Chapter 5
An Evolutionary Computing Based Approach for Optimal Target Coverage in Wireless Sensor Networks

Sheikh Nooruddin, Md. Milon Islam, and Fakhri Karray

Abstract Wireless Sensor Networks (WSNs) are widely used for surveillance and monitoring tasks. Coverage control of wireless sensor networks deals with optimization of sensor deployments to satisfy k–coverage of targets. In this paper, a mathematical model of coverage control while optimizing the overall cost is presented. A Genetic Algorithm (GA) is used to optimize the coverage control problem to minimize the cost while satisfying k–coverage constraint. Various initial sensor deployment models are tested and compared. Both static and dynamic hyperparameter tuning methods such as grid search, Dynamic Increasing of Low Mutation ratio/Dynamic Decreasing of High Crossover ratio (ILM/DHC), and Dynamic Decreasing of High Mutation ratio/Dynamic Increasing of Low Crossover ratio (DHM/ILC) are tested. The evolutionary computing based solution is able to optimize the placement of sensors for various coverage scenarios.

S. Nooruddin (✉) · M. M. Islam · F. Karray
Centre for Pattern Analysis and Machine Intelligence, Department of Electrical and Computer Engineering, University of Waterloo, N2L 3G1 Ontario, Canada
e-mail: sheikh.nooruddin@uwaterloo.ca

M. M. Islam
e-mail: milonislam@uwaterloo.ca

F. Karray
e-mail: karray@uwaterloo.ca

F. Karray
Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, United Arab Emirates

A. Zimmermann et al. (eds.), *Human Centred Intelligent Systems*, Smart Innovation, Systems and Technologies 310, https://doi.org/10.1007/978-981-19-3455-1_5

5.1 Introduction

Wireless sensor networks have become more widely used due to the rapid advancement of wireless technology and their lower cost. Wireless sensor networks monitor and record physical environments. WSNs have widespread applications such as habit monitoring, disaster tracking, security surveillance, and wildlife tracking [1–3]. To perform these tasks in the real–world, sensors are generally deployed to cover the entire surveillance area or a specific region of interest in the surveillance area. Coverage control problems deal with optimizing sensor positions to provide k–coverage of targets in a surveillance area. k–coverage in a sensor network is achieved when each target in the surveillance area is covered by at least k number of different sensors [4, 5]. As range of sensors increases, their cost also increases. Thus, coverage control problem can be formulated with constraints in a way to optimize the sensor placement while reducing the cost of overall placement and ensuring k–coverage.

For most optimal sensor deployment problems, the coverage control problem is considered to be NP-hard [6, 7]. Naïve or completely deterministic solutions such as the circle packing algorithm for coverage control cannot be optimally used for large areas. Various bio-inspired algorithms such as swarm intelligence, multi-objective optimization, evolutionary and ecology based optimization techniques have been historically used to solve the optimal target coverage problem in wireless sensor networks [8, 9].

In this paper, a mathematical model of the coverage control problem is presented. Genetic algorithm based approaches are used to optimize the cost while satisfying the set constraints. Various area-wide sensor deployment and initialization models are presented and their effects on the optimized solution are reported. Static hyperparameter tuning is performed to determine the relation between mutation points and crossover points in the genetic algorithm and their effect on finding the optimal solution. Dynamic hyperparameter tuning methods such as Dynamic Increasing of Low Mutation ratio/Dynamic Decreasing of High Crossover ratio, and Dynamic Decreasing of high Mutation ratio/Dynamic Increasing of Low Crossover ratio were also tested.

The rest of the paper is structured as follows. Section 5.2 describes the most recent and widely used approaches that have been developed to find out the optimal target coverage in WSNs. The mathematical model for the cost optimization problem formulation is presented in Sect. 5.3. Section 5.4 demonstrates the proposed methodology including genetic algorithm with selection, crossover, and mutation. The experimental results and brief discussion are depicted in Sect. 5.5. Section 5.6 concludes the paper with potentials future works.

5.2 Related Works

In recent years, several works [10–12] have been done to optimize the coverage and localization of WSNs using nature-inspired algorithms. The most recent and popular works that have been conducted to address the issues with area coverage are demonstrated as follows.

The authors of [13] used genetic algorithm to maximize the coverage of homogeneous WSNs, with the primary goal of calculating the total area occupied by the sensing devices. For maximum coverage, the developed system used a limited number of sensor nodes that did not intersect and did not extend beyond the reported area. The experiments for this work are carried out by fine-tuning the variables of the genetic algorithm until the maximum coverage is achieved, and it is finally demonstrated that the developed scheme performs well and is stable. Hanh et al. [14] proposed a modified version of a genetic algorithm named MIGA to address the challenge of optimizing coverage area in a sensor network with heterogeneous sensing coverages. The proposed approach presented the problem using a new entity model that included processes such as a hybridization of Laplace and arithmetic crossover, a local search, and various heuristic settings. This research found the best sensor placement and optimized the area coverage without intersecting the sensor nodes. The fitness value of this system involves the calculation of integral area that calculates the coverage of the area for the given sensor nodes. The experimental findings show that the proposed technique performed the best with respect to existing prominent algorithms for optimization. Tian et al. [15] introduced an improved genetic algorithm and binary ant colony optimization technique to address the optimal coverage issue in WSN. The binary code of the proposed methodology assumed that each ant had limited intelligence, and each route was associated with a relatively small storage area which significantly enhances computing performance. The developed technique exhibited a high coverage rate which effectively extended the lifetime of the network.

