Addressing Coverage Problem in Wireless Sensor Networks Based on Evolutionary Algorithms

Sahar Chehrazad[1], Hadi S. Aghdasi[1], Negin Shariati[2] and Mehran Abolhasan[2]
[1]Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran
[2]School of Computing and Communications, UTS, Sydney, Australia

***Abstract*—Wireless Sensor Networks (WSNs) are the key part of Internet of Things, as they provide the physical interface between on-field information and backbone analytic engines. An important role of WSNs -when collecting vital information- is to provide a consistent and reliable coverage. To Achieve this, WSNs must implement a highly reliable and efficient coverage recovery algorithm. In this paper, we take a fresh new approach to coverage recovery based on evolutionary algorithms. We propose EMACB-SA, which introduces a new evolutionary algorithm that selects coverage sets using a fitness function that balances energy efficiency and redundancy. The proposed algorithm improves network's coverage and lifetime in areas with heterogeneous event rate in comparison to previous works and hence, it is suitable for using in disaster management.**

***Index Terms*—Wireless sensor networks, internet of things, coverage recovery, evolutionary algorithms, disaster management**

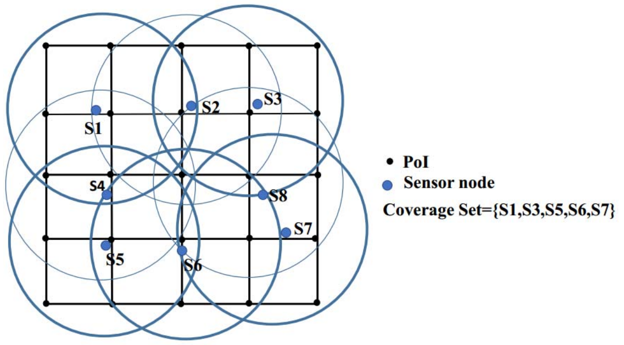


Fig. 1. Area Coverage with Wireless Sensor Networks

I. Introduction

Wireless sensor networks (WSNs) consist of small devices called sensor nodes which have limited sensing, processing and communicating abilities [1], [2]. Monitoring the desired area (area coverage) with the use of WSNs (as represented in Fig.1) plays a very important role in disaster management by finding the exact locations of trapped people and helps rescue teams in the optimal management of disasters. Sensor nodes gather environmental information (e.g. temperature, sound, and pictures) with the aim of fast detection of disaster, identifying and helping trapped people and helping rescue teams to provide a better management of resources [3]–[8]. Scheduling sensor nodes (selecting a subset of sensor nodes for covering whole area) in area coverage can provide us with monitoring the area for a longer time and thus more time for rescuing trapped people. Using evolutionary algorithms for scheduling sensor nodes is beneficial as they scale well to higher dimensional problems in one hand and can easily be adjusted to the problem on the other hand. Therefore these algorithms are obtained in this paper for solving the coverage problem.

Column generation method [9] uses heuristic algorithms to solve Pricing Sub problem (PS). Provided that, heuristics are unable to find a beneficial solution, integer linear programming (ILP) will be utilized to solve PS . Maximum lifetime scheduling method [10] at first, provides a multinomial time estimation procedure for the situation where the density of target points is restricted, and then a multinomial time constant factor estimation algorithm for the general case. Energy-Efficient Probabilistic Area Coverage method [11] first surveys the sensing probabilities of two points with a distance of d and gets the basic mathematical relation between them. If the sensing probability of one point is greater than the specified amount, the other point is covered as well. Hybrid Genetic Algorithm Using a Forward Encoding Scheme [12] merges a genetic algorithm with schedule transition operations, termed STHGA. The greatest gene value of each chromosome is continuously grown with the quality of the solution, which pertains to the number of disjoint complete coverage sets. Hybrid Memetic Framework [13] deals with both disjoint set covers (DSC) and dynamic-coverage-maintenance (DCM) problems simultaneously to maximize the lifetime of WSNs. Ant-Colony-Based Scheduling Algorithm [14] is concerned with finding the solution for the EEC (Energy-Efficient Coverage) problem with a scheduling method based on ant colony optimization in an unstructured WSN. The pheromone field is initialized with the location of sensor nodes and *PoIs* and the number of *PoIs* that can be covered by each sensor node. By doing this, the possibility of selecting sensor nodes which cover more *PoIs* at the beginning can be increased. In the construction graph, ant k adds sensor nodes one at a time till selected sensor set covers all *PoIs*. At the beginning, the ACB-SA algorithm organizes the coverage set with few sensor nodes and uses them continuously for several time steps. Once the energy of the previously organized subset was nearly depleted, the ACB-SA algorithm puts some new sensor nodes in the coverage set to cover all *PoIs*.

