authors have proposed a Q-learning based distributed non-cooperative multiagent reinforcement learning algorithm where each agent simultaneously and independently tries to maximize the expected sum of discounted rewards. These discounted rewards are represented by Nash Q-values. Each agent follows Nash equilibrium strategies for the next time interval. For a multiagent system the very first step is to recognize the joint actions for a given state. The Q-function is $Q(s,a_1, a_2,\ldots,a_n)$ instead of $Q(s,a)$. The future rewards are based on agents' joint optimal strategy.

In Nash Q-learning algorithm, the agents are indexed as i. The first step of Nash Q-learning algorithm is to assume the random Q-values at $t = 0$. At time t the agent i identifies its current state and take action accordingly to get positive reward. Afterward, it identifies actions taken by other agents, their rewards and the next state s^*. At time t the agent i updates its Q-value using Eq. 2.

$$Q^i_{t+1}(s,a_1,a_2\ldots,a_n) = (1-\alpha_t)\;Q^i_t\,(s,a_1,a_2\ldots,a_n) + \alpha_t\left[r^i_t + \gamma NQ^i_t(s^*)\right] \tag{2}$$

where α is learning rate ($0 < \alpha \leq 1$) and γ is discount factor ($0 < \gamma \leq 1$). $NQ^i_t(s^*)$ is given by Eq. 3.

$$NQ^i_t(s^*) = \pi^1(s^*)\ldots\pi^n(s^*).\,Q^i_t(s^*) \tag{3}$$

Nash Q-learning algorithm incorporates joint actions and updates its Q-value based on Nash equilibrium strategy specified by all agents over current Q-values. Using Nash Q-learning the agents achieve global optimality faster than single agent Q-learning.

4.2 Formal definitions

Definition 1 An agent takes suitable action in discrete time steps $t = 0,1,2,\ldots,n$. Each sensor node autonomously acts as an agent.

Definition 2 The entire sensing area A_s is an environment. It provides state information to the multiagent (sensor nodes) resides inside the sensing area A_s. The environment takes the agent's current state and action as input, and returns the output as a numerical reward as shown in Fig. 2.

Definition 3 Local state S_L represents the same type of information. For coverage, there are two local states of a sensor node: *{S_{L1} = Coverage Redundancy, and S_{L2} = Isolate}*. The sensor node can be in any one of the states. For network connectivity maintenance, the two local states of a sensor node are *{CS_{L1} = Connected to 1-hop neighbor, and CS_{L2} = not connected}*.

Definition 4 Global state S_G represents the global objective of the multi-agent system. For coverage *{S_G = coverage rate}* and for connectivity *{CS_G = Connectivity among sensor nodes}*.

Definition 5 Action A is the set of all possible activities; an agent can perform ($A \in A_i$). For coverage the possible set of actions is: *{A_1 = active, A_2 = sleep, and A_3 = customize the sensing range}* and for connectivity the possible set of actions is: *{CA_1 = active, and CA_2 = hibernate}*. Actions are selected either using ϵ-greedy method or Boltzmann exploration [25]. ϵ-greedy is advantageous in choosing the best action with probability 1-ϵ or random action with probability ϵ, whereas, Boltzmann exploration uses a temperature parameter T to balance exploration and exploitation [24]. In this paper, the actions are selected using the ϵ-greedy method.

Definition 6 Reward r is the feedback by which the success or failure of an agent's selected action is measured. For example: the coverage rate provided by the active sensor nodes is reward. If the coverage rate C_r is greater than or equals to threshold coverage rate τ ($C_r \geq \tau$), then it is a positive reward otherwise negative reward. Reward can be categorized as local reward and global reward. Coverage area provided by an agent is stated as local reward and the coverage rate provided by the active number of sensor nodes in one scheduling round is stated as global reward. The illustration of local and global reward is shown in Fig. 3.

Definition 7 Coverage rate C_r is the total amount of area covered by active sensor nodes to that of the sensing area A_s.

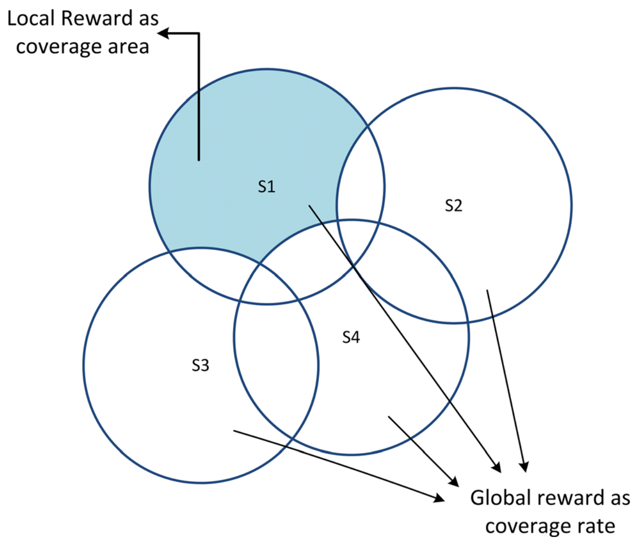

Fig. 3 Representation of local reward and global reward

$$C_r = \frac{\sum_{i=1}^{N_{active}} \pi R_{final\ i}^2}{A_s} \tag{4}$$

Definition 8 Network connectivity N_c states that there must exist at least one communication link between active sensor nodes.

Definition 9 For coverage maintenance, if the distance d between S_i and S_j is less than $2R_s$ ($d(S_i, S_j) < 2R_s$), then S_j is said to be the neighboring node S_{NN} of S_i. For connectivity maintenance, if the distance d between S_i and S_j is less than $2R_c$ ($d(S_i, S_j) < 2R_c$), then S_j is said to be the 1-hop neighboring node of S_i.

Definition 10 Convergence time C_T is the time taken by the learning algorithm to achieve its global optimality.

Definition 11 Active node ratio is defined as the total number of active nodes N_{active} to that of the total number of deployed sensor nodes N.

Definition 12 Policy π is state action mapping probability. It is an approach that an agent applies to identify the next best action based on current state.

Definition 13 Threshold value sets a level in achieving the particular goal. In this paper, the threshold value τ represents the minimum value set for achieving the coverage rate. The active sensor nodes must provide coverage up to the threshold level.

5 Coverage connectivity maintenance based on reinforcement learning (CCM-RL) protocol

This section presents the complete description of the proposed CCM-RL protocol. This protocol presents reinforcement learning based an energy-efficient node activity scheduling algorithm for randomly deployed sensor nodes. CCM-RL helps in maintaining the desired coverage rate and connectivity provided by the active sensor nodes in each scheduling round. Nash Q-learning [23], an algorithm for multiagent reinforcement learning is applied for node activity scheduling. A WSN comprises N number of static sensor nodes where each sensor node acts as an agent. Due to N number of sensor node, a WSN is defined as a multiagent system. It is difficult and time-consuming to achieve a global optimal path with a single agent Q-learning algorithm. Therefore, Nash Q-learning algorithm is applied for minimum convergence time. Along with activity scheduling of nodes, CCM-RL also considers two other issues of random deployment i.e. coverage redundancy and partial coverage.

5.1 CCM-RL protocol description

The CCM-RL protocol comprises of two phases: (1) Learning phase for coverage maintenance, and (2) Learning phase for connectivity maintenance. The complete description of these two phases is presented in Sects. 5.1.1. and 5.1.2.

