Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2007, Article ID 54731, 14 pages
doi:10.1155/2007/54731

Research Article

Optimal and Approximate Approaches for Deployment of Heterogeneous Sensing Devices

Rabie Ramadan,[1] Hesham El-Rewini,[1] and Khaled Abdelghany[2]

[1] *Department of Computer Science and Engineering, Southern Methodist University, Dallas, TX 75275-0122, USA*
[2] *Department of Environmental and Civil Engineering, Southern Methodist University, Dallas, TX 75275-0340, USA*

Received 1 July 2006; Revised 7 December 2006; Accepted 16 January 2007

Recommended by Marco Conti

A modeling framework for the problem of deploying a set of heterogeneous sensors in a field with time-varying differential surveillance requirements is presented. The problem is formulated as mixed integer mathematical program with the objective to maximize coverage of a given field. Two metaheuristics are used to solve this problem. The first heuristic adopts a genetic algorithm (GA) approach while the second heuristic implements a simulated annealing (SA) algorithm. A set of experiments is used to illustrate the capabilities of the developed models and to compare their performance. The experiments investigate the effect of parameters related to the size of the sensor deployment problem including number of deployed sensors, size of the monitored field, and length of the monitoring horizon. They also examine several endogenous parameters related to the developed GA and SA algorithms.

1. INTRODUCTION

Latest advances in wireless sensing technologies have considerably expanded their applications including military, homeland and border security, roadway safety and traffic surveillance, habitat monitoring, and wildlife and environment protection [1–3]. In most of these applications, a network of individual wireless sensors is used to collect state-describing data from a given field. This data is then transmitted through the network to one or more predefined sink nodes for processing. Clearly, the performance of a wireless sensor network (WSN) would largely depend on the characteristics and deployment scheme of individual sensors used to construct this network. Sensors could be characterized by their lifespan, power-saving capabilities, mobile capabilities, reliability, coverage range, and communication range. Using sensors with superior sensing capabilities together with accurate placement of these sensors in the field's hotspots would result in more effective surveillance. In this context, WSNs could generally be classified into (a) homogeneous versus heterogeneous and (b) ad hoc versus fully accessible [4, 5]. Homogeneous WSNs use identical set of sensors, while heterogeneous WSNs consider sensors that differ in one or more of the above characteristics. In ad hoc WSNs, sensors are randomly placed mainly due to limited access to the monitored field. Conversely, in fully accessible WSNs, full access to the monitored field is granted, and hence the deployment scheme of each sensor over the monitoring horizon is predefined before the actual placement of the sensors.

This paper studies fully accessible and heterogeneous WSNs. A modeling framework for the problem of deploying a set of heterogeneous sensors in a field with time-varying differential surveillance requirements is presented. In this framework, the problem is formulated as mixed integer mathematical program with the objective to maximize coverage of a given field. A set of constraints is defined for this mathematical program to guarantee that each zone in the monitored field achieves its required surveillance requirements. Constraints are also defined to ensure that no sensor is used beyond its capacity. The solution of this mathematical program yields the deployment scheme for each used sensor. Two metaheuristics are also used to solve this problem. The first heuristic adopts a genetic algorithm (GA) approach while the second heuristic implements a simulated annealing (SA) algorithm. A set of experiments is used to illustrate the capabilities of the developed models and to compare their performance. The experiments investigate the effect of parameters related to the size of the sensor deployment problem including number of deployed sensors, size of the monitored field, and length of the monitoring horizon. They also examine endogenous parameters related to the developed GA and SA algorithms.

The contribution of this research work is fourfold. First, the modeling framework considers the deployment of heterogeneous set of sensors. Most existing sensor deployment algorithms assume the deployment of identical sensors (e.g., see [1, 6, 7]). Thus, a more general framework is needed for large-scale surveillance operations in which multiple sets of heterogeneous sensors are integrated in one deployment plan. Second, main characteristics of the sensors such as lifespan, power saving and mobile capabilities, and reliability are explicitly considered. Third, the framework considers monitored fields with time-varying differential surveillance requirements. In other words, the framework develops a deployment scheme that is responsive to the temporal-spatial variation in the criticality of the different zones of the monitored field. Finally, developed algorithms in this framework generate near-optimal solution for large-size deployment problems in reasonable running time, which enables the use of these algorithms in applications that require real-time sensor deployment.

Early contribution to the problem of surveillance device deployment returns to Chvátal [8] who introduced the *art gallery problem*. In this problem, the goal is to determine the minimum number of observers required to secure an art gallery with a nonuniform geometry. Different versions of this problem have been studied to include mobile guard and guards with limited visibility (e.g., see [9]). Nonetheless, research in the area of sensing devices deployment has rapidly advanced with the emergence of wireless sensors networks. Most of the research work in this area has concentrated on studying the optimal formation of a WSN that can be used to collect data from a given field and to transmit this data to one or more sink points (e.g., [10, 11]). For example, Chakrabarty et al. [10] proposed a mathematical programming approach for sensor and target locations in distributed sensor networks. The formulation assumes homogenous sensing devices with perfect accuracy. Isler et al. [12] proposed concurrent and incremental deployment methods using sampling theory in which new nodes are deployed based on samples taken from some other randomly deployed nodes. Liu and Mahapatra [13] proposed an integer linear program to maximize the overall lifetime of WSNs. A heuristic is proposed where nodes are forced to send collected data as far as they could and bypass some intermediate nodes to save their energy. Hu et al. [14] proposed the deployment of superior set of sensors, called microservers in a hybrid deployment framework. These microservers are used to filter and route the data in order to reduce the load on other devices. Lee et al. [15] show mathematically and using simulation that using sensors with different lifetime might prolong the overall wireless sensor network's lifetime.

Deployment of mobile nodes has been described in several contexts. One common approach assumes availability of a superior leader that guides several mobile sensor nodes to their deployment positions. Wang et al. [16] proposed an algorithm that uses mobile nodes to cover the blind spots in the monitored area based on static nodes bidding data. In addition, Poduri and Sukhatme [17] introduced an algorithm based on artificial potential fields for self-deployment of mobile sensor nodes. The algorithm aims to achieve maximum coverage of the monitored fields. Furthermore, Howard et al. [18] presented an incremental deployment approach that uses the information gathered from previously deployed nodes in unknown environment to guide deploying the rest of the nodes. Issues related to sensors reliability are presented in [19–21] considering the deployment of stationary sensing devices with imprecise detection capabilities. The objective is to maximize coverage of the monitored field for target detection using a set of devices with probabilistic precessions. In addition, the effect of sensors aging on coverage performance is studied in [22]. Furthermore, example of deploying sensors with self-scheduling (state-switching) capability for energy saving can be found in [23, 24]. The goal is to prolong the network lifetime through scheduling sensor nodes to be inactive during periods with slow or no activities (e.g., off-peak periods).