All above-mentioned algorithms did not have a special look at heterogeneous event rate in their works. Since the events are distributed unequally in the area with heterogeneous event rate, if sensor nodes which are placed in the areas with more event rates are used for successive time steps, they will be depleted after a short time. Therefore, in some applications like disaster management, the network may be unable to cover some areas after a while. The proposed algorithm is based on evolutionary algorithms. It not only meets the criteria in the previous works, but also considers event rate heterogeneous in the environment. The EMACB-SA algorithm improves network's coverage and lifetime in comparison to previous ACB-SA algorithm. The organization of this paper is as follows: In Section II, System Model and a list of abbreviations used in this paper are demonstrated. In Section III theoretical analysis and detailed description of the proposed algorithm are represented. Section IV provides an overview of Simulation Results which shows that the proposed algorithm is more efficient in comparison to the existing algorithms. Finally Section V is allocated to the conclusion of the proposed EMACB-SA algorithm. Table I represents a list of abbreviations used in this paper.

978-1-74052-390-5/17/$31.00 ©2017 IEEE

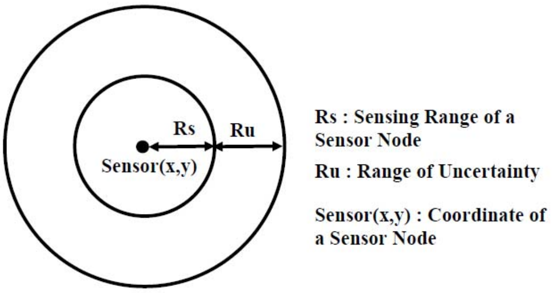

Rs
Ru
Sensor(x,y)
Rs : Sensing Range of a Sensor Node
Ru : Range of Uncertainty
Sensor(x,y) : Coordinate of a Sensor Node

Fig. 2. Rs and Ru of a sensor node

II. System Model

Our area of interest contains *N* sensor nodes which are small and capable of communicating in short distances. In this paper, we used a probabilistic sensor detection model which is more realistic [14]. Sensor nodes detect the event at the Point of Interest (*PoI*) by the intensity of the received signal or energy. This intensity is reduced as the distance between the *PoI* and the sensor node increases. The probability of event detection is a decreasing function of the distance between the *PoI* and the center of the sensor node. The equation for this model is as follows. $\lambda i(j)$ is the detection probability of the sensor node *i* about events at *PoIj*.

$$\lambda i(j) = \begin{cases} 0, & \text{if } d_{ij} > Rs + Ru \\ e^{-a(d_{ij}-Rs)^m}, & \text{if } Rs < d_{ij} \leq Rs + Ru \\ 1, & \text{if } d_{ij} \leq Rs \end{cases} \qquad (1)$$

Where d_{ij} is the Euclidean distance between the sensor node i and *PoIj*. The variables *a* and *m* are decay factors. *Rs* and *Ru* are sensing range of a sensor node and its range of uncertainty in event detection respectively (Fig.2). Events at *PoIj* are definitely detected by sensor node *I* when d_{ij} is less than *Rs*. As d_{ij} increases, the detection probability is reduced. When d_{ij} is greater than *Ru*, the detection probability becomes zero. The variables *a*, *m*, *Rs*, and *Ru* depend on the characteristics of the sensor node and the environment.