5.1.1 Reinforcement learning for coverage maintenance

The learning process in a sensor node starts immediately after the random deployment of the sensor nodes. The agents explore the entire environment. At time $t = 0$, the learning rate α, discount factor γ, and random Q-values are initialized. The learning process starts at time t from any randomly selected sensor node S_i. S_i identifies all of its neighboring nodes S_{NN} and observes its local state S_L ($S_L \in S_{L1}, S_{L2}$). $S_L \in S_{L1}$ represents that S_i is in the coverage redundancy state. S_i then computes the amount of coverage redundancy among its neighboring nodes using Euclidean distance formula. Afterward, S_i choose the best action A *{A_1 = active, A_2 = sleep, and A_3 = customize the sensing range}* and ($A \in A_1, A_2, A_3$) to mitigate the coverage redundancy. If the selected action is either A_1 or A_3 then the coverage area CA_i $\left(CA_i = \pi {R_{final}}_i^{\,2}\right)$ provided by S_i is assigned as positive reward. $S_L \in S_{L2}$ represents that S_i is in an isolated state. S_i need not to perform any action and the coverage area CA_i $\left(CA_i = \pi {R_s}_i^{\,2}\right)$ of S_i is assigned as positive reward. The Q-values are updated iteratively using Eq. 2 to achieve global optimality. Figure 4 represents the flow chart of the above description.

If the global reward is greater than the threshold level after scanning N sensor nodes, then the learning is said to be convergent and the time required to achieve this is known as convergence time. Otherwise, the entire process is repeated to achieve the goal. To achieve the coverage rate C_r ($C_r \geq \tau$) is the global optimality of the learning algorithm. The output of this phase is the total number of active sensor nodes and their coverage rate. It becomes the input for connectivity maintenance phase as shown in Fig. 5.

The pseudo code of node activity scheduling for coverage maintenance is presented in Algorithm 1. This algorithm presents the step by step procedure for one scheduling round. The similar steps are executed for the rest of the scheduling rounds. The sensor nodes, which are active in one scheduling round will be in sleep mode for the next scheduling round. At the beginning of the algorithm all important parameters are initialized. Each sensor node is assigned with a unique ID *S_ID*. The S_i sensor nodes with unique IDs are stored in *active_sensors* and *uncovered_set*. Initially, all sensor nodes are active. At the beginning of a scheduling round, a random sensor node S_i is selected and all of its neighbors are identified. The neighboring nodes S_{NN} are stored in the ascending order of their distance from S_i in a list. S_i picks the first node from the list. There exists coverage redundancy because of the minimum distance between S_i and S_{NN}. The algorithm then computes a variable to customize the sensing range by x units. This helps in mitigating the coverage redundancy. If the sum of coverage area provided by S_i and S_{NN} after customizing their sensing range by x units is larger than the coverage provided by $max(coverage(S_i), coverage(S_{NN}))$, then reduce the sensing range of both the sensor nodes by x units. S_i and S_{NN} are kept in active mode. Otherwise, put the sensor node that provides minimum coverage in sleep mode. This process is repeated for all deployed sensor nodes. At the end of the scheduling round, the coverage rate is computed. If the coverage rate provided by active sensor nodes is greater than the threshold coverage rate τ, then the scheduling round is saved. Otherwise, Algorithm 1 is executed again with any other randomly selected sensor node.

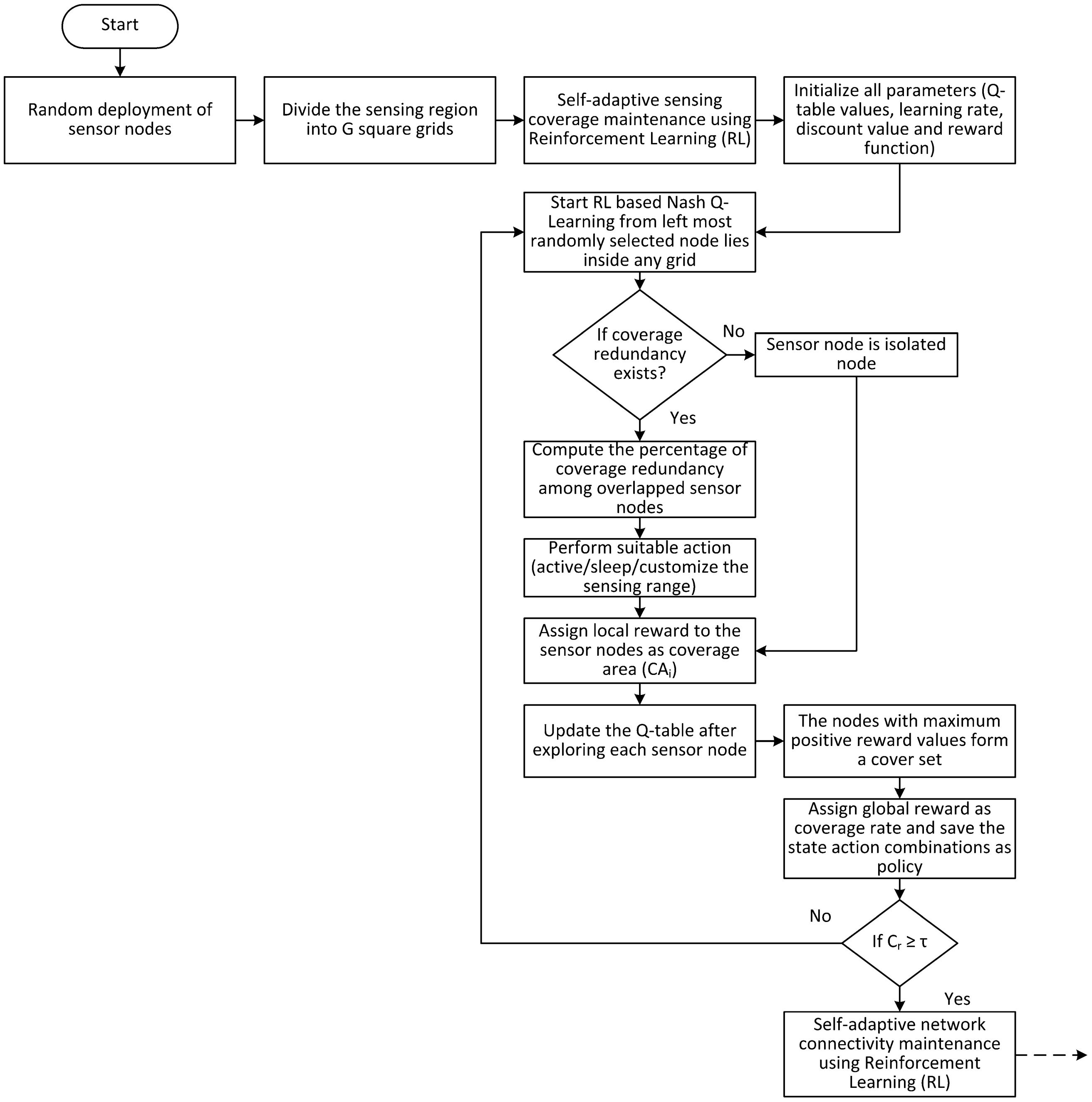
Start
Random deployment of sensor nodes
Divide the sensing region into G square grids
Self-adaptive sensing coverage maintenance using Reinforcement Learning (RL)
Initialize all parameters (Q-table values, learning rate, discount value and reward function)
Start RL based Nash Q-Learning from left most randomly selected node lies inside any grid
If coverage redundancy exists?
No
Sensor node is isolated node
Yes
Compute the percentage of coverage redundancy among overlapped sensor nodes
Perform suitable action (active/sleep/customize the sensing range)
Assign local reward to the sensor nodes as coverage area (CA$_i$)
Update the Q-table after exploring each sensor node
The nodes with maximum positive reward values form a cover set
Assign global reward as coverage rate and save the state action combinations as policy
No
If $C_r \geq \tau$
Yes
Self-adaptive network connectivity maintenance using Reinforcement Learning (RL)