In another work, the authors of [16] demonstrated a technique that can maximize coverage with the least number of nodes placed randomly in the target location based on Harmony Search Algorithm (HSA). A variable length encoding technique is utilized here to represent the number of nodes in each possible option. The fitness function includes the number of nodes, coverage ratio, and distance between the sensor nodes. The experiment revealed that the proposed technique achieved high performance with minimal cost in terms of strength, consistency, and efficiency. However, the system failed to address the issue of connectivity and power consumption. Binh et al. [17] considered enhanced Cuckoo Search and Chaotic Flower Pollination algorithms to improve area coverage with a fixed number of heterogeneous sensor nodes. The implementation of these two algorithms outperformed the state-of-the-art promising methods in terms of reliability, computing time, and fast convergence. However, the performance of the proposed system is shown to be somewhat unsteady. The authors of [18] developed a framework to calculate the optimal positions of homogeneous sensors for providing area coverage. The scope of this study is focused on an approach for identifying non-penetrable objects. In this system, a co-operative gradual coverage technique is approximated and the topic is described as a Mixed Integer Nonlinear Programming (MINLP). The fitness value is to minimize the cost of the sensors placement. The proposed scheme is implemented using a heuristic search algorithm.

5.3 Mathematical Model

For the sake of illustration of the approach, we present and explore the following mathematical model for the localization and coverage control problem. Our objective is to provide an optimal sensor deployment scheme while minimizing cost of the deployment and ensuring the k–coverage of the targets.

5.3.1 Assumptions

In this preliminary investigation, we assume the following:

i. The location or co-ordinates of the targets are known.
ii. The available types of sensors for deployment are known.
iii. The cost of each type of available sensor unit is known.
iv. The range of each type of available sensor unit is known.
v. The set of potential sites to install sensors is known at all times.
vi. Sensors have omnidirectional coverage, meaning, regardless of the angle between a sensor and a target, if the target is within the sensor's effective range, the target is considered to be covered.
vii. Environmental effects on radio signal, transmission power, as well as various delays associated with large-scale wireless sensor networks are ignored.
viii. The surveillance area is assumed to be flat or two-dimensional in nature, meaning, there are no hills or dips in the area we are exploring. This ignores transmission or reception problems associated with natural elevations.
ix. It is assumed that each possible potential site can only hold one unit of sensor or target. A site cannot hold multiple sensors. A site similarly cannot hold one target and one sensor.

5.3.2 Notation

The following mathematical notations are used in this paper.

Sets

i. **T**: the set of targets that needs to be detected in the surveillance area. t_k denotes a target in site k.
ii. **N**: the set of available types of sensor nodes.
iii. **S**: the set of potential sites where the sensors can be installed.
iv. $\mathbf{D_a(i)}$: the set of targets that can be covered by a sensor of type a which is placed at site i.

$$\mathbf{D_a(i)} = \{j | d(i, j) \le r_a\}$$

v. **C**: the set of costs of the respective type of sensor. The set of **N** and set of **C** is equal in size. c_a denotes the unit cost of sensor of type a.
vi. **R**: the set of the range of available sensor types. The set of **N** and set of **R** is equal in size. r_a denotes the range of a sensor of type a.

Decision Variables

i. $d(i, j)$: the Euclidean distance between a node in site i and a node in site j. The co-ordinates of site i and j in Cartesian co-ordinate system are (x_i, y_i) and (x_j, y_j) respectively. Equation (5.1) presents how the Euclidean distance is measured.

$$d(i, j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{5.1}$$

ii. x_i^a: a binary variable such that it denotes whether a sensor of type a has been placed at site i. $x_i^a = 0$ denotes the sensor of type a has not been placed at site i. $x_i^a = 1$ denotes the sensor of type a has been placed at site i.

Parameters

1. k: is the coverage for each target that must be achieved. In wireless sensor networks, k-coverage is achieved if every target is covered by at least k number of different sensors.
2. x: is the length of the surveillance area.
3. y: is the width of the surveillance area.

Objective Function

To model the target coverage and localization problem while minimizing cost, the following objective function is considered:

$$\text{minimize} \quad \sum_{i=1}^{I} \sum_{a=1}^{A} c_a x_i^a \tag{5.2a}$$

$$\text{subject to} \quad \sum_{a=1}^{A} \sum_{j \epsilon D_a(i)} x_i^a \geq k, \qquad i = 1, 2, \ldots, I. \tag{5.2b}$$

$$\sum_{a=1}^{A} x_i^a \leq 1, \qquad i = 1, 2, \ldots, I. \tag{5.2c}$$

The total cost of a solution is calculated by summing all the costs of the sensor units available in that solution. This objective function is minimized to eventually arrive at the optimal solution. The objective function presented in Eq. (5.2a) minimizes the cost of the solution while also minimizing the number of sensors used. A target must be covered by at least k different types of sensors at any point in the solution. The target is considered to be covered if it is within the omnidirectional sensing radius of the type of sensor used. The first constraint presented in Eq. (5.2b) ensures the k-coverage of each target. The second constraint presented in Eq. (5.2c) ensures the assumption that a single potential site can only hold a single sensor or target. In all these cases, i is indicating potential sites to install the sensors.