III. Proposed Algorithm

In this paper, the proposed algorithm improves area coverage and extends network's lifetime. In the EMACB-SA algorithm, the selection of new coverage set occurs in each time step. A new fitness function is presented to balance energy efficiency and redundancy. A coverage set in which sensor nodes have more remaining energy and cover less redundant *PoIs* is selected in each time step. The importance of these criterion can be defined by coefficients a and b as represented in the equation 3. These coefficients get the values between zero and 1. If the remaining energy is more important to us, a higher amount is given to a. Else, b will get a higher amount. In order to further reduce the redundancy, a greedy algorithm is used in the EMACB-SA algorithm due to which each ant adds selected sensor node to its coverage set if it covers any new *PoI*.

A. Efficient Modified Ant Colony Based Scheduling Algorithm (EMACB-SA)

Suppose that a lot of sensor nodes are distributed in the environment. Table II represents the list of acronyms which are used in this paper.

At the beginning to initialize the pheromone field, the location of sensor nodes and *PoIs* are calculated [14] and put in two matrix named *LSen* and *LPoI* respectively. At the beginning of each time step, *Icov* and Tow_I are calculated. Ants start their travel using the proposed construction graph. Ant *k* chooses a sensor node by roulette-wheel selection randomly in which each sensor node *I* has the selection probability of P(I):

TABLE I
List of Abbreviations

Abbreviation	Definition
ACB-SA	Ant Colony Based Scheduling Algorithm
DCM	Dynamic-Coverage-Maintenance
DSC	Disjoint Set Covers
EC	Energy for Coverage in each active time step which equals to 0.1
EEC	Energy-Efficient Coverage
EI(ts)	Initial Energy which is equal to 1 at ts=0 and the remaining energy of sensor I at each time step (ts) is saved in it.
EMACB-SA	Efficient Modified Ant Colony Based Scheduling Algorithm
ILP	Integer Linear Programming
LPoI	Location of PoI
LSen	Location of Sensor
MaxIter	Maximum Iteration
NA	Number of Ants
NP	Number of PoIs
NS	Number of Sensor nodes
PoI	Point of Interest
PS	Pricing Sub problem
Rs	Sensing Range of sensor node
Ru	Range of Uncertainty
ts	time step
WSNs	Wireless Sensor Networks

$$P(I) = Tow_I / \sum_{n=1}^{NS} Tow_I \qquad (2)$$

Each ant selects sensor nodes one at a time. Then greedy algorithm checks whether the selected sensor node covers any new *PoI*. At the beginning of the EMACB-SA algorithm, the *PoIs* covered by each sensor node are saved in a matrix named *Sen-POI*. Total number of *PoIs* covered by selected sensor nodes are kept in another matrix named *Ant-POI*. Each time a new sensor node is selected and added to the coverage set, *Ant-POI* is updated. Therefore uncovered *PoIs* can be found. The sensor node is added to the coverage set if it covers any new *PoI*. Otherwise, it is dropped. A simple example of the greedy algorithm is demonstrated in Fig.3. This will prevent the selection of sensor nodes which have no benefit. By using this method, sensor nodes are not activated uselessly and the lifetime of the network is increased.

If ant *k* finds a set of sensor nodes which cover all *PoIs*, ant $k + 1$ will start its travel. After all ants (*MaxAnt*) finish their selections, the best solution can be found with the use of proposed fitness function which balances energy efficiency and redundancy. In order to find the best solution two criterion, the cost of the selected sensor nodes and redundancy, should be considered. The higher the remaining energy of the sensor node *I* is, the lower

TABLE II
List of Acronyms

Acronyms	Definition
a,b	Numbers between zero and one and $a + b = 1$
Bcost	Matrix which contains the lowest C_A(ka) in each iteration
Bcover	Matrix which contains the coverage set with lowest C_A(ka) in each iteration
C_A(ka)	Cost of selected sensor set by ant ka
$Cost_W$	Worst cost which happens when the ant selects all sensors as the solution of the problem
Icov(ts)	A matrix which includes the number of POIs that is covered by sensor I and is calculated at the beginning of each time step
Cost(I)	Cost of sensor node I that is equal to K^{EI}
K	random number between zero and one
n	Number of sensor nodes in the selected coverage set by ant
Red_A(ka,ts)	Number of times that each PoI is covered by more than one sensor node in time step (ts) which equals to $\sum_{n=1}^{n} Icov(ts) - NP$
Red_W	Worst redundancy happens when the ant selects all sensor nodes as the solution of the problem which equals to $\sum_{n=1}^{NS} Icov(1) - NP$
Tow_I(ts)	Pheromone field which is updated for each sensor node I at the beginning of each time step which equals to $EI(ts) \times Icov(ts)$