Maximizing coverage is one of the fundamental objectives of most sensor networks. Nonetheless, coverage is considered in different contexts in the literature. For example, Cardei and Wu [11] categorized the coverage in static wireless sensor networks into area, point, and barrier coverage. Gage [25] classified the coverage into blanket, barrier, and sweep coverage. In addition, Huang and Tseng [26] considers the coverage as a decision problem that determines whether every point is covered by at least k sensors, where k is predefined. This concept is extended by Zhou et al. [27] considering k-connected coverage in which k sensors are connected. Moreover, Poduri and Sukhatme [17] measure the sensor network quality of service by finding the uncovered or low observed areas and highly observed areas in the monitored field. In this paper, the definition of coverage is slightly different. Coverage refers to monitoring the highest important areas in the deployment field by the highest reliable sensors. For example, in border security applications, covering the mountain areas might not be as important as covering the flat areas that people or vehicles can easily pass through. Thus, we use higher reliable sensors in covering hotspots in the monitored filed.

The paper is organized as follows. The sensor deployment problem is formally defined and modeled in Section 2. Section 3 describes the optimal solution approach for this problem. Section 4 presents the GA and SA algorithms used to solve this problem. Experiments that illustrate the performance of these algorithms are presented in Section 5. Finally, the paper concludes in Section 6.

2. PROBLEM DEFINITION AND MODELING

A field $F(A)$ is given to be monitored for a time horizon T using a set of heterogeneous sensing/surveillance devices S. This monitored field is divided into a number of zones A. Each zone $i \in A$ is associated with a time-varying weight function w_i^t, where $t \in T$. This weight function defines the importance of the observations (surveillance requirement) in this zone over the horizon T. The given sensors may differ in their capabilities such as lifespan, allowed number of state-switching, allowed number of moves, and reliability. Moreover, sensor movement cost, described in terms of loss in the sensor's energy, may differ based on the length and the time of the move. A sensor's lifespan L_s is measured by the initial energy of sensor s. The cost factor e_s in terms of energy is associated with each lifetime unit when sensor s is being activated at any unit of time. In addition, sensors are assumed to have limited number of state-switching P_s in which a sensor $s \in S$ changes its state from "on" to "off" or vice versa. For example, a sensor $s \in S$ could be switched to "off" at time $t \in T$ to save its lifetime (energy) for other time periods and zones with higher security requirement.

Moreover, a sensing device could be stationary or mobile. If a stationary device is deployed on a zone $i \in A$, this device is assumed to remain in this zone for the entire lifespan of the device. On the contrary, a mobile sensor can cover multiple zones over a time period T. All mobile devices are assumed to have no restrictions on the start or the end locations of their deployment, but they have restrictions M_s per sensor on the number of moves from zone to another. A sensor transfer between two zones is assumed to be associated with a cost. This cost is expressed in terms of the loss in the device's energy E^t_{sij}. Nevertheless, each sensor $s \in S$ is characterized by a predefined reliability R^t_s that typically changes over time.

A limited set of heterogeneous sensing devices in terms of R^t_s, L_s, and P_s is given in addition to the movement cost E^t_{sij} and the lifetime cost e_s; the objective is to determine their optimal deployment scheme such that the field coverage is maximized. Coverage is maximized when observations with the highest importance are collected. At the same time, sensors with high reliability are assigned to high weight zones. Nevertheless, the coverage is also maximized by serving a zone with only one sensor at any given time and by keeping the sensors active as much as possible.

3. OPTIMAL SOLUTION APPROACH

In this section, we address the optimal solution approach. A mathematical formulation of the problem described Section 2 is developed. The formulation is based on integer linear programming (ILP); this ILP is implemented using CPLEX 8.0. The objective function and list of constraints developed for this program are as follows.

Define

(i) $x^t_{si} = 1$ if device s exists in active state on zone i at time interval t, and 0 otherwise,

(ii) $y^t_{si} = 1$ if device s exists in inactive state on zone i at time interval t, and 0 otherwise,

(iii) $m^t_{sij} = 1$ if device s is moved from zone i to zone j at time interval t, and 0 otherwise,

(iv) $\text{on}^t_{si} = 1$ if device s is turned to active state at time interval t on zone i, and 0 otherwise,

(v) $\text{off}^t_{si} = 1$ if device s is deactivated at time interval t on zone i, and 0 otherwise,

The objective function is defined by the following equation.

Maximize

$$\sum_t \sum_i \sum_s w^t_i \cdot x^t_{si} \cdot R^t_s, \tag{1}$$

where

$$x^t_{si} = \begin{cases} 1 & \text{if the sensing device is deployed in active state} \\ & \text{during time interval } t, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The set of constraints is formulated as follows.

(i) Deployment constraints to relate x^t_{si} and y^t_{si}:

$$x^t_{si} + y^t_{si} \le 1 \quad \forall t, i, s, \tag{3}$$

$$y^{t+1}_{si} \ge x^t_{si} - \sum_j x^{t+1}_{sj} \quad \forall t, i, s, \tag{4}$$

$$y^{t-1}_{si} \ge x^t_{si} - \sum_j x^{t-1}_{js} \quad \forall t, i, s, \tag{5}$$

$$y^{t+1}_{si} \ge y^t_{si} - \sum_j x^{t+1}_{js} \quad \forall t, i, s, \tag{6}$$

$$y^{t-1}_{si} \ge y^t_{si} - \sum_j x^{t-1}_{js} \quad \forall t, i, s, \tag{7}$$

where

$$y^t_{si} = \begin{cases} 1 & \text{if the sensing device is deployed in inactive} \\ & \text{state during time interval } t, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