5.4 Proposed Methodology

In this section, the general steps associated with genetic algorithm based approaches as well as the specifics of optimizing the aforementioned mathematical coverage model with genetic algorithm are outlined. At first, the initial deployment models tested in this paper are presented. Coverage model for targets is also provided.

5.4.1 Initial Deployment Models

Various initial deployment models were tested to generate the initial solutions. These initial deployment models are briefly discussed below:

Deterministic Minimal Cost Initialization For this initialization model, the type of sensor that has minimal cost per unit is determined at first. The minimal number of sensors of that type needed to cover the entire surveillance area is then calculated. These unit sensors are divided in the entire surveillance area in an equidistant manner from one another in both directions. Equation (5.3) calculates the total number of sensors of minimal cost required to cover the entire surveillance area. n_m denotes the number of required sensors of minimal cost to cover all points in the surveillance area and R_m denotes the range of the sensor type of minimal cost. Equation (5.4) calculates the needed distance between sensors of minimal cost to install them in an equidistant fashion along the x-direction in the surveillance area. Equation (5.5) calculates the needed distance between sensors of minimal cost to install them in an equidistant fashion along the y-direction in the surveillance area. d_x and d_y denote the distance between along the x and y axis.

$$n_m = \frac{(x \cdot y)}{2 \cdot \pi \cdot r_m} \tag{5.3}$$

$$d_x = \frac{x}{\sqrt{n_m}} \tag{5.4}$$

$$d_y = \frac{y}{\sqrt{n_m}} \tag{5.5}$$

Semi-deterministic Initialization This initialization model is a generalization of the deterministic minimal cost initialization model. In this case, the number of required sensors of minimal cost for covering the entire surveillance area is determined at first using Eq. (5.3). The distance between sensors along the x–direction and the distance between sensors along the y-direction are also determined using Eqs. (5.4) and (5.5), respectively. The possible sensor installation sites are then calculated, and a sensor of random type is installed in each potential site. This includes both deterministic and random factors into the initialization.

Random Initialization This initialization is a further generalization of the semi-deterministic model. In this case, the number of required sensors to cover the entire surveillance area for each type of sensor is first calculated. The average of these calculations is then taken to calculate the total number of sensors that will be deployed. Random co-ordinates are chosen equal to the number of deployed sensors. Random types of sensors are then deployed to the previously randomly selected installation points. Equation (5.6) shows how the number of sensors to be deployed is calculated if there are three types of sensors. Here, n_a, n_b, n_c can be calculated using Eq. (5.3). n is the total number of sensors that need to be deployed.

$$n = \frac{n_a + n_b + n_c}{3} \tag{5.6}$$

5.4.2 *Coverage Model*

According to the mathematical model stated above, each sensor type has a sensing radius. r_a denotes the sensing radius of a sensor of type a. t_k denotes a target is located in site k. The binary variable x_i^a denotes weather a sensor of type a is located in site i. The coverage of a target located in site k and a sensor located in site i is 1 if the Euclidean distance between the sites is less than the range of the sensor. The coverage is 0 otherwise. The binary coverage model is denoted as in Eq. (5.7).

$$f(t_k, x_i^a) = \begin{cases} x_i^a, & d(k,i) \le r_a \\ 0, & d(k,i) > r_a \end{cases} \tag{5.7}$$

$$F(t_k) = \sum_{i \in \mathbf{S}} \sum_{a \in \mathbf{N}} f(t_k, x_i^a) \tag{5.8}$$

The total coverage of a target can be calculated by adding up all the coverage of that target site for all possible sensor locations in the surveillance area, as shown in Eq. (5.8). The total coverage of a target must be greater or equal to the k-coverage parameter to consider that target to be covered or localized.

5.4.3 Genetic Algorithm Based Optimization Model

We aim at minimizing the cost of the sensor deployment while ensuring k-coverage for each target using genetic algorithm. The genetic algorithm based solution [19] starts with a population of solutions. The solutions are generated by using the deployment models. Each solution in the population is then evaluated to determine the overall cost of the solution. The selection process is performed to select parents that will be used to generate children for the next generation. The population is then improved in the next generation using the genetic operators-crossover and mutation. Mutation and crossover are the sources of variation in subsequent population. The steps of the genetic algorithm and specific implementation details are briefly described in the following subsections.

Population Genetic algorithm requires an initial population to start the optimization procedure. The population is a collection of individual chromosomes. In this case, each individual chromosome is a unique solution to the coverage control problem. At first, a deployment model is used to generate a random solution. In this case, the solution is a two-dimensional matrix of size $[x, y]$, where each position in the matrix is a representation of the respective co-ordinate in the surveillance area. The value in each position of the matrix denotes the types of sensor installed in the respective co-ordinates. If this solution satisfies all the constraints, the solution matrix is flattened into a one-dimensional matrix and converted into binary representations. The solution is finally inserted into the population as an individual. Figure 5.1 illustrates the solution generation to population generation process.