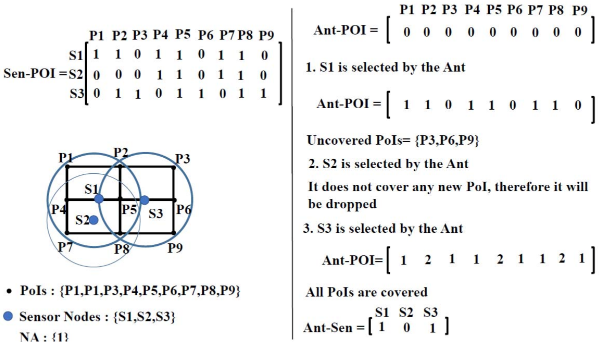
Cost(I) will be. In this case the C_A(ka) will get a lower value. This increases the probability of selecting the coverage set in which sensor nodes have higher remaining energy. On the other hand, as the redundancy of the selected coverage set increases, the C_A(ka) gets a higher value. This reduces the probability of coverage set with higher redundancy.Following fitness function by adaptive weighted sum method [15] was proposed to make a trade-off between these criterion:

$$\begin{aligned} C_A(ka) = a \times ((\sum_{I=1}^{n} Cost(I))/Cost_W) \\ +(b \times Red_A(ka,ts))/Red_W \end{aligned} \tag{3}$$

The selected coverage set with the lowest C_A(ka) and its cost are saved in two matrix named *Bcover* (Best cover) and *Bcost* (Best cost) respectively. In the next iteration, Tow_I is updated for the sensor nodes in *Bcover* as follows:

$$Tow_I = Tow_I + (Bcost / \sum_{ka=1}^{NA} C_A(ka)) \tag{4}$$

In the next iteration, all above-mentioned works are done. The ant with the lowest C_A (ka) is selected. If its cost is lower than the one saved in *Bcost*, it will be replaced by *Bcost*. Then the selected coverage set in new iteration will be replaced by the one in *Bcover*. For selecting a coverage set in each time step, *MaxIter* iteration is completed and the coverage set which is in the matrix *Bcover* is activated in that time step. At the end of each time step, *EC* (a predefined amount) is reduced from the remaining energy (*EI*) of each active sensor node. if there is a sensor node with *EI<EC*, the percent of lost coverage with the death of that sensor node is calculated (the percent of *PoIs* that are not covered anymore is calculated and reduced from the coverage of the last time step). Finally, dead sensor nodes and uncovered *PoIs* are omitted from the matrix of sensor nodes and *PoIs* respectively. Otherwise, it is continued in the next time step. It is also checked at the end of each time step, whether all sensor nodes are dead. In this case, the network is dead and our algorithm stops working. Otherwise, it is continued in the next time step.



$$\text{Sen-POI} = \begin{array}{c|ccccccccc|} & P1 & P2 & P3 & P4 & P5 & P6 & P7 & P8 & P9 \\ S1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ S2 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ S3 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \end{array}$$

- PoIs : {P1,P1,P3,P4,P5,P6,P7,P8,P9}

● Sensor Nodes : {S1,S2,S3}

NA : {1}

$$\text{Ant-POI} = \begin{array}{ccccccccc} P1 & P2 & P3 & P4 & P5 & P6 & P7 & P8 & P9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

1. S1 is selected by the Ant

$$\text{Ant-POI} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Uncovered PoIs= {P3,P6,P9}

2. S2 is selected by the Ant

It does not cover any new PoI, therefore it will be dropped

3. S3 is selected by the Ant

$$\text{Ant-POI} = \begin{bmatrix} 1 & 2 & 1 & 1 & 2 & 1 & 1 & 2 & 1 \end{bmatrix}$$

All PoIs are covered

$$\text{Ant-Sen} = \begin{array}{ccc} S1 & S2 & S3 \\ 1 & 0 & 1 \end{array}$$

Fig. 3. An Example of the Greedy Algorithm

In the proposed algorithm, the death time of sensor nodes is delayed, and sensor nodes which cover more *PoIs* can be used in several time steps. In fact, it enables us to have full coverage of the environment in WSNs for a longer time. The flowchart of the EMACB-SA algorithm is represented in Fig.4.