Fig. 4 Learning phase for coverage maintenance using Nash Q-learning

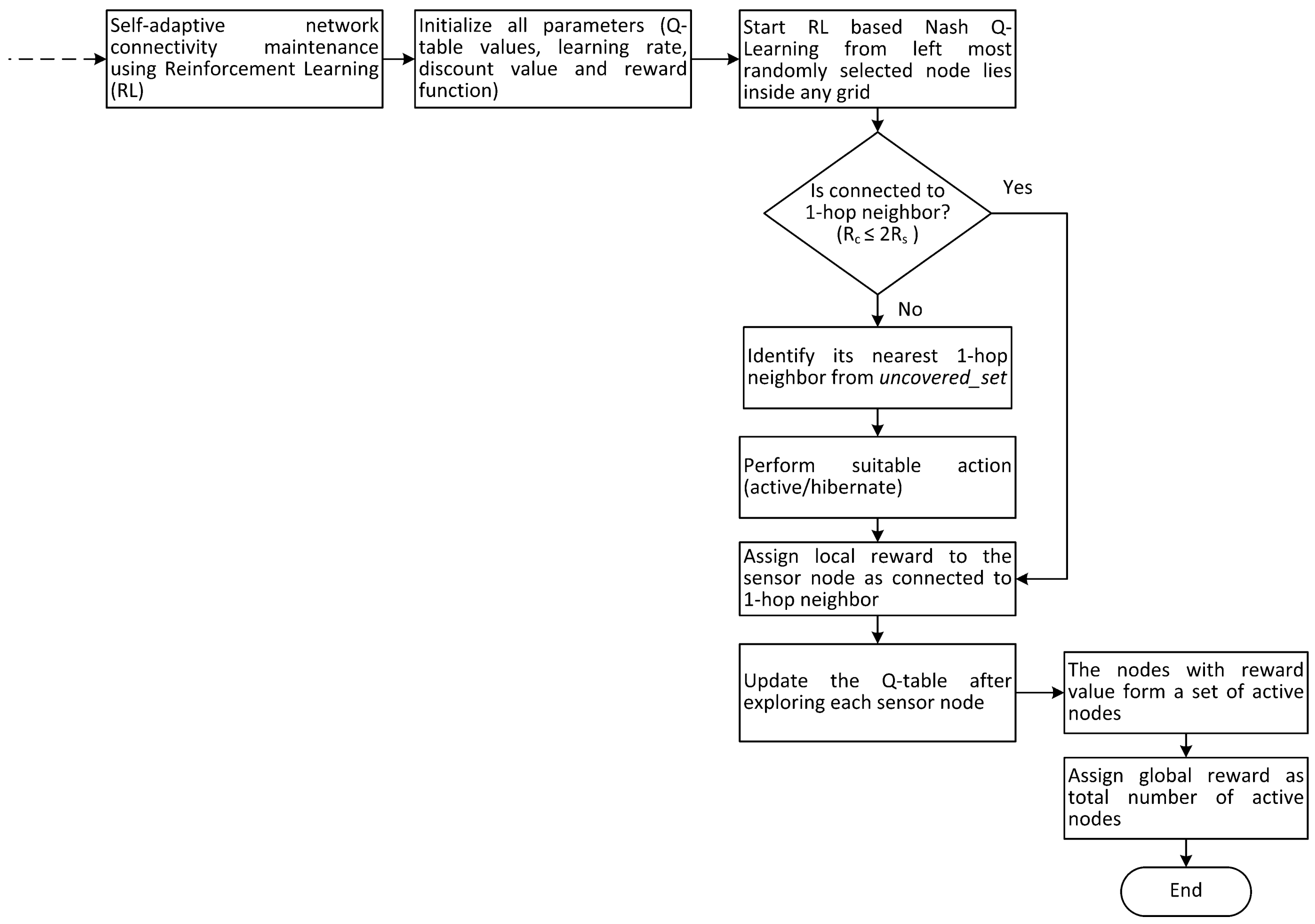
Self-adaptive network connectivity maintenance using Reinforcement Learning (RL)
Initialize all parameters (Q-table values, learning rate, discount value and reward function)
Start RL based Nash Q-Learning from left most randomly selected node lies inside any grid
Is connected to 1-hop neighbor? ($R_c \leq 2R_s$)
Yes
No
Identify its nearest 1-hop neighbor from *uncovered_set*
Perform suitable action (active/hibernate)
Assign local reward to the sensor node as connected to 1-hop neighbor
Update the Q-table after exploring each sensor node
The nodes with reward value form a set of active nodes
Assign global reward as total number of active nodes
End

Fig. 5 Learning procedure for connectivity maintenance using Nash Q-learning

Algorithm 1: Node activity scheduling for coverage maintenance

```
1.  Initialize:
        $N \leftarrow$ total number of deployed sensor nodes
        $S_i \leftarrow$ index of total number of deployed sensor nodes          $1 \le i \le N$
        $S\_ID \leftarrow$ IDs list of all deployed sensor nodes
        $S_i \in S\_ID$                                                       $1 \le S\_ID \le N$
        $R_s \leftarrow$ sensing range of $S_i$
2.  begin
3.  for each $S_i \in S\_ID$
4.    $active\_sensors \leftarrow S\_ID$
5.    $uncovered\_set \leftarrow S\_ID$
6.    $action[S_i] \leftarrow 'active'$, for $\forall\ i = 1,2, \ldots,N$     // set action of all sensor nodes as active
7.    $iteration \leftarrow 0$
8.    $i \leftarrow S_i$
9.    while (number of sensor nodes in $uncovered\_set > 0$) do
10.     if $i = 0$
11.       $i \leftarrow$ random sensor node from $active\_sensors$
12.     endif
13.     $uncovered\_set \leftarrow uncovered\_set - [i]$     // remove sensor node $i$ from $uncovered\_set$
14.     for sensor $S_j$ in $active\_sensors$
15.       if $(S_i \neq S_j)$ and $distance(S_i, S_j) < (R_s[S_i] + R_s[S_j])$
16.           $n\_node[S_i] \leftarrow S_j$                    // append $S_j$ to the neighbor list of $S_j$
17.           $n\_distance[S_i] \leftarrow distance[S_i, S_j]$    // append distance of neighbor $S_j$ in a list
18.       endif
19.     if no sensor node in the neighbor set of $S_i$, then
20.       $i \leftarrow 0$
21.     else compute $S_{NN}$ nearest neighbor sensor nodes of $S_i$, by using $n\_distance$
22.       compute $x$, a customization in sensing range of $S_i$ and $S_{NN}$ to remove the coverage
          redundancy
23.       if the sum of coverage area provided by $S_i$ and $S_{NN}$ after customization in their sensing
          range by $x$ is larger than the coverage provided by $max(coverage(S_i), coverage(S_{NN}))$, then
24.           $R_s[S_i] \leftarrow R_s[S_i] - x$          // customize $R_s$ of $S_i$
25.           $R_s[S_{NN}] \leftarrow R_s[S_{NN}] - x$    // customize $R_s$ of $S_{NN}$
26.           $uncovered\_set \leftarrow uncovered\_set - S_{NN}$   // remove $S_{NN}$ from $uncovered\_set$
27.       else
28.           identify sensor with minimum coverage $S_{min} \leftarrow min(R_s[S_i, S_{NN}])$
29.           $action[S_{min}] \leftarrow 'sleep'$                  // put $S_{min}$ in sleep mode
30.           $active\_sensors \leftarrow active\_sensors - S_{min}$    // remove $S_{min}$ from $active\_sensors$
31.           $uncovered\_set \leftarrow uncovered\_set - S_{min}$    // remove $S_{min}$ from $uncovered\_set$
32.       endif
33.     endif
34.     if multiple sensor nodes are neighbors of $S_i$ then
35.       $i \leftarrow$ nearest neighbor of $S_i$ using $n\_node$ list
36.     endif
37.     $iteration \leftarrow iteration + 1$
38.   end while
39.   compute coverage rate
40.   for sensor $S_i$ with active state
41.     $Coverage\ rate \leftarrow Coverage\ rate + S\_coverage[S_i]$
42.     $cover\_set \leftarrow$active senor nodes
43. end
```

5.1.2 Reinforcement learning for connectivity maintenance

The total number of active sensor nodes acquired from Algorithm 1 is the input for connectivity maintenance phase. The main purpose behind this phase is to maintain at least one communication link between active sensor nodes and mitigate the network partitioning. Figure 5 presents the flow chart for connectivity maintenance.