Evaluation and Selection After the initialization process, each individual in the population is evaluated using the fitness criteria presented in Eq. (5.2a). After the fitness of every individual in a population is calculated, the selection process of the genetic algorithm begins. The selection process is the process by which parents

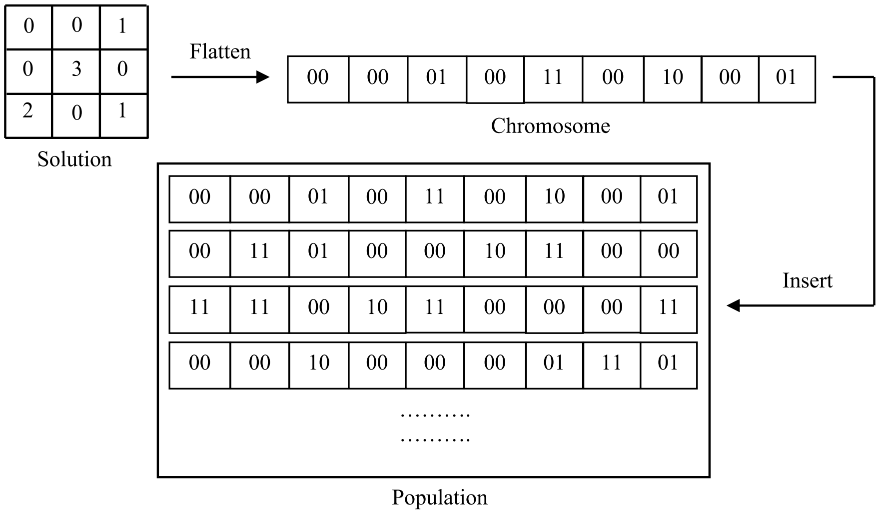

Fig. 5.1 Generation of the population from deployment models

are selected for generating offspring for future generations. The selection process depends somewhat on fitness. Number of chromosomes selected for crossover is equal to the population size. However, a single chromosome can be selected multiple times for crossover.

There are various selection mechanisms available, such as n-tournament, roulette wheel, and stochastic ratio based selection. Selection mechanisms defer by how participants are selected and how fitness is utilized in the selection process. n-tournament is a selection process which combines both fitness ranking and randomness in the selection process. Tournament selection makes genetic algorithm resilient towards local optima. The number of participants in the tournament is determined by n. If $n = 3$, three participants are selected at random from the population to participate in the tournament.

The individual with the lowest cost value in the tournament is selected as a parent for reproduction. In cases where higher objective value indicates better solution, the individual with the highest object value is chosen. Tournament selection is used as our selection process. Selection enables the genetic algorithm to exploit the good solutions within the problem space.

Crossover After parents are selected for mating, crossover is performed to generate the offspring. Generally, two offspring are generated from two parents.

Crossover can be of different types such as: single-point crossover, two-point crossover, and multi-point crossover. In single-point crossover, a random point is selected. The part from the random point to the end of the chromosome is interchanged between the parents to generate the two offspring. In two-point crossover, two random points are selected. The parts between the two random points are interchanged between parents to generate the two offspring. In multi-point crossover, multiple points are randomly selected. The points are then sorted and parts between pairs of points are interchanged between parents to generate the two offspring. A crossover rate in genetic algorithm is defined as the probability that crossover will happen between two selected individuals. The crossover ratio defines the percentage of the selected population that will perform the crossover operation.

Mutation Mutation enables the exploration of the problem space, similar to crossover. Mutation rate is defined as the probability that a copy of an allele changes or mutates to another copy of allelic form in a single generation. Mutation randomly changes a bit in the chromosome. In case of the coverage control problem, the mutation is implemented such that mutation changes any element of the chromosome randomly to any of the following states: remove a sensor from that element or change the sensor type to a random type. Mutation ratio defines the percentage of the population that will perform the mutation operation.

Stopping Criterion After mutation, the genetic algorithm replaces the parents with their offspring to get the next generation. The genetic algorithm then performs evaluation, selection, crossover, and mutation repeatedly until the stopping criterion is reached. The stopping criterion can be the set max number of generations, or very small average changes in multiple generations, or a set period of execution time.

Hyperparameter Tuning Both static and dynamic methods [20] can be utilized to tune GA hyperparameters. Grid Search is a static tuning method that tests all combinations of given parameters to determine the performance of an algorithm. ILM/DHC and DHM/ILC are both dynamic parameter-tuning methods that vary the crossover and mutation ratios of GA's based on the generation. ILM/DHC initializes with very low mutation ratio and very high crossover ratio. As the generation number increases, ILM/DHC algorithm increases mutation ratio and decreases the crossover ratio, eventually maximizing mutation ratio and minimizing crossover ratio in the final generation. DHM/ILC is the exact opposite of ILM/DHC.

5.5 Experimental Results and Discussion

In this section, the performance of the genetic algorithm based coverage control optimization problem will be analyzed.