IV. Simulation Results

All simulations were implemented on a PC with an Intel(R) Core i7-5500u operating at 3 GHz and 12 GB of RAM. We implemented the simulator in MATLAB. The network is modelled as a (100×100) square grid in a $(100m \times 100m)$ Euclidean plane. 40 sensor nodes and 10000 *PoI*s are located on the grid points respectively (one *PoI* on each point). The sensing range of sensor nodes (*Rs*) and Range of uncertainty (*Ru*) are both considered 20m. Both *a* and *m* are set to 0.5. *EI* is set to one at the beginning and it is saved in a matrix named *E* with the size of 1×40. In each time step, *EC* is reduced from *EI* of all active sensor nodes. In the simulation, 40 ants are used. The matrix *Sen-POI* with the size of 40×10000 includes *PoIs* which are covered by each sensor node. The matrix *Ant-Sen* with the size of 40×40 includes sensor nodes which are selected by each ant. The matrix *Ant-POI* with the size of 40×10000 includes *PoIs* which are covered after selection of sensor nodes. Two scenarios are considered for simulating the proposed algorithm to illustrate its efficiency in disaster management. As ant colony algorithms are based on random selection of sensor nodes, the proposed algorithm was run several times and the simulation results were saved in the structure. By averaging the saved results for both scenarios, the area coverage, the number of all sensor nodes (active & sleep) and redundancy were computed. To evaluate the performance of the EMACB-SA algorithm, it simulation results were compared with the ACB-SA algorithm.

A. First Scenario

In this scenario, each sensor node senses one event at each active time step and $EC = 0.1$ is reduced from its *EI*. Sensor nodes are distributed in the area randomly to cover 10000 *PoIs*. In the EMACB-SA algorithm, the new coverage set is selected in each time step. In the ACB-SA algorithm, the selected coverage set is activated for 10 successive time steps. All sensor nodes have died at the end of 10th-time step. Therefore, the new coverage set is selected in each ten-time steps.