(ii) Assignment constraints:

$$\sum_i \left(x^t_{si} + y^t_{si}\right) \le 1 \quad \forall t, s, \tag{9}$$

$$\sum_s x^t_{si} \le 1 \quad \forall t, i. \tag{10}$$

(iii) Mobility constraints:

$$m^t_{sij} \ge \left(\left(x^{t+1}_{sj} + y^{t+1}_{sj}\right) + \left(x^t_{si} + y^t_{si}\right)\right) - 1 \quad \forall t, i, j, i \neq j, s, \tag{11}$$

$$m^t_{sij} \le x^{t+1}_{sj} + y^{t+1}_{sj} \quad \forall t, i, j, s, \tag{12}$$

$$m^t_{sij} \le x^t_{si} + y^t_{si} \quad \forall t, i, j, s, \tag{13}$$

$$\sum_i \sum_j \sum_t m^t_{sij} \le M_s \quad \forall s. \tag{14}$$

(iv) State-switching constraints:

$$\text{on}^t_{si} \ge \left(x^{t+1}_{si} + y^t_{si}\right) - 1 \quad \forall t, i, s, \tag{15}$$

$$\text{on}^t_{si} \le x^{t+1}_{si} \quad \forall t, i, s, \tag{16}$$

$$\text{on}^t_{si} \le y^t_{si} \quad \forall t, i, s, \tag{17}$$

$$\text{off}^t_{si} \ge \left(y^{t+1}_{si} + x^t_{si}\right) - 1 \quad \forall t, i, s, \tag{18}$$

$$\text{off}^t_{si} \le y^{t+1}_{si} \quad \forall t, i, s, \tag{19}$$

$$\text{off}^t_{si} \le x^t_{si} \quad \forall t, i, s, \tag{20}$$

$$\sum_t \sum_i \left(\text{on}^t_{si} + \text{off}^t_{si}\right) \le P_s \quad \forall s. \tag{21}$$

(v) Lifespan constraints:

$$\sum_t \sum_i e_s x_{si}^t + \sum_i \sum_j \sum_t E_{sij}^t m_{sij}^t \leq L_s \quad \forall s. \tag{22}$$

(vi) Binary constraints:

$$\{x_{si}^t, y_{si}^t, m_{sij}^t, \text{on}_{si}^t, \text{off}_{si}^t\} = 1 \text{ or } 0 \quad \forall t, i, s. \tag{23}$$

As shown in (1), the objective function maximizes the field coverage which is described as the sum over all time intervals of the products of the observation weight w_i^t, the decision variable x_{si}^t ($x_{si}^t = 1$ if device s exists in active state in zone i in time interval t), and the reliability of the used device R_s^t.

Constraints in (3) ensure that a sensing device is either active or inactive during any time interval. Constraints in (4)–(7) determine the value of the binary variable y_{si}^t based on the value of x_{si}^t. If a sensing device is set to be active in zone i during time interval t, and this device is not used in any zone during the next (previous) time interval, this device is assumed to be inactive and to stay in this zone during the next (previous) time interval. Similarly, if a device is set to be inactive in zone i during time interval t, and this device is not used in any zone during the next (previous) time interval, this device is assumed to remain inactive in the same zone during the next (previous) time interval. The greater than or equal signs used in these constraints prevents the infeasibility of the constraints if the value of $\sum_j x_{sj}^{t+1}$ or $\sum_j x_{sj}^{t-1}$ is turned to be 1 and the value x_{si}^t is equal to 0. However, this might lead y_{si}^{t+1} or y_{si}^{t-1} to be 0 or 1 in some cases. These cases are handled by constraints in (9). Constraints in (9) ensure that each zone is covered by at most one device in any time interval. Also, at each time interval, a surveillance device is covering at most one zone, which is guaranteed in constraints (10).

Constraints in (11)–(13) determine if sensing device s is moved from zone i to zone j at the end of interval t. They compare zones where sensing device s is deployed during time intervals t and $t + 1$. The binary variable m_{sij}^t is set to one if they are different. Constraint (11) uses the "≥" sign instead of "=" sign to avoid the infeasibility in case $(x_{sj}^{t+1} + y_{sj}^{t+1})$ and $(x_{si}^t + y_{si}^t)$ are equal to 0. Again, this might lead m_{sij}^t to have 0 or 1 which is handled by constraints (12) and (13). Constraints in (14) ensure that the number of moves made by a device is less than or equal to the maximum number of moves allowed for this device.

The state switching of a sensing device from active state to inactive state and vice versa are determined in constraints (15)–(20). The binary variables x_{si}^t and y_{si}^t are examined for each sensing device while being deployed in every zone. If both variables are equal to one in two successive time intervals, this indicates that the device's state is altered. The total number of state switchings for each sensing device is computed and compared to the maximum number of switches allowed for each device as given in constraints (21). Constraints in (22) ensure that each sensing device is not utilized beyond its lifespan through the sum over x_{si}^t multiplied by the cost factor e_s of sensor's lifetime. The consumption of a device's lifespan is computed as the sum over all intervals in which the device is active plus the loss in the device lifespan associated with its moves along the different zones. Finally, the integrality of all binary variables is preserved in constraints (23).

4. METAHEURISTIC APPROXIMATE APPROACHES

The deployment problem, described above, is intractable in its general form as well as in many special cases. The art gallery problem which was proven to be NP-hard problem [28] represents a restricted case of the sensor deployment problem. Therefore, seeking sensors optimal deployment scheme for large-scale problems might be impractical.

In this section, we present two approximate metaheuristics to solve large-scale sensor deployment problems. These heuristics adopt genetic algorithm and simulated annealing approaches, respectively. Details on modeling and implementation issues of these two algorithms are provided in the following subsections.

4.1. The genetic algorithm approach

Genetic algorithms are optimization and search techniques inspired by evaluation. They use the principles of genetics and natural selection. GA has been used to solve a wide range of combinatorial problems in different areas. The common major steps, as shown in Figure 4(a), for any genetic algorithm are as follows.

(1) *Generation*: generate an initial population of chromosomes.
(2) *Evaluation*: evaluate the cost of each individual chromosome.
(3) *Selection*: determine the fitness of each individual chromosome.
(4) *Reproduction*: reproduce based on fitness, giving more chances to fit individuals to reproduce. In general, use crossover and mutation operators for reproduction.
(5) Go to 2, until *stopping* criteria are met.

The details of the algorithm and its advanced features can be found in [29–31].

Applying GA to the sensor deployment problem, chromosomes are designed to describe a feasible deployment plan for the set of available sensors. The length of each chromosome (number of genes) is taken to be equal to $|A| * |T| * |S|$, where $|A|$, $|T|$, $|S|$ are number of zones, number of intervals in the monitoring horizon, and number of sensors, respectively. All genes are initially set to zero; however, if a sensor $s \in S$ is deployed at zone $z \in A$, at time interval $t \in T$, the correspondent gene is set to 1. Figure 1 illustrates the structure of a chromosome used to represent the deployment of two sensors ($s1$ and $s2$) in a field of three zones ($z1$, $z2$, and $z3$) which is monitored for two time intervals ($t1$ and $t2$). As shown in the figure, $s1$ is used for the two time intervals and is moved once to cover $z3$ at $t1$ and $z1$ at $t2$. Thus, $s1$ does not exercise the state-switching capability. Sensor $s2$ is used only for the first time interval ($t1$); then it is turned off at time $t2$.