5.5.1 Simulation Involving Large Area

Genetic algorithm has numerous hyperparameters that can be tuned. Table 5.1 presents the parameters of the simulation as well as the genetic algorithm hyper-

Table 5.1 Hyperparameters for the simulation

Parameter name	Parameter value
Area of simulation	500×500 m^2
Number of targets	17
Types of sensors	3
Cost of sensors	[300, 170, 65]
Range of sensors	[100, 70, 30] m^2
GA-population size	250
GA-Maximum number of generations	2000
GA-crossover rate	0.8
GA-mutation rate	0.1
GA-crossover points	2
GA-mutation points	100

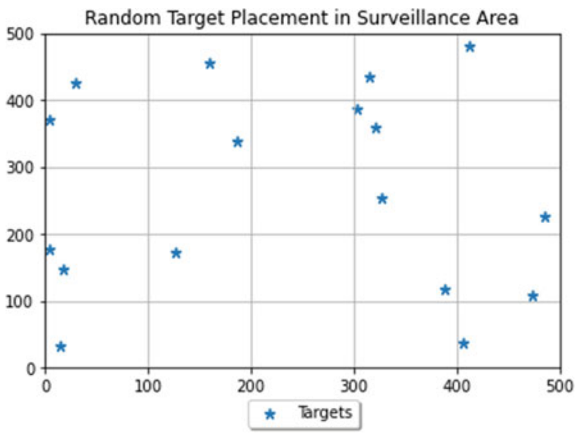

Fig. 5.2 Random target placement in the surveillance area

parameters. As presented in Table 5.1, the number of targets used for simulation is 17. Three types of sensors are available for deployment. Type-1 sensor has the highest range and cost, while type-3 sensor has the lowest range and cost. Figure 5.2 demonstrates the random target placement in the surveillance area.

Figure 5.3a illustrates the best solution found using the genetic algorithm approach when $k = 1$ and semi-deterministic initialization was used to generate the starting population. In this case, the average initial starting cost of the solutions was 232,500 and the cost of the best solution was 2750. Figure 5.3b depicts the best solution found when k was set to 1 and random initial placement was used to generate the starting population. In this case, the average cost of the starting population was 395,255 and the cost of the best solution was 2090.

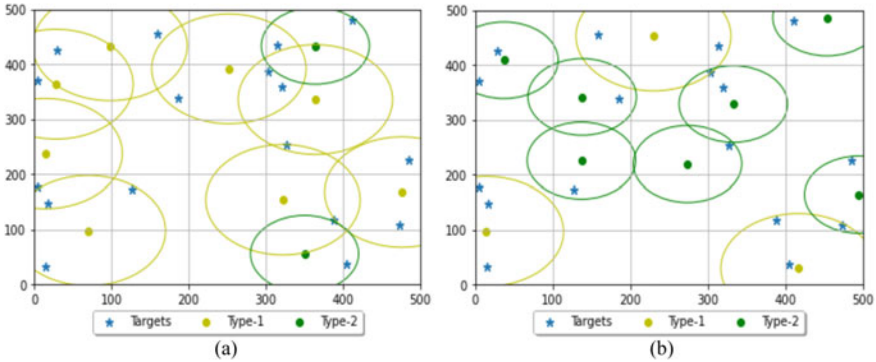

Fig. 5.3 Best solution for $k = 1$ with **a** semi-deterministic, **b** random initial placement

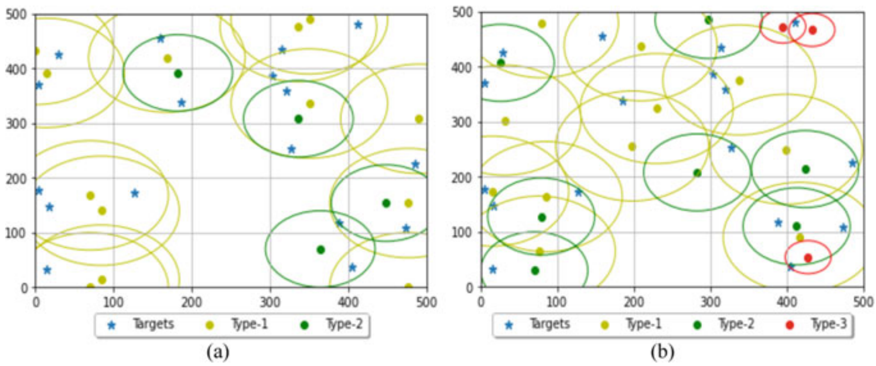

Fig. 5.4 Best solution for $k = 2$ with **a** semi-deterministic, **b** random initial placement

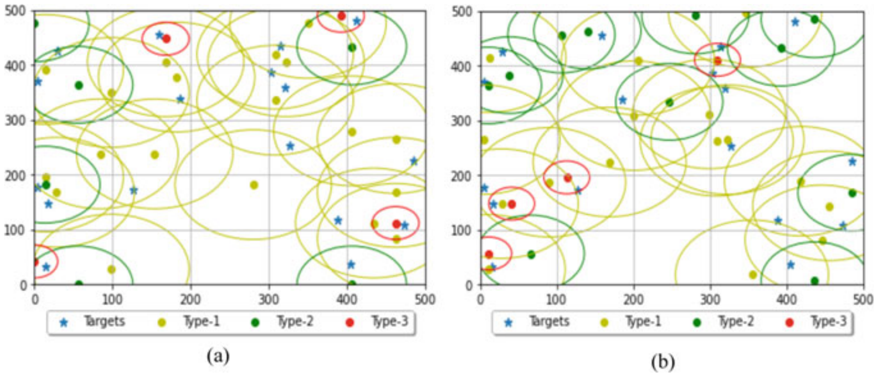

Fig. 5.5 Best solution for $k = 3$ with **a** semi-deterministic, **b** random initial placement