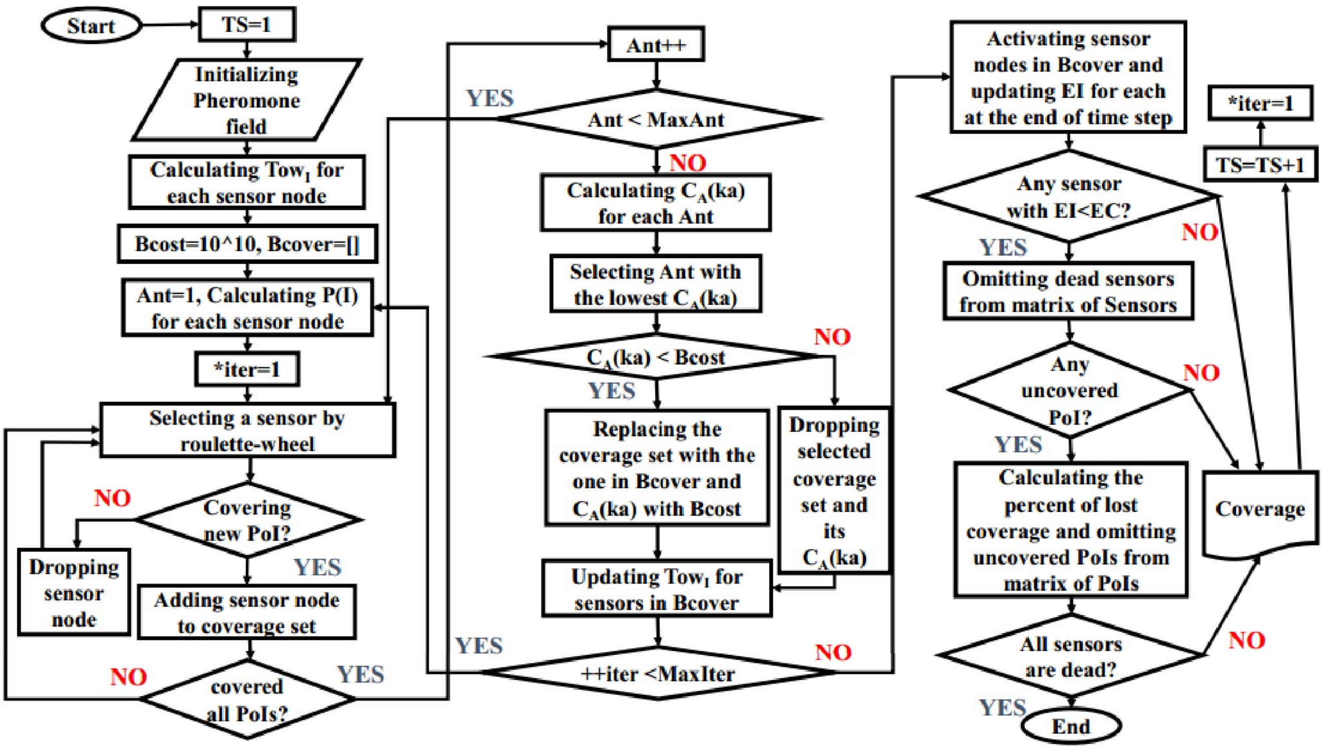

Start
TS=1
Initializing Pheromone field
Calculating Tow$_I$ for each sensor node
Bcost=10^10, Bcover=[]
Ant=1, Calculating P(I) for each sensor node
*iter=1
Selecting a sensor by roulette-wheel
NO
Covering new PoI?
Dropping sensor node
YES
Adding sensor node to coverage set
NO
YES
covered all PoIs?
Ant++
YES
Ant < MaxAnt
NO
Calculating C$_A$(ka) for each Ant
Selecting Ant with the lowest C$_A$(ka)
NO
C$_A$(ka) < Bcost
YES
Replacing the coverage set with the one in Bcover and C$_A$(ka) with Bcost
Dropping selected coverage set and its C$_A$(ka)
Updating Tow$_I$ for sensors in Bcover
YES
NO
++iter <MaxIter
Activating sensor nodes in Bcover and updating EI for each at the end of time step
*iter=1
TS=TS+1
Any sensor with EI<EC?
YES
NO
Omitting dead sensors from matrix of Sensors
Any uncovered PoI?
NO
YES
Calculating the percent of lost coverage and omitting uncovered PoIs from matrix of PoIs
Coverage
All sensors are dead?
NO
YES
End

Fig. 4. Flowchart of the proposed algorithm

B. Second Scenario

In this scenario, some sensor nodes are in the regions with heterogeneous event rate. These sensor nodes sense more than one event at each active time step. Consequently, $EC = 0.2$ is reduced from their *EI*. Other sensor nodes sense one event at each active time step, and $EC = 0.1$ is reduced from their *EI*. Sensor nodes are distributed in the area randomly to cover 10000 *PoIs*. In the EMACB-SA algorithm, the new coverage set is selected in each time step. In the ACB-SA algorithm, if any sensor node in the selected coverage set belongs to a region with heterogeneous event rate, the coverage set will be activated for five successive time steps. At the end of the fifth-time step, some sensor nodes are dead. Sensor nodes which belong to the region with homogenous event rate still have $EI = 0.5$ and will be activated in another time steps. If the selected coverage set includes sensor nodes from both regions and some sensor nodes have $EI = 0.5$, it can be activated only for 5 successive time steps. Therefore, in the ACB-SA algorithm, the new coverage set is selected in each ten or five-time steps.

C. Area Coverage

The results in Fig.5 indicate that in the first scenario, the coverage of the ACB-SA algorithm reaches to less than 50 at 50th-time step. This occurs at 63rd time step in the EMACB-SA algorithm. It can be concluded that the EMACB-SA algorithm is 1.26 times better than the ACB-SA algorithm. In the second scenario, the coverage of the ACB-SA algorithm reaches to less than 30 at 39th-time step, while in the EMACB-SA algorithm this occurs at 50th time step. It can be concluded that EMACB-SA algorithm is 1.3 times better than the ACB-SA algorithm in this case.