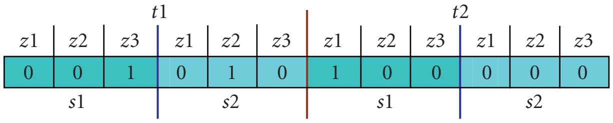


Figure 1: Example of a chromosome for the sensor deployment problem.

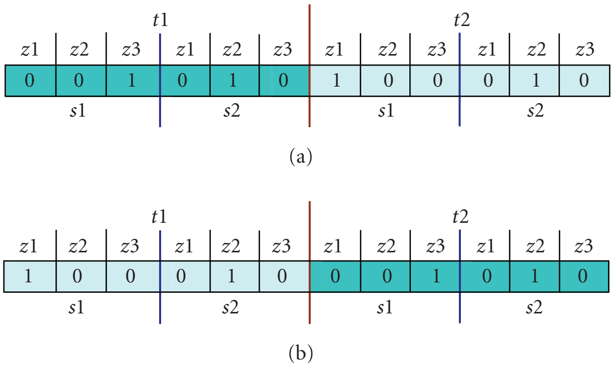


Figure 2: Time-exchange crossover operation (a) before crossover and (b) after crossover. $t1$ and $t2$ of chromosome are exchanged to generate a new chromosome.

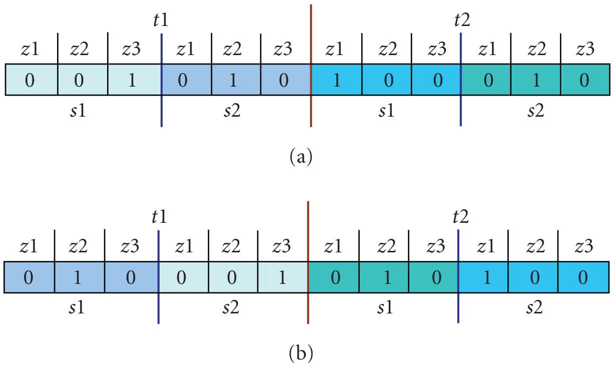


Figure 3: Sensor-exchange crossover operation (a) before crossover and (b) after crossover. $s1$ and $s2$ are exchanged at both times $t1$ and $t2$ to generate a new chromosome.

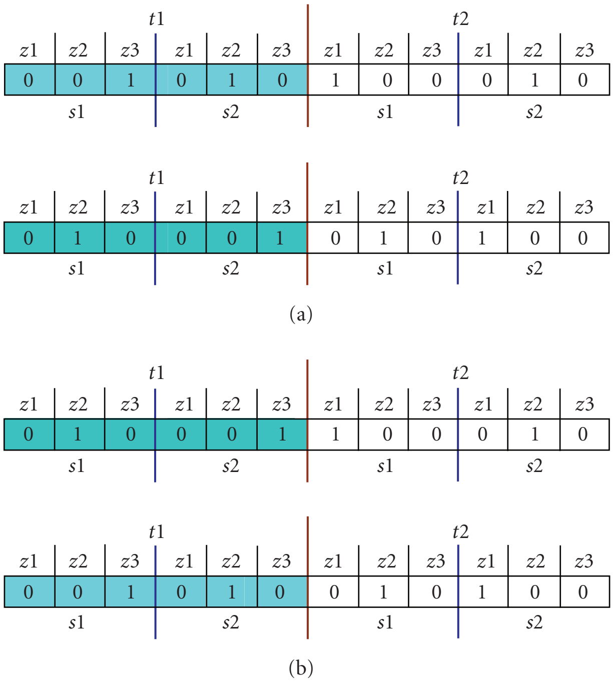
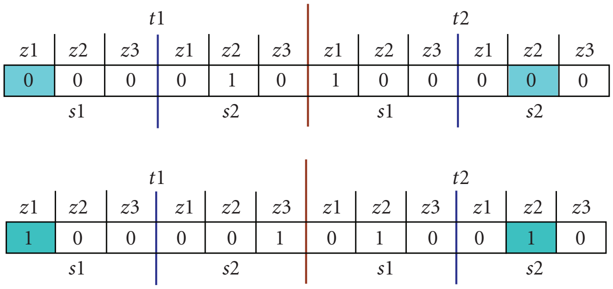


Figure 4: Time-exchange crossover operation between two chromosomes (a) before crossover and (b) after crossover. $t1$ in both chromosomes are exchanged to generate two new chromosomes.

The algorithm allows two chromosome generation mechanisms: single chromosome per iteration or multiple chromosomes per iteration. In the single-chromosome case, an initial chromosome is randomly generated. For a new chromosome to be generated in the subsequent iteration, two different crossover operators are adopted, namely time exchange (TE) and sensor exchange (SE). Using the TE crossover operator, sensors deployment pattern in two randomly selected time intervals are exchanged. Figure 2 illustrates the application of the TE crossover operator. The sensors deployment pattern during time intervals $t1$ and $t2$ is exchanged. The SE crossover operator exchanges the deployment pattern of two randomly selected sensors over the entire horizon. Figure 3 presents an example on the SE crossover operator. The deployment patterns of sensors $s1$ and $s2$ are exchanged over the entire horizon ($t1$ and $t2$).