Table 5.2 Effect of the initialization method on optimal cost

Initialization Method	Value of k	Initial average cost	Cost of best solution
Semi-deterministic	1	232,500	2750
Random	1	395,255	2090
Semi-deterministic	2	231,210	4580
Random	2	393,025	4685
Semi-deterministic	3	221,240	6990
Random	3	405,500	6930

Figure 5.4a demonstrates the best solution found when $k = 2$ and the semi–deterministic placement was used as the initialization procedure. In this case, the average initial cost was 231,210 and the cost of the best solution was 4580. This increased cost is because of the fact that the coverage criteria has increased. Figure 5.4b illustrates the best-found solution when k was set to 2 and random initialization was used as the initialization method. In this case, the average initial cost was 393,025 and the best solution had the cost of 4685.

Figure 5.5a depicts the case when k was set to 3 and the semi-deterministic initialization algorithm was used for population generation. In this case, the average initial cost was 221,240 and the best solution had a cost of 6990. Figure 5.5b shows the case when $k = 3$ and the random initialization was used for generating the initial population. The average cost of the initial population was 405,500 and the cost of the best solution was 6930.

Table 5.2 presents a concise view of the effect of initialization on final optimized output. When $k = 1$, the random placement performed a lot better than semi-deterministic placement. When $k = 2$, the semi-deterministic placement performed better than random placement. When $k = 3$, both of the placement methods were similar in nature in terms of optimized cost.

The optimal solutions presented in Figs. 5.6, 5.7, and 5.8 do not have any type-3 sensor deployed. These solutions avoid the minimal range sensor type and use the larger ranged type sensors to satisfy the k–coverage criteria.

5.5.2 *Static Hyperparameter Tuning*

Two hyperparameters, namely the number of crossover points and the number of mutation points were tested using grid search to monitor the performance of the genetic algorithm in finding low-cost solutions. For the static hyperparameter tuning, the considered simulation area was 50×50 m^2, and the range of sensors was approximately 10 m^2, 7 m^2, and 3 m^2. A wide range of crossover points and mutation points were considered. All other hyperparameters were similar to the parameters shown in Table 5.1. Figure 5.6 presents the minimal cost solution found for each com-

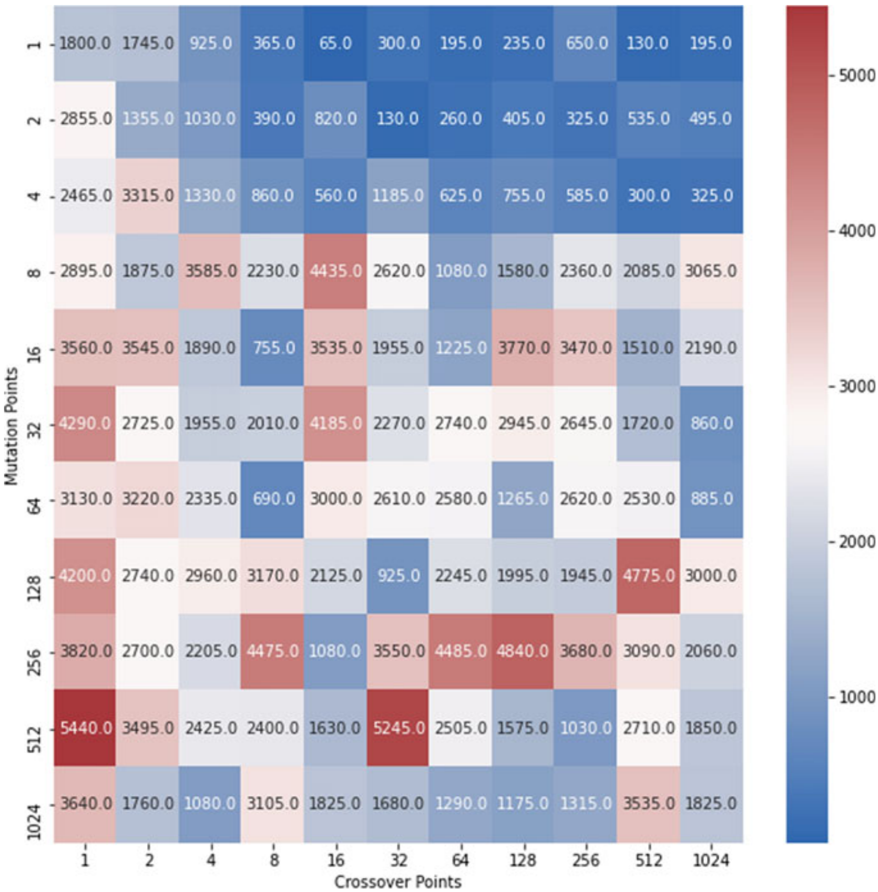

Fig. 5.6 The lowest cost for each mutation point and crossover point combination

bination of crossover and mutation points. The minimum cost of 65 was found when only 1 mutation point and 16-point crossover were used. In that case, the optimal solution used a single maximal range sensor to cover all the targets. Overall, lower mutation points lead to relatively minimal cost solutions, as the solutions are more stable through generations as lesser random changes are being made. The highest cost of 5440 was found when 512 mutation points and single-point crossover were used. The large number of random mutations leads to unstable solutions. Figure 5.7 shows the average cost and the quartiles of the first 30 generations of the best solution. The median and the mean cost decreases gradually over the generations. Various outliers depicting the minimum and maximum cost values are observed for each generation.