D. Number of all sensor nodes (active & sleep)

Fig.6 demonstrates that the number of alive sensor nodes in the EMACB-SA algorithm is 1.23 times more than the ACB-SA algorithm. The reason for the early death of sensor nodes in the ACB-SA algorithm is activating selected coverage set for successive time steps. In the EMACB-SA algorithm, the death time of first sensor node occurs at 17th steps, while this occurs in the ACB-SA algorithm at 10th-time step. It can be deduced that the death time of first sensor node using the EMACB-SA algorithm occurs 1.7 times later than the ACB-SA algorithm.

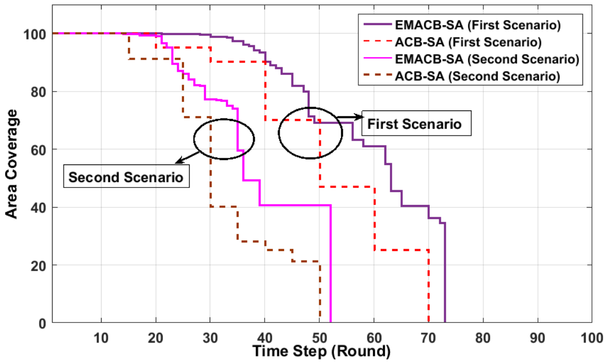


Fig. 5. Area Coverage in both scenarios

In the second scenario, the death time of first sensor node occurs at 15th time step in the EMACB-SA algorithm, while this occurs at the 10th-time step in the ACB-SA algorithm. It is derived that the death time of first sensor node using the EMACB-SA algorithm occurs 1.5 times later than the ACB-SA algorithm.

E. Redundancy

Fig.7 (the first scenario) illustrates that the redundancy of the EMACB-SA algorithm is 7 percent better than the ACB-SA algorithm. After 50th time step, the redundancy of the EMACB-SA algorithm is 98 percent better than the ACB-SA algorithm.

In the second scenario (Fig.8), the redundancy of the EMACB-SA algorithm is 1.73 times higher than the ACB-SA algorithm

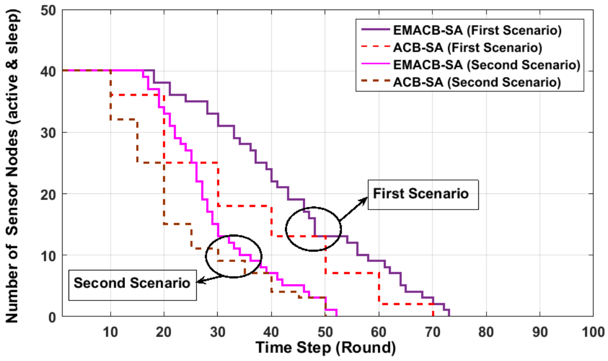


Fig. 6. Number of sensor nodes (active & sleep) in both scenarios

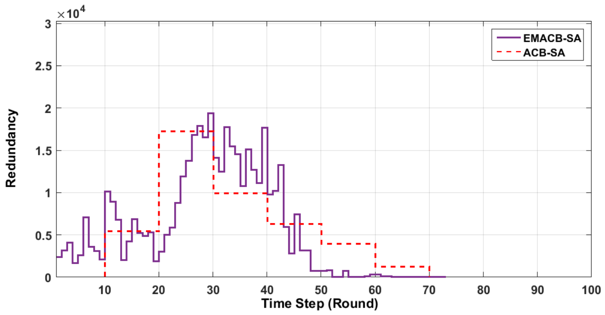


Fig. 7. Redundancy in the environment with homogeneous event rate

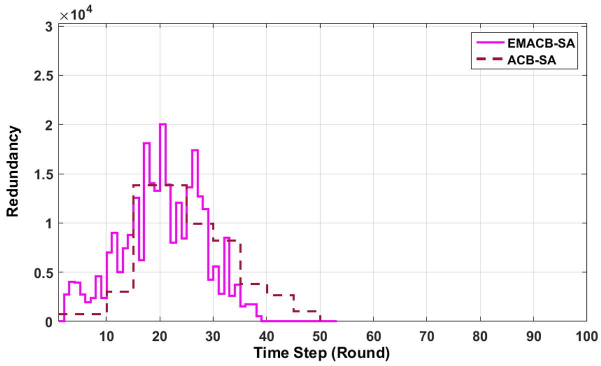


Fig. 8. Redundancy in the environment with heterogeneous event rate

till 29th-time step. After 29th time step, the redundancy of the EMACB-SA algorithm is 66 percent better than the ACB-SA algorithm.