In the multiple chromosomes mechanism, n chromosomes per iteration are generated, where n is a predefined parameter. The initial n chromosomes can be generated randomly or based on a guided algorithm. In case a guided algorithm is used, the first chromosome is generated randomly; then the second chromosome is generated through applying one of the two crossover operators described above. These two chromosomes are then used to generate the rest of the population as described hereafter. For n new chromosomes to be generated, crossover and mutation operators are applied on chromosomes generated in the previous iteration. Two different crossover operators are adopted, which are time exchange (TE) and best chromosome (BC). The TE crossover is similar to the case where a single chromosome per iteration is generated. However, in the case where multiple chromosomes per iteration are generated, the sensor deployment patterns, in the same time interval, in two different chromosomes are exchanged. Figure 4 presents an example on the application of the TE crossover operator. Two chromosomes exchanges their sensor deployment patterns in time interval $t1$ resulting in two new constraints. The BC crossover operator is similar to the TE operator with the exception that only the two fittest chromosomes are used as parents for all new chromosomes. Following the crossover operations, the mutation operation is used to prevent the search from getting trapped in the local minima and also to prevent chromosomes repetition. In the mutation step, some



t1						t2					
z1	z2	z3	z1	z2	z3	z1	z2	z3	z1	z2	z3
0	0	0	0	1	0	1	0	0	0	0	0
s1			s2			s1			s2		

t1						t2					
z1	z2	z3	z1	z2	z3	z1	z2	z3	z1	z2	z3
1	0	0	0	0	1	0	1	0	0	1	0
s1			s2			s1			s2		

FIGURE 5: Mutation operation; $s1$ at $t1$ and $s2$ at $t2$ are muted to be used on $z1$ and $z2$, respectively.

of the generated chromosomes genes are randomly altered (from zero to one or vice versa). Thus, the search is directed to a new area in the search space. Figure 5 depicts an example of the mutation step for a single chromosome.

Crossover and mutation operations could result in unfeasible chromosomes as some sensors might exceed their capabilities (e.g., lifespan, maximum number of moves, and maximum number of allowed switches). As such, a feasibility check routine is applied for each newly generated chromosome to ensure that all chromosomes in the population are representing feasible deployment patterns.

The fitness of each generated chromosomes is measured using the fitness function given below. One should note the similarity between this fitness function and the objective function of the mathematical program in Section 2,

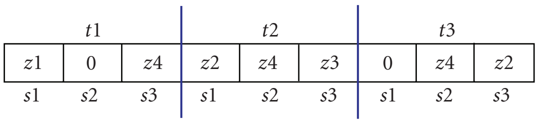
$$F(x) = \sum_{t} \sum_{i} \sum_{s} w_i^t \cdot R_s^t, \qquad (24)$$

where x is the chromosome identifier.

Given the fitness value for each new chromosome, chromosomes are added as part of the current population only if they outperform the current available solution. The algorithm stopping criteria could be based on a fixed number of iterations or a given number of iterations in which the solution does not improve.

4.2. The simulated annealing approach

Simulated annealing (SA) is a randomized search technique for highly nonlinear problems [2]. In its search process, the algorithm is similar to using a bouncing ball that can bounce over mountain from valley to valley based on the ball's temperature until the highest tip is found. The algorithm starts by generating an initial feasible solution and computing its performance. This solution is stored as the best solution obtained so far. Neighborhood of this solution is searched and a new solution is generated. If the new solution's performance is greater than the highest gain found so far (uphill move), the new solution is accepted and saved. If the gain of the new solution is less than the upper bound performance found so far, still accept this new inferior solution but with some probability (downhill move). The probability of accepting inferior solutions is reduced after each iteration (increase of the ball temperature). The process continues until no better solution is found indicating that the maximum possible temperature is achieved. A formal description of the SA algorithm can be described using the following main steps.



t1			t2			t3		
z1	0	z4	z2	z4	z3	0	z4	z2
s1	s2	s3	s1	s2	s3	s1	s2	s3

FIGURE 6: Example on the simulated annealing solution in which 4 zones are monitored by 3 sensors for 3 time units.

Define

(i) X_0 = initial solution, and the best solution so far,

(ii) X_k = current solution,

(iii) $N(X_k)$ = neighborhood of the current solution,

(iv) $G(X_k)$ = performance of the current solution,

(v) $\hat{X}$ = variable to keep the best solution,

(vi) β_k = current temperature,

(vii) β_f=final temperature (highest temperature value),

(viii) α = heating rate for temperature schedule,

(ix) $P(X_{k+1}, X_k)$ = probability of acceptance of a new solution (X_{k+1}) given that the solution is (X_k). This probability is calculated as follows:

$$P(X_{k+1}, X_k) = e^{(G(X_{k+1})-G(X_k))/(\beta_f-\beta_k)}. \qquad (25)$$

Step 1. Set $k = 1$ and select the initial temperature β_1 and the final temperature β_f.

Select an initial solution X_1 and set $X_0 = X_1$.

Step 2. Select a new solution X_{k+1} from $N(X_k)$.

If $G(X_{k+1}) > G(X_0)$, *set* $X_0 = X_{k+1}$ and update $\hat{X}$; then, go to Step 3.

If $G(X_{k+1}) \leq G(X_0)$, *generate* $U_k \sim$ uniform (0,1).

If $U_k \leq P(X_{k+1}, X_k)$, *set* $X_{k+1} = X_k$; otherwise *set* $X_0 = X_{k+1}$; go to Step 3.

Step 3. Update $\beta_{k+1} = \beta_k/\alpha$.

If $(\beta_{k+1} \geq \beta_f)$ stop; else set $k = k + 1$ and go to Step 2.

Applying the SA algorithm for the sensors deployment problem, a solution is represented by a string of integers. The length of this string is $|S| * |T|$. In this string, each sensor-time interval is assigned a zone such that no two sensors are allowed to be deployed on the same zone in the same time interval. If a sensor is not used in one time interval, the corresponding cell in this solution string is assigned to zero. Figure 6 illustrates an example for representing a solution generated by SA algorithm. Similar to the GA algorithm, generated solutions are subjected to feasibility check to ensure the satisfaction of all constraints described in Section 2. In addition, the algorithm can be extended to generate multiple solutions per iteration. In such case, different neighborhoods are explored at the same time. The incumbent value $\hat{X}$ maintains the best solution from all of generated solutions per iteration. A comparison of the SA algorithm performance

Table 1: Performance of the GA and SA algorithms compared to the optimal solution.

Exp no.	No. of zones	No. of sensors	Horizon	Optimal solution running time (s)	GA single solution per iteration time exchange crossover		SA single solution per iteration	
					Objective function (%)	Running time (%)	Objective function (%)	Running time (%)
1	10	5	12	1960	80	0.04	79	0.04
2	20	5	12	32030.8	85	0.009	83	0.008
3	25	5	12	39670	85	.01	85	0.009
4	20	3	12	2400	79	0.05	70	0.04
5	20	5	12	32030.8	85	0.009	83	0.008
6	20	10	12	50056	85	0.006	81	0.006
7	20	5	3	300.8	90	0.4	85	0.3
8	20	5	6	2300	92	0.1	85	0.1
9	20	5	12	32030.8	85	0.009	83	0.008

considering single solution and multiple solution implementations is presented hereafter.