At time $t = 0$, Q-values, learning rate α, and discount factor γ are initialized. At time t, the learning process is started with any randomly selected sensor node from *active_sensors*. Identify its 1-hop neighbors and perform a suitable action from the action set CA $\{CA_1 = active$, and $CA_2 = hibernate\}$ and ($CA \in CA_1, CA_2$). If the active sensor node does not have any 1-hop neighbor, then identify the most eligible $S_{eligible}$ node from *uncovered_set* and then perform a suitable action from the action set CA. The total number of active sensor nodes is the global reward obtained from this phase. Algorithm 2 represents the pseudo code of node activity scheduling for connectivity maintenance.

Algorithm 3 presents the pseudocode for coverage connectivity maintenance based on reinforcement learning CCM-RL. The rewards are observed using Algorithm 1 and Algorithm 2 and then Q-table is updated using Eq. 2. The Nash Q-learning for multiagent system is explained in Sect. 4.1. The updating of Q-table is an iterative process to achieve the global optimality of rewards.

Algorithm 2: Node activity scheduling for connectivity maintenance

```
1.   Input:
         $S_{active}$← total number of active sensor nodes from Algorithm 1
2.   begin
3.   while (number of sensor nodes in $S_{active} > 0$) do
4.       $S_r$← randomly select a sensor node from $S_{active}$
5.       $S_r^{NN}$← identify 1-hop neighbors of $S_r$
6.       if (no senor in $S_r^{NN}$), then
7.           identify the most eligible node $S_{eligible}$ from uncocered_set      // from Algorithm 1
8.           action[$S_{eligible}$] ←'hibernate'
9.           $S_{active}$←$S_{active}$ +1
10.          uncovered_set←uncovered_set - $S_{eligible}$
11.          active_sensors←active_sensors+1
12.      endif
13.  end while
14.  end
15.  Output: total number of active_sensors
```

Algorithm 3: CCM-RL Algorithm

```
1. Initialize number of agents $A_i$ and their states and actions set
2. initialize $Q_{Ai}(s,a_1, a_2,\ldots,a_n)$, $\alpha$, $\gamma$ and $iteration\_count = 0$ at time $t = 0$
3. repeat until the global optimality have been achieved
4. for each agent $A_i$ at time $t$ $(t=1,2,3\ldots,n)$
5.     select greedy optimal action $A$ in a state which maximizes the accumulated reward
6.     observe the reward using Algorithm 1 and Algorithm 2
7.     update Q-table for agent $A_i$ using equation 2
8.     evaluate the updated Q-table using global reward $r$
9. end for
```

5.2 Computational complexity

Computational complexity is defined as the number of resources and computation time required to execute an algorithm. The computational complexity of Algorithm 1 and Algorithm 2 is based on the number of the deployed sensor nodes N. The number of sensor nodes are the prime metric for coverage and connectivity maintenance. The computational complexity of Algorithm 1 is $O(N^3)$ and Algorithm 2 is $O(N^2)$ where N is the number of deployed sensor nodes. Section 6.6 presents the convergence time of Algorithm 1, Algorithm 2 and Algorithm 3 with respect to the number of sensor nodes.

6 Performance evaluation

In this section, the performance of the CCM-RL is evaluated through extensive simulations performed on OMNeT++ and Python. The simulations are performed to ensure the accuracy and reliability of CCM-RL protocol. The performance of CCM-RL is evaluated using seven parameters for both small and large size WSN respectively, such as: coverage rate, number of cover set formation, average number of active sensor nodes for coverage, average number of active sensor nodes for connectivity maintenance, number of iterations to achieve global optimality, percentage of wrong decisions, and convergence time. The proposed protocol is compared with Coverage Contribution Area (CCA) [5], Partial Coverage with Learning Automata (PCLA) [12] and Probabilistic Coverage Protocol (PCP) [37]. The three parameters used to compare the performance of CCM-RL with CCA, PCLA, and PCP respectively, such as: average number of active sensor nodes, coverage rate, and average power consumption in each cover set. The Figs. 16, 17, and 18 show that CCM-RL performs better as compared to CCA, PCLA, and PCP. Table 2 presents the list of parameters and their values used for simulation.

6.1 Random deployment of sensor nodes and implementation of CCM-RL

A set of 100 sensor nodes having sensor IDs S1, S2,…, S100 is randomly deployed inside 2500 m^2 sensing region. These sensor nodes have uniform sensing range $R_s = 15$ m. The coverage area of each sensor node is considered as a circle. Figure 6 presents the deployment of these sensor nodes. Figure 7 shows the resultant active number of sensor nodes in one scheduling round after the implementation of Algorithm 1, Algorithm 2, and Algorithm 3. It shows that out of 100 only 25 sensor nodes are active. These nodes provide coverage rate up to a threshold level and also ensure network connectivity. The activation of only 25% of nodes in one scheduling round reduces the energy consumption rate. Moreover, Fig. 7 shows that after the execution of algorithms the sensing range of S17, S20, S21, S24, S36, S37, S38, S44, S47, S49, S61, S73, S77, S78, S81, S87, S90 and S96 is customized to mitigate the coverage redundancy among sensor nodes. The customization in sensing power also reduces energy consumption for processing and communicating the data bits. The sensing range of S10, S41, S64, S67, S69, S76 and S97 remains same.

6.2 Analysis of number of cover sets, average number of active sensor nodes, and coverage rate for small and large scale WSN

The sets of 50, 100, 150 and 200 sensor nodes are randomly deployed in the small sensing region of area 100*100 m^2. Figure 8 shows the results respectively,

Table 2 List of parameters

Name of the parameter	Values	
	For small network size	For large network size
Number of sensor nodes (N)	50, 100, 150, and 200	250, 500, 1000, and 1500
Sensing area (A_s)	100*100 m^2	1000*1000 m^2
Sensing range (R_s)	10 m, 20 m, and 25 m	25 m, 30 m, and 35 m
Communication range (R_c)	20 m, 40 m, and 50 m	50 m, 60 m, and 70 m
Threshold value (τ)	0.65	
Total energy (E)	100 J	
Power consumed by active sensor nodes	57 mA	
Power consumed by sleep sensor nodes	0.40 µA	
Learning rate (α)	0.1	
Discount factor (γ)	0.7	
Maximum time limit for each scheduling round	20 min	