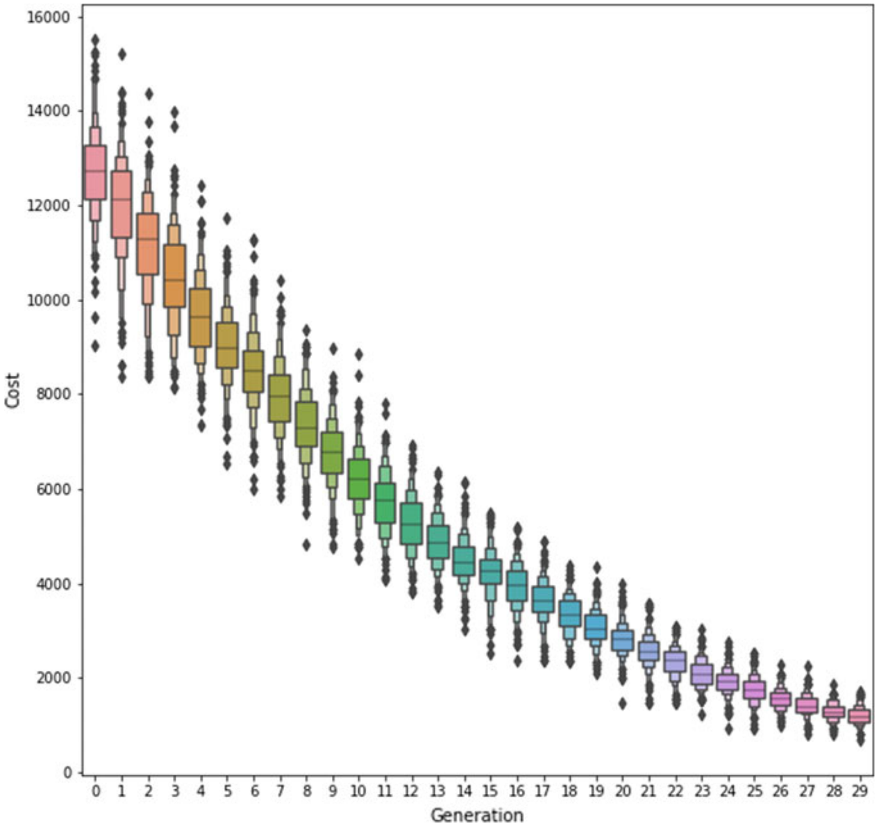

Fig. 5.7 Average cost of each generation for the best solution

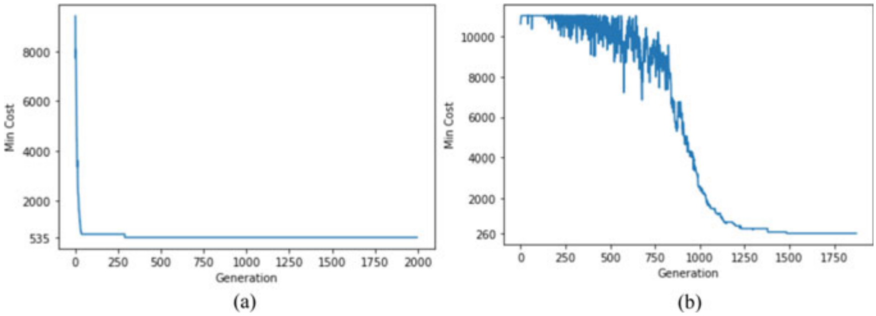

Fig. 5.8 Minimum cost in each generation for **a** ILM/DHC and **b** DHM/ILC method

5.5.3 Dynamic Hyperparameter Tuning

Two dynamic hyperparameter tuning methods, namely ILM/DHC and DHM/ILC were tested. Both methods vary the mutation and crossover ratio based on the generation. In both cases, 16-point crossover and 1 mutation point were used. Figure 5.8a shows the minimum cost found in each generation for the ILM/DHC method. The solution converged very fast. A minimum cost of 535 was found. Figure 5.8b shows the minimum cost of each generation for the DHM/ILC method. In the first 1000 generations, the minimum cost in each generation was very noisy. The cost varied widely. Eventually, the minimum cost of 260 was found around 1500 generations. Throughout the tests, we used a population size of 250, which can be considered a small population size. The ILM/DHC method converged very fast for the small population, whereas the DHM/ILC method converged slowly. Moreover, the DHM/ILC method was able to converge to a better solution than the ILM/DHC method from a cost perspective for this small population. ILM/DHC performs better for small populations. DHM/ILC performs better for larger populations. Both static hyperparameter tuning methods and dynamic hyperparameter tuning methods were able to converge to optimal and sub-optimal minimal cost solutions.