5. EXPERIMENTAL RESULTS

5.1. GA and SA benchmarking and comparison

The mathematical program described in Section 3 is used to provide an optimal solution for the sensor deployment problem. The commercial optimization package CPLEX 8.0 running on a 2.4 GHz machine with 2 GB memory is used to generate the optimal solution for different problem settings. This optimal solution is used to benchmark the performance of solutions obtained by the GA and SA. Three different sets of experiments are conducted. These experiments study the effect of increasing number of zones, number of sensors, and time horizon on the running time required to generate the optimal solution, respectively. In all experiments, the time-varying observations on the different zones were generated randomly following a uniform distribution $U(0, 200)$. In addition, a heterogeneous set of sensors is assumed. The sensors' lifespan L_s is generated randomly as function of the length of monitored horizon, while M_s and P_s are generated randomly based on L_s. For example, if the monitoring horizon is T intervals, the sensor lifespan is generated randomly using the uniform distribution $U(1, T)$ and both M_s and P_s use a uniform distribution function $U(1, L_s)$. In addition, sensors reliability R_s^t is generated randomly using a uniform random generator $R(0, 1)$, where 0 and 1 represent 0% and 100% reliability, respectively. Furthermore, the lifespan cost e_s is set to unity throughout these experiments.

As illustrated in Table 1, the running time required to generate the optimal solution increases exponentially with the increase in the size of the problem. For instance, a running time of 1960 seconds is recorded for a problem of 10 zones, 5 sensors, and a horizon of 12 intervals. This running time jumps to 39670 seconds when the number of zones is increased to 25. Problem settings with dimensions beyond the ones presented in the table could not be generated using the machine mentioned above. The results indicate that both GA and SA algorithms provide high-quality solutions. In the experiment with the lowest performance (experiment 4 using the SA), 70% of the optimal objective function value is obtained. Furthermore, up to 85%, on average, of the corresponding optimal performance is recorded when the GA algorithm is used in experiments 1–9. The running time of both algorithms is noticeably small compared to that of the optimal solution. For example, in experiment 6, the running times of the GA and SA algorithms are observed to be 0.006% of the optimal solution's running time.

On the other hand, the SA seems to converge faster than GA algorithm. These results are confirmed in Figure 7 which illustrates the comparison results of the GA and SA algorithms when different numbers of chromosomes/solutions per iteration are considered.

The number of chromosomes/solutions per iteration is set to range from 1 to 50. The figure presents the comparison in terms of objective performance and running time. In this set of experiments, 300 zones are monitored for 12 time intervals using 200 sensors. Zones weights and sensors capabilities are generated randomly as mentioned above. As shown in Figure 8(a), the genetic algorithms outperform the simulated annealing algorithm in terms of the objective function. On average, the recorded objective performance for the SA algorithm is almost 96% of the genetic algorithm performance. However, the SA running time is less than that of the GA running time by about 9%. This set of experiments also illustrates the impact of the number of chromosomes/solutions per iteration on the solution performance. In general, increasing the number of solutions per iteration resulted in convergence at a better objective function for both algorithms. This is achieved on the expense of the running time, however. For example, the objective performance of a single chromosome/solution is almost 70% of that when 50 chromosomes/solutions per iteration are generated. The required time for a single chromosome/solution is approximately 0.02% of the 50 chromosomes/solutions per iteration case.

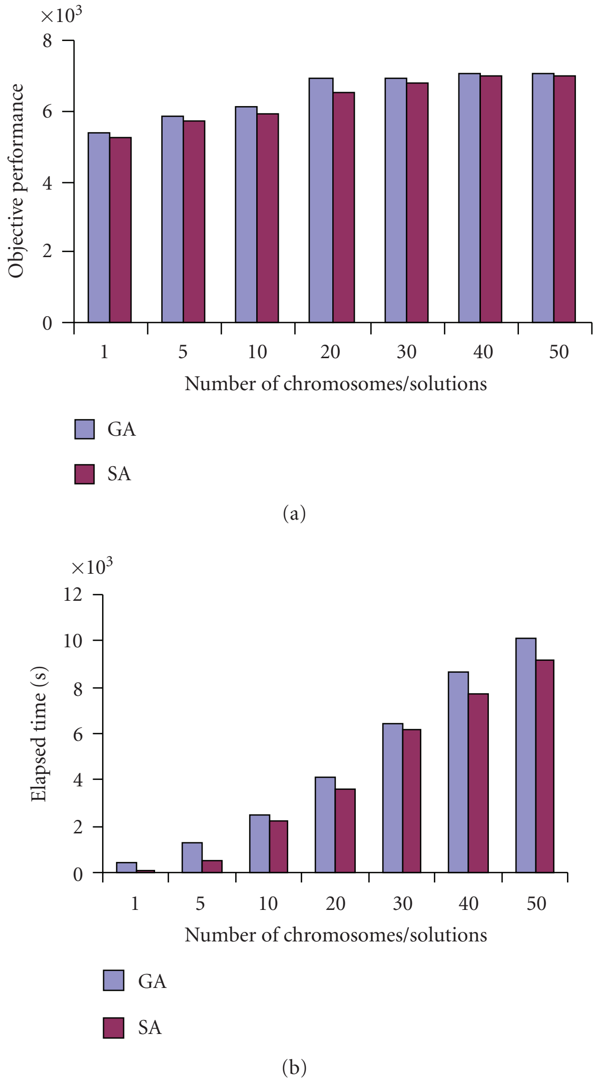


Figure 7: A comparison between genetic and simulated annealing algorithms with different chromosomes/solutions per iteration: (a) objective performance and (b) elapsed time.