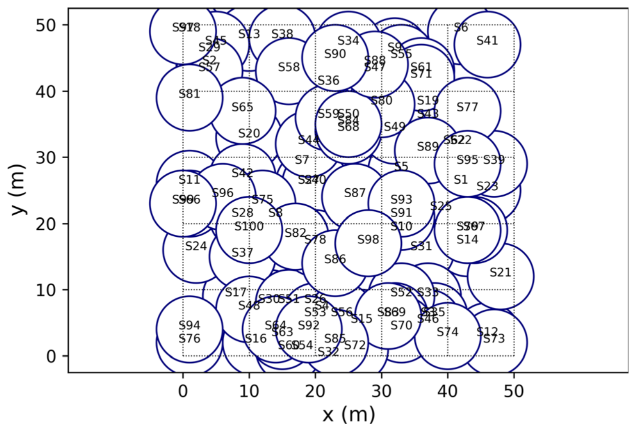

Fig. 6 Deployment of 100 sensor nodes inside 2500 m^2 sensing region

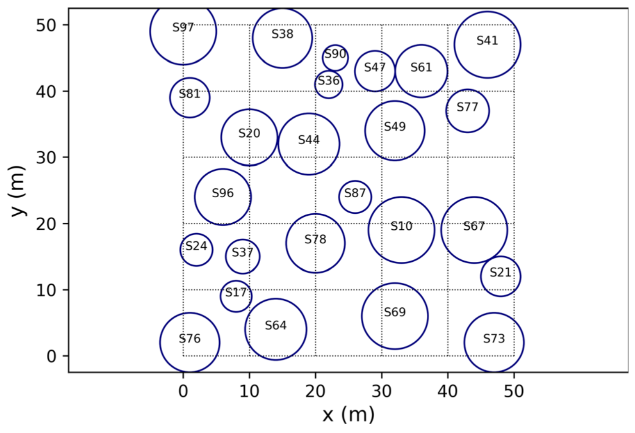

Fig. 7 Active number of sensor nodes after the implementation of Algorithm 1, 2, and 3

which are number of cover sets formation, average number of active sensor nodes in one scheduling round, and coverage rate for small scale sensing region. These results are evaluated for 10 m, 20 m, and 25 m sensing range. Similarly, Fig. 9 shows the results for large scale WSN. The sets of 250, 500, 1000, and 1500 sensor nodes are randomly deployed in large sensing area 1000*1000 m^2. The results are evaluated for three different sensing ranges, which are 25 m, 30 m, and 35 m. Figures 8(a), 9(a), 8(c), and 9(c) show that there is an increase in cover sets formation and coverage rate with the increase in the sensing range. Figures 8(b) and 9(b) show that the average number of active sensor nodes reduces with the increase in the sensing range for a fixed number of sensor nodes. This analysis helps in estimating the number of sensor nodes with homogenous sensing range require to cover a sensing region up to a threshold level.

6.3 Average number of active senor nodes for network connectivity

It is not necessary that the average number of active sensor nodes that provide threshold coverage rate must have connectivity among themselves. Algorithm 2 has been proposed to analyze and maintain the connectivity among the sensor nodes. Figure 10 represents the results that show the average number of active sensor nodes require to preserve network connectivity for different number of sensor nodes. The total number of active sensor nodes increases for connectivity maintenance as compared to coverage maintenance. For a fixed sensing area 1000*1000 m^2 the average number of active sensor nodes decreases with the increase in their sensing range.

6.4 Analysis of number of iterations for small and large scale WSN

This section explains about the number of iterations that have been occurred to achieve the global reward for given number of fixed sensor nodes. Figures 11 and 12 show the

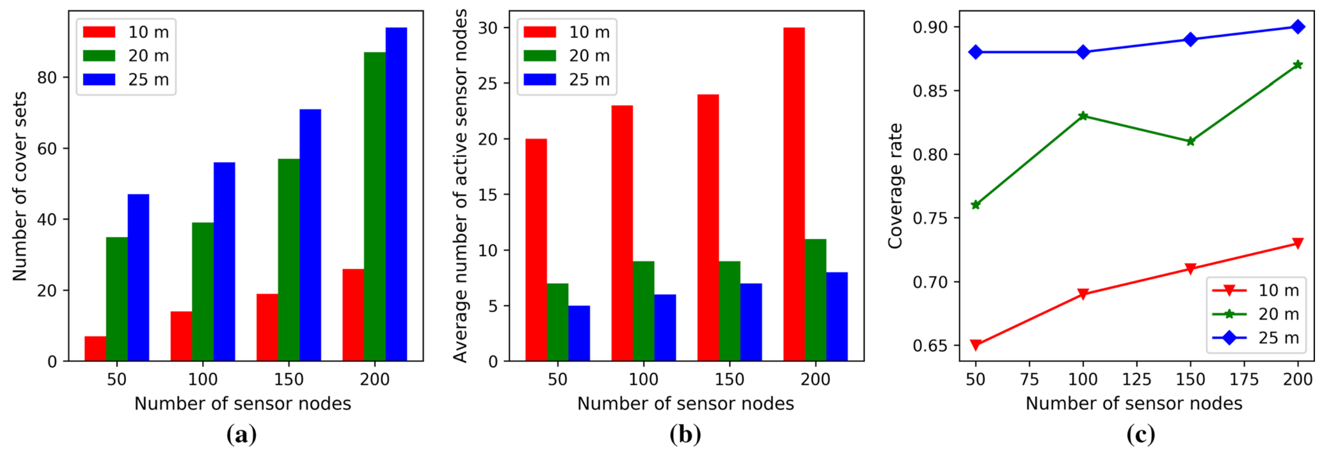

Fig. 8 Analysis of parameters **a** number of cover sets, **b** average number of active sensor nodes, and **c** coverage rate for fixed number of sensor nodes and small sensing area 100*100 m^2

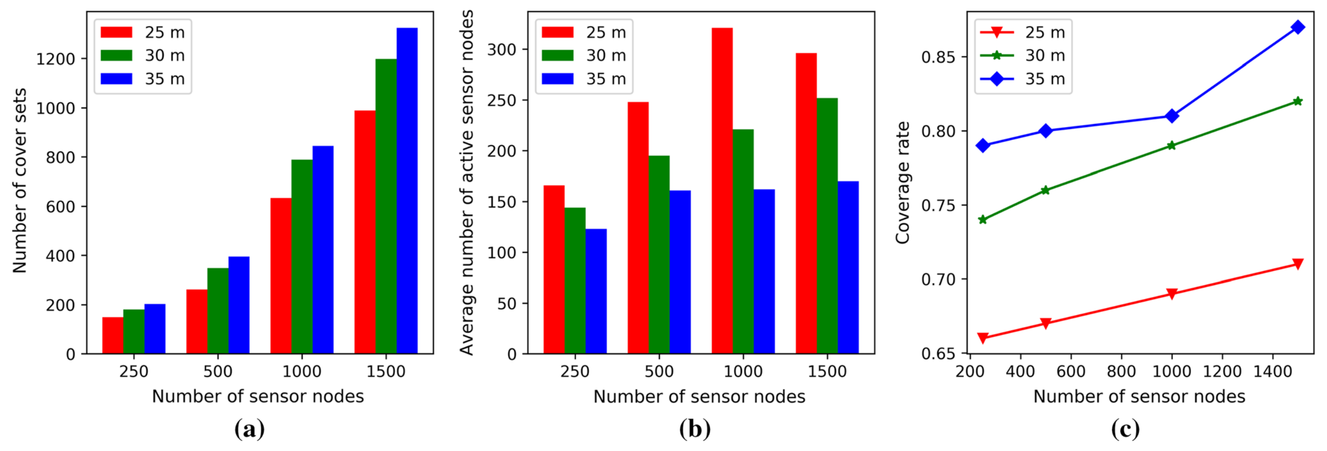

Fig. 9 Analysis of parameters **a** number of cover sets, **b** average number of active sensor nodes, and **c** coverage rate for fixed number of sensor nodes and large sensing area 1000*1000 m^2

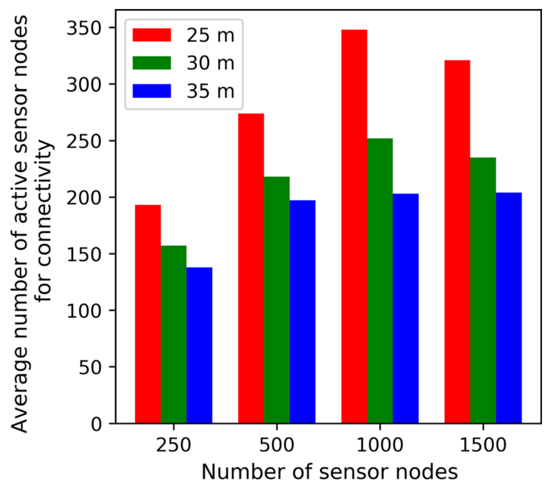

Fig. 10 Average number of active sensor nodes for network connectivity versus number of sensor nodes