V. Conclusion

This paper presented EMACB-SA algorithm considering different scenarios. In the first scenario, the event rate is homogenous while in the second scenario, the heterogeneous event rate is considered. For both scenarios, the area coverage, number of sensor nodes (active & sleep) and redundancy were computed. In the proposed algorithm, a new coverage set is selected at each time step which increases the redundancy. In order to reduce the redundancy, a new fitness function and a greedy algorithm was proposed. Each sensor node is checked while being selected. The selected sensor node will be added to ant's coverage set if it covers a new *PoI*. Otherwise, the sensor node will be dropped.

Simulations results prove that the EMACB-SA algorithm has increased the area coverage as well as the network's lifetime. It delayed the death time of the first sensor node. Hence the proposed algorithm is a promising solution for area coverage in the environments with heterogeneous event rate.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, *A survey on sensor networks*, IEEE Communications Magazine, Aug. 2002.

[2] I. F. Akyildiz and I. H. Kasimoglu, *Wireless sensor and actor networks: Research Challenges*, Broadband and Wireless Networking Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, 19 May 2004.

[3] P. Uplap and P. Sharma, *Review of Heterogeneous/Homogeneous Wireless Sensor Networks and Intrusion Detection System Techniques*, Proc. of Int. Conf. on Recent Trends in Information, Telecommunication and Computing, ITC, 2014.

[4] E. Cayirci and T. Coplu, *SENDROM: sensor networks for disaster relief operations Management*, Wireless Networks, pp.409-423, 2007.

[5] S. Saha and M. Matsumoto, *A framework for disaster management system and protocol for rescue operation*, TENCON 2007 - 2007 IEEE Region 10 Conference, 2007.

[6] G. Tuna, V.C. Gungor and K. Gulez, *An autonomous wireless sensor network deployment system using mobile robots for human existence detection in case of disasters*, Ad Hoc Networks, pp.54-68, 2014.

[7] M. Erd, *Event monitoring in emergency scenarios using energy efficient wireless sensor nodes for the disaster information management*, International Journal of Disaster Risk Reduction, pp.33-42, 2016.

[8] E. Ergen, *Blockage assessment of buildings during emergency using multiple types of sensors*, Automation in Construction, pp.71-82, 2015.

[9] F. Castano, A. Rossi, M. Sevaux and N. Velasco, *A column generation approach to extend lifetime in wireless sensor networks with coverage and connectivity constraints*, Elsevier Computers and Operation research, pp.1-11, 2013.

[10] Z. Lu, W. Li and M. Pan, *Maximum lifetime scheduling for target coverage and data collection in wireless sensor networks*, IEEE Transactions on Vehicular Technology, pp.1-14, 2014.

[11] Q. Yang, J. Li, J. Chen and Y. Sun, *Energy-Efficient Probabilistic Area Coverage In Wireless Sensor Networks*, IEEE Transactions on Vehicular Technology, pp.1-11, 2014.

[12] X. Hu, J. Zhang and Y. Yu, *Hybrid Genetic Algorithm Using a Forward Encoding Scheme for Lifetime Maximization of Wireless Sensor Networks*, IEEE Transactions On Evolutionary Computation, vol.14, no.5, pp.766-781, 2010.

[13] C. Chen, S. Chandra and C. Chuang, *Hybrid Memetic Framework for Coverage Optimization in Wireless Sensor Networks*, IEEE Transactions On Cybernetics, pp.1-14, 2014.

[14] JW. Lee and JJ. Lee, *Ant-Colony-Based Scheduling Algorithm for Energy-Efficient Coverage of WSN*, IEEE Sensors Journal, vol.12, no.10, pp.3036-3046, 2014.

[15] X. Yu and M. Gen, *Introduction to Evolutionary Algorithms*, Springer-Verlag London Limited, 2010.