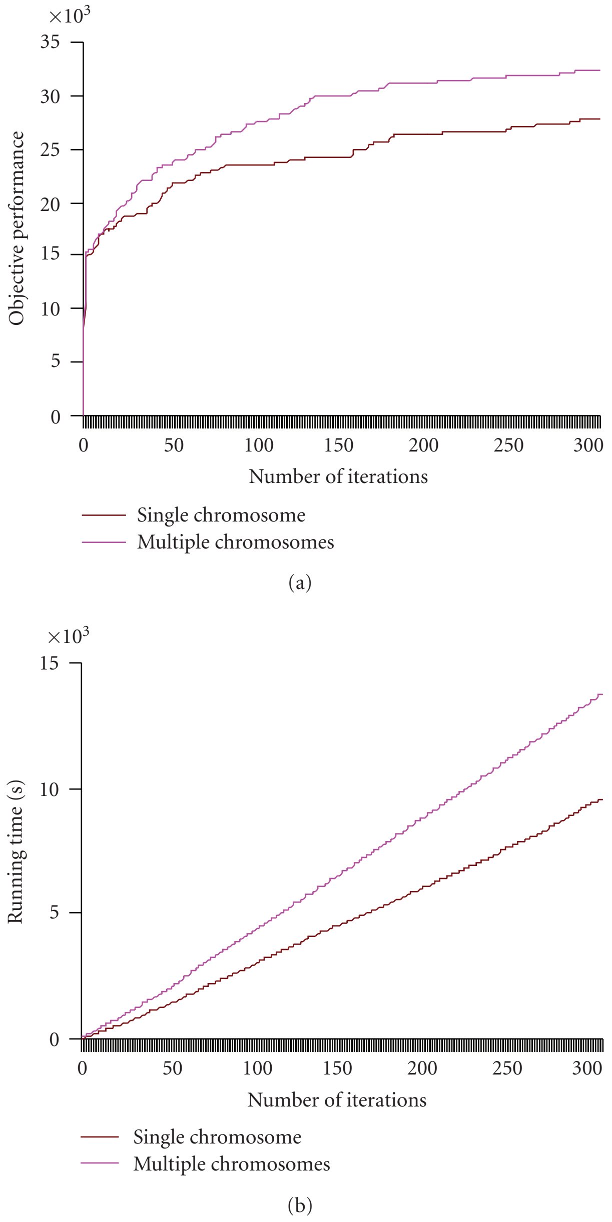
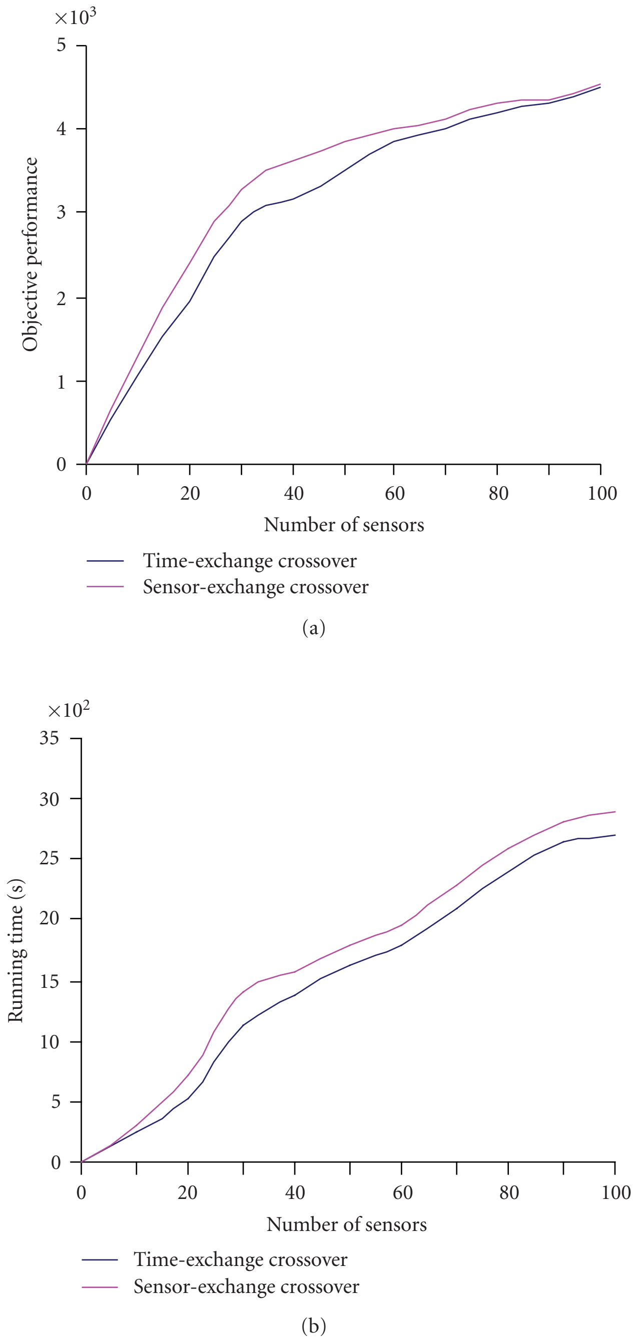


Figure 8: GA performance progress with number of iterations: (a) objective performance and (b) elapsed time.

5.2. GA-related results

In this section, we present results related to the GA algorithm. First, the algorithm convergence pattern is presented for the cases of single and multiple chromosomes. Then, the effect of crossover and mutation strategies on the solution quality is illustrated. In all of the experiments conducted in this section, sensors lifespan is generated based on a uniform distribution $U(1, T)$ and other parameters such as state-switching, mobility and mobility cost are generated randomly based on the lifespan of the sensors. Sensors reliability is also generated randomly based on a uniform distribution $R(0, 1)$. Figure 8 shows the objective performance and corresponding running time for a deployment problem with 100 zones, 50 sensors, and 12 time intervals. The value of the objective performance and the corresponding cumulative running time are recorded after every iteration. SE crossover operator and 100% mutation are used in these experiments. As shown in Figure 8(a), generating 10 chromosomes per iteration results in convergence at higher objective using less number of iterations. For instance, in the multiple chromosomes case, an objective of 24841 units is recorded at iteration 64. This value is achieved at iteration 156 in the single-chromosome case. On the other hand, the running time of multiple chromosomes is higher than the time recorded for the single-chromosome case. As shown in Figure 8(b), the running time in the single-chromosome implementation is almost 60% of that recorded in the multiple chromosomes implementation.