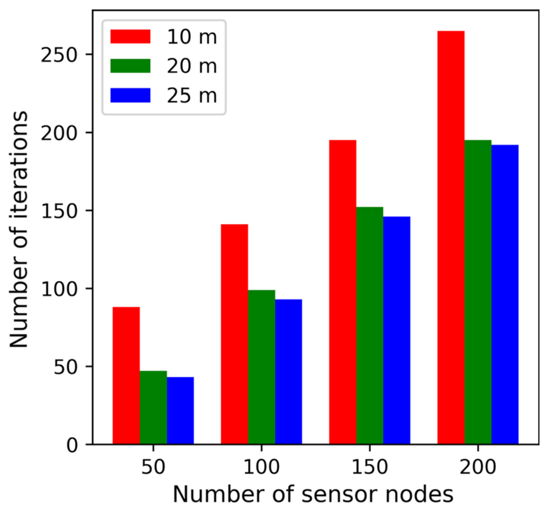

Fig. 11 Number of iterations occurred for fixed number of sensor nodes deployed inside small sensing area 100*100 m^2

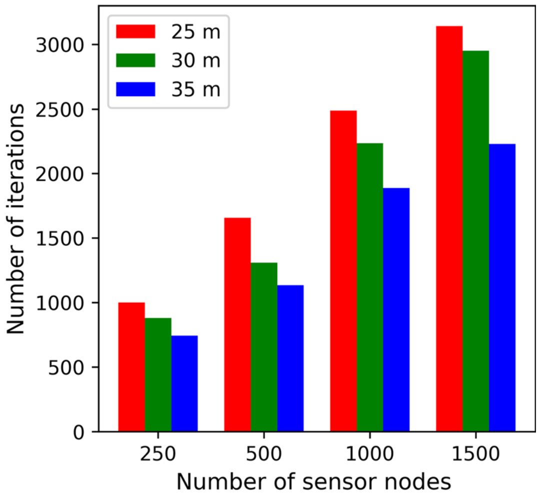

Fig. 12 Number of iterations occurred for number of sensor nodes deployed inside large sensing area 1000*1000 m^2

number of iterations that has taken place for small and large scale WSN. The parametric values of small and large scale WSN are same as considered in Sect. 6.2. The graphical results in Figs. 11 and 12 represent that for a fixed number of sensor nodes the number of iterations decreases with the increase in their sensing range.

6.5 Percentage of wrong decision versus number of senor nodes

Reinforcement learning is an iterative process of learning to achieve global rewards. In each learning step the agent generates the output as a reward. The reward can be either a positive reward or negative reward. Only those rewards are considered which help in achieving the global optimality and remaining rewards are considered as wrong decisions. This section represents the percentage of wrong decisions occur with respect to the total number of sensor nodes. Figure 13 shows that for a fixed large sensing area 1000*1000 m^2 the percentage of wrong decisions decreases with the increase in the number of sensor nodes and their sensing range. The results are evaluated for 25 m, 30 m, and 35 m sensing range.

6.6 Convergence time versus number of sensor nodes

Convergence time is the time required by Algorithm 1, 2 and 3 to achieve global optimality. Figure 14 shows the convergence time with respect to the total number of deployed sensor nodes. This time includes the time consumed for activity scheduling of sensor nodes, coverage rate computation, and connectivity maintenance. It shows that for a fixed sensing area 1000*1000 m^2 the convergence time is directly proportional to the number of sensor nodes. The convergence time increases with the increase in the number of sensor nodes.

6.7 Effect of learning rate (α) on convergence time

The learning rate reflects the efficiency of an agent to learn the optimum policy that helps in achieving the global optimality. The value of learning rate lies between 0 and 1 ($0 < \alpha \leq 1$). The larger value of α minimizes the learning ability of an agent. The results generated with large value of α are not correct. In this paper, the learning rate $\alpha = 0.1$ is considered. Figure 15 shows the effect of learning rate on convergence time (in seconds). This represents that the convergence time decreases with the increase in the value of learning rate. Therefore, the smaller value of α helps in achieving better learning outcomes.

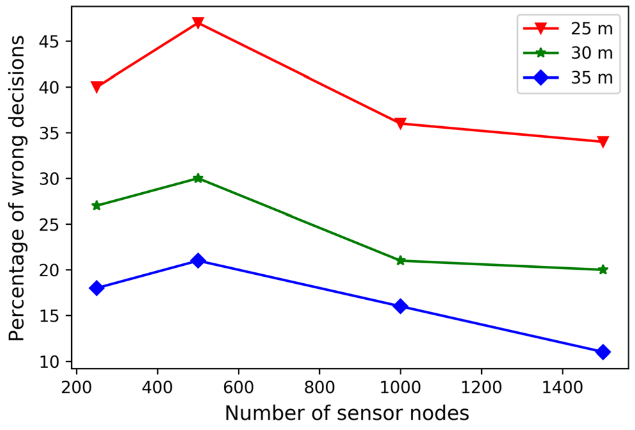

Fig. 13 Percentage of wrong decision versus number of sensor nodes for large sensing area 1000*1000 m^2

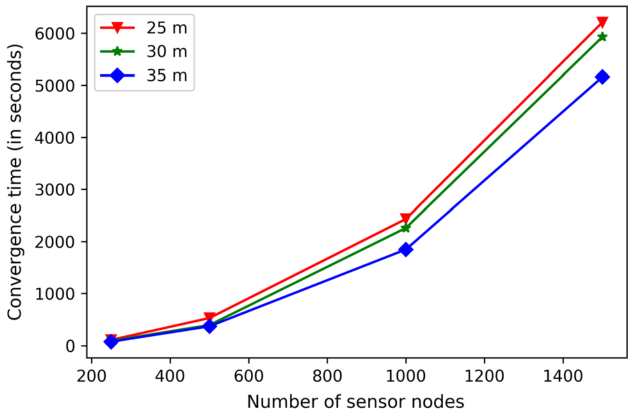

Fig. 14 Convergence time versus number of sensor nodes for large sensing area 1000*1000 m^2

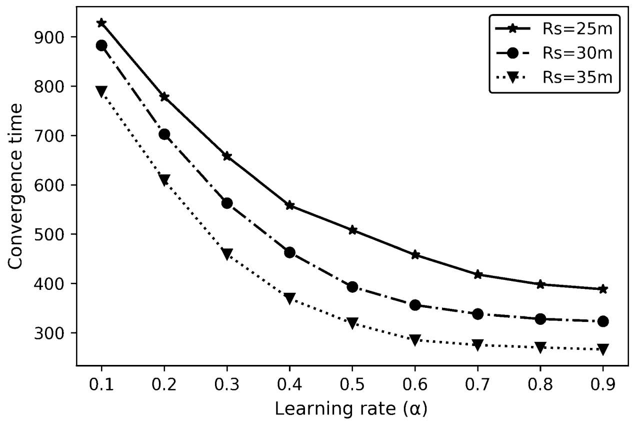

Fig. 15 Effect of learning rate (α) on convergence time

6.8 Comparison of CCM-RL with CCA, PCLA, and PCP

The proposed CCM-RL protocol is a node activity scheduling protocol. The main objective of this protocol is to enable the deployed sensor nodes learn their best action that can provide maximum coverage rate and maintain network connectivity. Node activity scheduling activates a subset of sensor nodes, instead of the entire set in each scheduling round. CCM-RL has proposed three algorithms to fulfill this objective. However, the accuracy of the algorithm can only be proven after its comparison with already existed protocols. Therefore, the performance of CCM-RL is compared to CCA, PCLA and PCP on the basis of three parameters respectively, which are average number of active sensor nodes, coverage rate, and average power consumption in each cover set (in Watts). Figure 16 shows that CCM-RL enables a lesser number of sensor nodes to be active in one scheduling round as compare to CCA, PCLA, and PCP. The lesser number of active sensor nodes in each scheduling round increases the number of cover sets and network lifetime.