5.6 Conclusion

In this paper, an evolutionary computing technique was used to optimize sensor coverage in an area while minimizing cost and ensuring k–coverage of targets. The genetic algorithm based approach was successful in generating optimized solutions for various k–coverage scenarios. Various initialization techniques were tested and their effect on the final optimized solution was presented. Grid search method was used for static hyperparameter tuning. Combinations of crossover points and mutation points were tested. Two dynamic hyperparameter tuning methods, namely ILM/DHC and DHM/ILC were tested to determine the effects of crossover ratio and mutation ratio. Relatively lower mutation points and multi-point crossover resulted in optimal solutions with minimal cost. Both ILM/DHC and DHM/ILC converged to suboptimal solutions. However, DHM/ILC showed noisy minimum costs during initial generations. In the future, GA based approaches can be employed to optimize cost in more complex radio models incorporating real-life scenarios such as geological structures, signal strength, angle between target and sensor. Floating point GAs can be used to more accurately fine-tune the sensor positions. Various elitism models can be tested in the future to determine the effects on number of generations needed to reach optimal solution.

References

1. Chowdhury, S.M., Hossain, A.: Different energy saving schemes in wireless sensor networks: a survey. Wirel. Pers. Commun. **114**, 2043–2062 (2020)
2. Ponde, S., Lomte, S.: An energy-efficient MAC protocols for wireless sensor networks. In: Advances in Intelligent Systems and Computing, pp. 177–187 (2020)
3. Singh, O., Rishiwal, V., Chaudhry, R., Yadav, M.: Multi-objective optimization in WSN: opportunities and challenges. Wirel. Pers. Commun. **121**, 127–152 (2021)
4. Elhoseny, M., Tharwat, A., Yuan, X., Hassanien, A.E.: Optimizing K-coverage of mobile WSNs. Expert Syst. Appl. **92**, 142–153 (2018)
5. Liu, C., Du, H.: t, K-sweep coverage with mobile sensor nodes in wireless sensor networks. IEEE Internet Things J. **8**, 13888–13899 (2021)
6. Amutha, J., Sharma, S., Nagar, J.: WSN strategies based on sensors, deployment, sensing models, coverage and energy efficiency: review, approaches and open issues. Wirel. Pers. Commun. **111**, 1089–1115 (2020)
7. Liang, D., Shen, H., Chen, L.: Maximum target coverage problem in mobile wireless sensor networks. Sensors **21**, 184 (2020)
8. Swain, A., Swain, K.P., Palai, G., Nayak, S.R.: Optimization of wireless sensor networks using bio-inspired algorithm. In: Smart Sensor Networks Using AI for Industry 4.0. CRC Press, Boca Raton, pp. 1–24 (2021)
9. Kaur, P., Rani, S.: Nature-inspired optimization algorithms for localization in static and dynamic wireless sensor networks: a survey. In: Lecture Notes in Networks and Systems, pp. 219–225 (2021)
10. Sharma, A., Chauhan, S.: Target coverage computation protocols in wireless sensor networks: a comprehensive review. Int. J. Comput. Appl. **43**, 1065–1087 (2021)
11. Singh, A., Sharma, S., Singh, J.: Nature-inspired algorithms for wireless sensor networks: a comprehensive survey. Comput. Sci. Rev. **39**, 100342 (2021)
12. Tripathi, A., Gupta, H.P., Dutta, T., Mishra, R., Shukla, K.K., Jit, S.: Coverage and connectivity in WSNs: a survey. Research issues and challenges. IEEE Access **6**, 26971–26992 (2018)
13. Zorlu, O., Sahingoz, O.K.: Increasing the coverage of homogeneous wireless sensor network by genetic algorithm based deployment. In: 2016 Sixth International Conference on Digital Information and Communication Technology and its Applications (DICTAP). pp. 109–114. IEEE (2016)
14. Hanh, N.T., Binh, H.T.T., Hoai, N.X., Palaniswami, M.S.: An efficient genetic algorithm for maximizing area coverage in wireless sensor networks. Inf. Sci. (Ny) **488**, 58–75 (2019)
15. Tian, J., Gao, M., Ge, G.: Wireless sensor network node optimal coverage based on improved genetic algorithm and binary ant colony algorithm. EURASIP J. Wirel. Commun. Netw. **2016**, 104 (2016)
16. Alia, O.M., Al-Ajouri, A.: Maximizing wireless sensor network coverage with minimum cost using Harmony search algorithm. IEEE Sens. J. **17**, 882–896 (2017)
17. Binh, H.T.T., Hanh, N.T., Van Quan, L., Dey, N.: Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks. Neural Comput. Appl. **30**, 2305–2317 (2018)
18. Shahidehpour, M., Wu, H.: Applications of wireless sensor networks for area coverage in microgrids. IEEE Trans. Smart Grid. **9**, 1590–1598 (2016)
19. Katoch, S., Chauhan, S.S., Kumar, V.: A review on genetic algorithm: past, present, and future. Multimed. Tools Appl. **80**, 8091–8126 (2021)
20. Hassanat, A., Almohammadi, K., Alkafaween, E., Abunawas, E., Hammouri, A., Prasath, V.B.S.: Choosing mutation and crossover ratios for genetic algorithms: a review with a new dynamic approach. Information. **10**, 390 (2019)