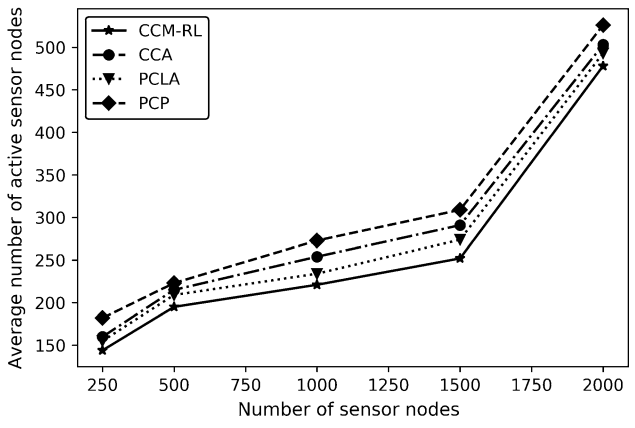

Fig. 16 Average number of active sensor nodes versus number of sensor nodes

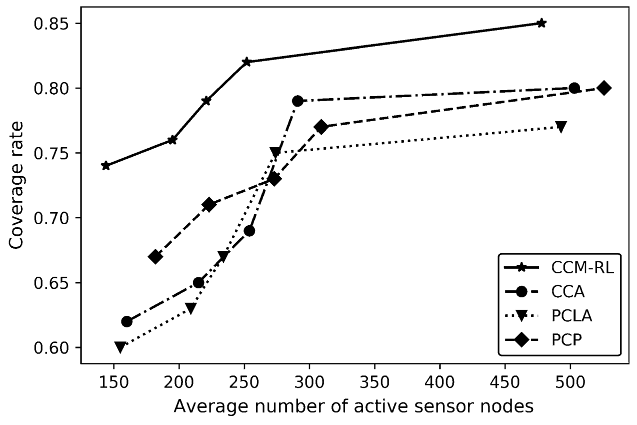

Fig. 17 Coverage rate versus average number of active sensor nodes

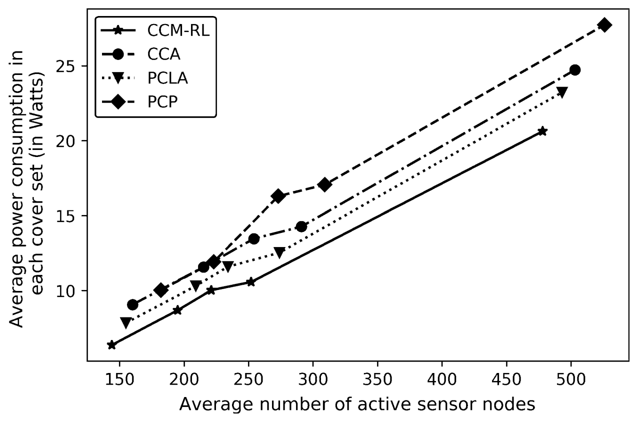

Fig. 18 Average power consumed by active sensor nodes in each cover set

Figure 17 compares the performance of CCM-RL in terms of coverage rate. It shows that with lesser number of active sensor nodes CCM-RL provides maximum coverage rate. Moreover, CCM-RL has reduced the coverage redundancy to mitigate the problem of redundant packet generation. Figure 18 shows that CCM-RL consumes less power for processing and communicating the data bits. The less power consumption enhances the network lifetime. Hence, WSN can function for larger duration as compared to CCA, PCLA and PCP. These comparisons show that CCM-RL is efficient in terms of maximizing the coverage rate and minimizing the power consumption.

7 Conclusion

Coverage and connectivity are two fundamental issues of wireless sensor network. These issues become more critical for randomly deployed sensor nodes. Random deployment is a biased deployment strategy of sensor nodes, which imposes many challenges such as uneven deployment, coverage redundancy, partial coverage, and unwanted resource consumption. To resolve these issues, this paper has proposed a Reinforcement Learning based Coverage and Connectivity Maintenance (CCM-RL) protocol. This protocol autonomously and intelligently chooses the best activity for a sensor node that can maximize the overall coverage rate and maintain network connectivity. The sensor nodes learn their best action using Nash Q-learning. This protocol also provides a mechanism to reduce the coverage redundancy among sensor nodes by customizing their sensing range. CCM-RL's performance assessments demonstrate its accuracy and reliability with respect to the average number of active sensor nodes, number of cover sets, coverage rate, and energy consumption.

Nash Q-learning is the learning algorithm for multiagent. It enables the agents of a multiagent system to learn their best action in a particular state. There are many other learning algorithms for multiagent systems in machine learning such as Distributed-Q learning, Minimax-Q learning, Friend-or-foe Q learning, Correlated-Q learning and Optimal adaptive learning (OAL). The future scope of this paper is to implement these algorithms for teaching the agents and compare the performance in terms of convergence time, rewards, and global optimality.

References

1. Wang, X., Xing, G., Zhang, Y., Lu, C., Pless, R., & Gill, C. (2003). Integrated coverage and connectivity configuration in wireless sensor networks. In *Proceedings of the 1st international conference on embedded networked sensor systems* (pp. 28–39).
2. Zhang, H., & Hou, J. C. (2005). Maintaining sensing coverage and connectivity in large sensor networks. *Ad Hoc and Sensor Wireless Networks, 1,* 89–124.
3. Onur, E., Ersoy, C., Deliç, H., & Akarun, L. (2007). Surveillance wireless sensor networks: Deployment quality analysis. *IEEE Network, 21*(6), 48–53.
4. Mini, S., Udgata, S. K., & Sabat, S. L. (2014). Sensor deployment and scheduling for target coverage problem in wireless sensor networks. *IEEE Sensor Journal, 14*(3), 636–644.
5. Yu, J., Wan, S., Cheng, X., & Yu, D. (2017). Coverage contribution area based k-coverage for wireless sensor networks. *IEEE Transactions on Vehicular Technology, 66*(9), 8510–8523.
6. Jiang, B., Ravindran, B., & Cho, H. (2012). Probability-based prediction and sleep scheduling for energy-efficient target tracking in sensor networks. *IEEE Transactions on Mobile Computing, 12*(4), 735–747.
7. Zorbas, D., Glynos, D., Kotzanikolaou, P., & Douligeris, C. (2010). Solving coverage problems in wireless sensor networks using cover sets. *Ad Hoc Networks, 8*(4), 400–415.
8. Mohamadi, H., Ismail, A. S., & Salleh, S. (2014). Solving target coverage problem using cover sets in wireless sensor networks based on learning automata. *Wireless Personal Communications, 75*(1), 447–463.
9. Seah, M. W. M., Tham, C. K., Srinivasan, V., & Xin, A. (2007). Achieving coverage through distributed reinforcement learning in wireless sensor networks. In *Proceedings of the 3rd international conference on intelligent sensors, sensor networks and information* (pp. 425–430).
10. Esnaashari, M., & Meybodi, M. R. (2010). A learning automata based scheduling solution to the dynamic point coverage problem in wireless sensor networks. *Computer Networks, 54*(14), 2410–2438.
11. Chen, H., Li, X., & Zhao, F. (2016). A reinforcement learning-based sleep scheduling algorithm for desired area coverage in solar-powered wireless sensor networks. *IEEE Sensors Journal, 16*(8), 2763–2774.
12. Mostafaei, H., Montieri, A., Persico, V., & Pescapé, A. (2017). A sleep scheduling approach based on learning automata for WSN partial coverage. *Journal of Network and Computer Applications, 80,* 67–78.
13. Esnaashari, M., & Meybodi, M. R. (2010). Data aggregation in sensor networks using learning automata. *Wireless Networks, 16*(3), 687–699.
14. Lu, Y., Zhang, T., He, E., & Comşa, I. S. (2018). Self-learning-based data aggregation scheduling policy in wireless sensor networks. *Journal of Sensors, 2018*, 9647593.
15. Shahina, K., & Vaidehi, V. (2018). Clustering and data aggregation in wireless sensor networks using machine learning algorithms. In *Proceedings of the 2018 international conference on recent trends in advance computing* (pp. 109–115).
16. Forster, A., & Murphy, A. L. (2009). CLIQUE: Role-free clustering with Q-learning for wireless sensor networks. In *Proceedings of the 29th IEEE international conference on distributed computing systems* (pp. 441–449).
17. Aoudia, F. A., Gautier, M., & Berder, O. (2018). RLMan: An energy manager based on reinforcement learning for energy harvesting wireless sensor networks. *IEEE Transactions on Green Communications and Networking, 2*(2), 408–417.
18. Guo, W., & Zhang, W. (2014). A survey on intelligent routing protocols in wireless sensor networks. *Journal of Network and Computer Applications, 38,* 85–201.
19. Ye, D., & Zhang, M. (2017). A self-adaptive sleep/wake-up scheduling approach for wireless sensor networks. *IEEE Transactions on Cybernetics, 48*(3), 979–992.
20. Savaglio, C., Pace, P., Aloi, G., Liotta, A., & Fortino, G. (2019). Lightweight reinforcement learning for energy efficient communications in wireless sensor networks. *IEEE Access, 7,* 29355–29364.
21. Raj, A. B., Ramesh, M. V., Kulkarni, R. V., & Hemalatha, T. (2012). Security enhancement in wireless sensor networks using machine learning. In *Proceedings of the IEEE 14th international conference on high performance computing and communication* (pp. 1264–1269).
22. Kulkarni, R. V., Forster, A., & Venayagamoorthy, G. K. (2010). Computational intelligence in wireless sensor networks: A survey. *IEEE Communications Surveys & Tutorials, 13*(1), 68–96.
23. Hu, J., & Wellman, M. P. (2003). Nash Q-learning for general-sum stochastic games. *Journal of Machine Learning Research, 4*(Nov), 1039–1069.
24. Gosavi, A. (2009). Reinforcement learning: A tutorial survey and recent advances. *INFORMS Journal on Computing, 21*(2), 178–192.
25. Bu, L., Babu, R., & De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 38*(2), 56–172.
26. Yau, K. L. A., Komisarczuk, P., & Teal, P. D. (2012). Reinforcement learning for context awareness and intelligence in wireless networks: Review, new features and open issues. *Journal of Network and Computer Applications, 35*(1), 253–267.

27. Yau, K. L. A., Goh, H. G., Chieng, D., & Kwong, K. H. (2015). Application of reinforcement learning to wireless sensor networks: models and algorithms. *Computing, 97*(11), 1045–1075.
28. Al-Karaki, J. N., & Gawanmeh, A. (2017). The optimal deployment, coverage, and connectivity problems in wireless sensor networks: Revisited. *IEEE Access, 5,* 8051–18065.
29. Borasia, S., & Raisinghani, V. (2011). A review of congestion control mechanisms for wireless sensor networks. In *Technology systems and management* (pp. 201-206). Berlin: Springer.
30. Ishmanov, F., Malik, A. S., & Kim, S. W. (2011). Energy consumption balancing (ECB) issues and mechanisms in wireless sensor networks (WSNs): A comprehensive overview. *European Transactions on Telecommunications, 22*(4), 151–167.
31. Du, X., & Chen, H. H. (2008). Security in wireless sensor networks. *IEEE Wireless Communications, 15*(4), 60–66.
32. Wang, B. (2011). Coverage problems in sensor networks: A survey. *ACM Computing Surveys (CSUR), 43*(4), 1–53.
33. Yetgin, H., Cheung, K. T. K., El-Hajjar, M., & Hanzo, L. H. (2017). A survey of network lifetime maximization techniques in wireless sensor networks. *IEEE Communications Surveys & Tutorials, 19*(2), 828–854.
34. More, A., & Raisinghani, V. (2017). A survey on energy efficient coverage protocols in wireless sensor networks. *Journal of King Saud University-Computer and Information Sciences, 29*(4), 428–448.
35. Cardei, M., Wu, J., & Lu, M. (2006). Improving network lifetime using sensors with adjustable sensing ranges. *International Journal of Sensor Networks, 1*(1–2), 41–49.
36. Kumar, D. P., Amgoth, T., & Annavarapu, C. S. R. (2019). Machine learning algorithms for wireless sensor networks: A survey. *Information Fusion, 49,* 1–25.
37. Hefeeda, M., & Ahmadi, H. (2007). A probabilistic coverage protocol for wireless sensor networks. In *Proceedings of IEEE international conference on network protocol* (pp. 41–50).
38. Mohamadi, H., Salleh, S., Razali, M. N., & Marouf, S. (2015). A new learning automata-based approach for maximizing network lifetime in wireless sensor networks with adjustable sensing ranges. *Neurocomputing, 153,* 11–19.
39. Chenait, M., Zebbane, B., Benzaid, C., & Badache, N. (2015). Sleep scheduling with predictive coverage redundancy check in wireless sensor networks. In *Proceedings of the international symposium on wireless communication systems* (pp. 366–370).
40. Basheer, S., Mathew, R. M., Ranjith, D., Sathish Kumar, M., Sundar, P., & Balajee, J. M. (2019). An analysis on barrier coverage in wireless sensor networks. *Journal of Computational and Theoretical Nanoscience, 16*(5–6), 2599–2603.
41. Niculescu, D., & Nath, B. (2001). Ad hoc positioning system (APS). In *Proceedings of the IEEE global telecommunications conference*, vol. 5 (pp. 2926–2931).
42. Zhu, J., & Papavassiliou, S. (2003). On the energy-efficient organization and the lifetime of multi-hop sensor networks. *IEEE Communications Letters, 7*(11), 537–539.
43. Lauer, M., & Riedmiller, M. (2000). An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *Proceedings of the 17th international conference on machine learning*.
44. Littman, M. L. (2001). Friend-or-foe Q-learning in general-sum games. *ICML, 1,* 322–328.
45. Greenwald, A., Hall, K., & Serrano, R. (2003). Correlated Q-learning. In *ICML*, vol. 3 (pp. 242–249).
46. Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings* (pp. 157–163).
47. Wang, X., & Sandholm, T. (2003). Reinforcement learning to play an optimal Nash equilibrium in team Markov games. In *Advances in neural information processing systems* (pp. 1603–1610).

Anamika Sharma received M.Tech. degree in Computer Science and Engineering in 2014 from Punjab Technical University, Jalandhar, Punjab and B.Tech. degree in Information Technology in 2011, from Himachal Pradesh University, Shimla, India. She is pursuing Ph.D. degree in Computer Science and Engineering at National Institute of Technology Hamirpur, Himachal Pradesh, India. Her research interests include computer networks, wireless sensor networks, and QoS in WSNs.

Siddhartha Chauhan received the Ph.D. degree in Computer Science and Engineering from National Institute of Technology Hamirpur, Himachal Pradesh, India, in 2013 and the M.Tech. degree in Computer Science and Engineering from Indian Institute of Technology Roorkee, Uttrakhand, India, in 2003. He has published many research papers in reputed journals and international conferences. He is currently with Computer Science and Engineering Department, National Institute of Technology Hamirpur, Himachal Pradesh, India. His research interests include mobile ad hoc network, and wireless sensor network.