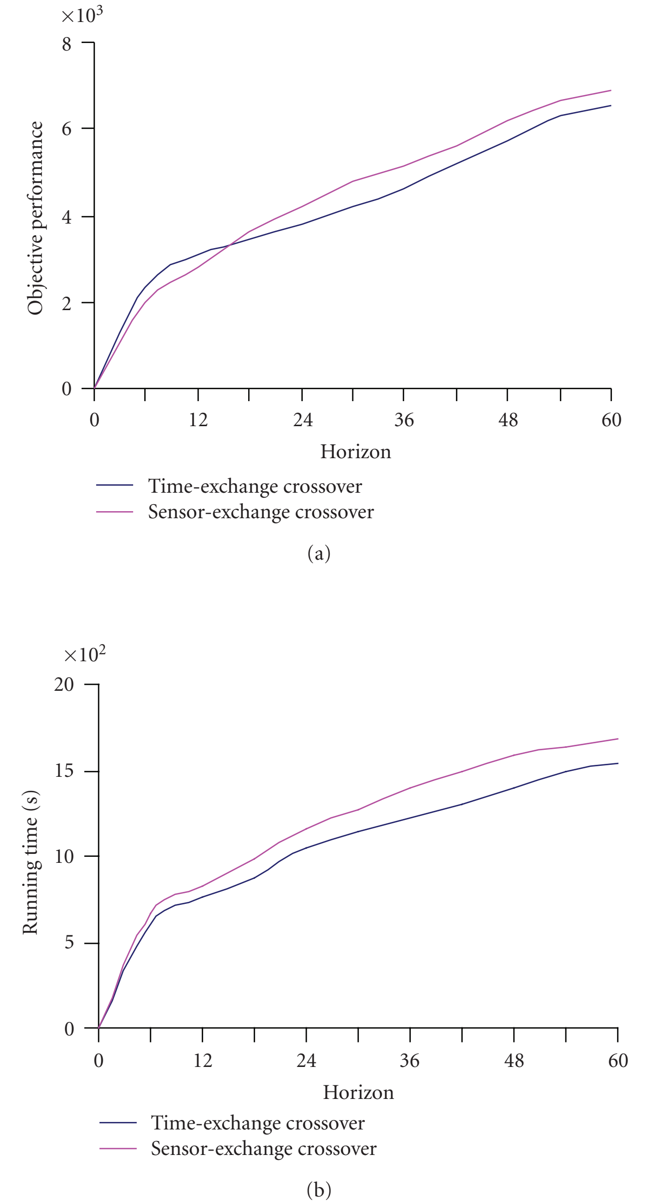
The GA performance associated with using different crossover and mutation strategies are also studied. The TE



(a)

(b)

FIGURE 9: A comparison between time-exchange and sensor-exchange crossover operators performance with increasing number of sensors: (a) objective performance and (b) running time.



(a)

(b)

FIGURE 10: A comparison between time-exchange and sensor-exchange crossover operators performance with increasing the horizon: (a) objective performance and (b) running time.

and SE crossover strategies are first compared. Two sets of experiments are considered. In the first set of experiments, 100 zones are observed for 12 time intervals. The problem size is increased in terms of number of sensors. In the second set of experiments, the problem size is increased through increasing the time horizon. The field size is 100 zones covered with 50 sensors. Figures 9 and 10 present objective value and running associated for both crossover strategies. As shown in Figure 9, SE crossover operator outperforms the TE operator. However, as the number of sensors becomes close to the number of zones, both operators recorded similar coverage performance. On the other hand, using SE crossover operator results in greater running time. Thus, for higher coverage performance, one might recommend using SE crossover operator as long as the number of sensors is less than the number of zones. Otherwise, TE is recommended from the running time point of view.

As the horizon length increases, the TE crossover strategy becomes more superior in terms of coverage performance. This is also associated with an increase in the running time. As illustrated in Figure 10, for horizons beyond

20 time intervals, the SE strategy yields higher coverage performance. A corresponding pattern is recorded for the algorithm running time. The time difference shown in Figures 9(b) and 10(b) returns to the additional processing time required for the SE operator to ensure feasibility of generated chromosomes.

As mentioned above, mutation in GA plays an important role in directing the solution towards different search spaces. Figure 11 illustrates the effect of using different mutation percentages on the average objective performance. Mutation percentage is measured as the ratio between number of altered genes and total chromosome length. A field of 100 zones is monitored for 12 units of time using 50 sensors. Ten chromosomes per iteration and TE crossover operator are used throughout these experiments. The mutation percentage ranges from 0% to 100%. The results show that as the mutation percentage increases, the objective performance increases. For instance, at 20% mutation rate, an objective of 9376 units is recorded. The objective increased to 10389 units at 100% mutation rate.

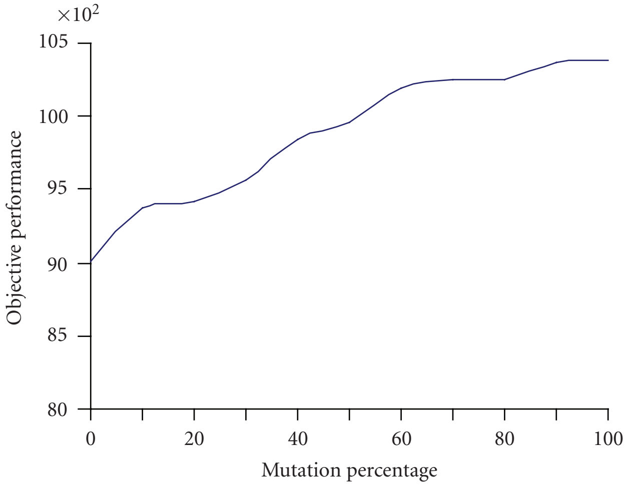


Figure 11: GA performance with different mutation percentages.

5.3. SA-related results

The SA objective function convergence pattern is presented in this section. For this purpose, the SA algorithm is applied for a problem of 100 zones, 12 time units, and 50 sensors. The heating rate is selected to be 0.99 and a sample of 300 iterations is recorded. The starting temperature is assumed to be 2 temperature units and the final temperature is set to 50. Sensors configurations are generated randomly based on uniform distribution as mentioned in Section 5.2. As shown in Figure 12, the algorithm starts by pivoting at solutions with low objective values since the acceptance probability of a new solution is initially high. This may lead the algorithm to fall in a local minimum such as the fall that occurred at iteration 154. Also, the incumbent value $\widehat{X}$ maintains the highest objective value which is 7939 reached at iteration 114. As the temperature increases, the acceptance probability is decreased and the chance of pivoting at low performance solutions decreases. This pattern is clearly showed starting at iteration 200.

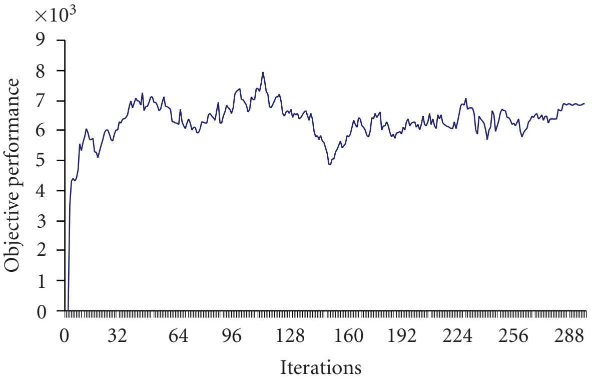


Figure 12: Objective function progress with number of iterations.

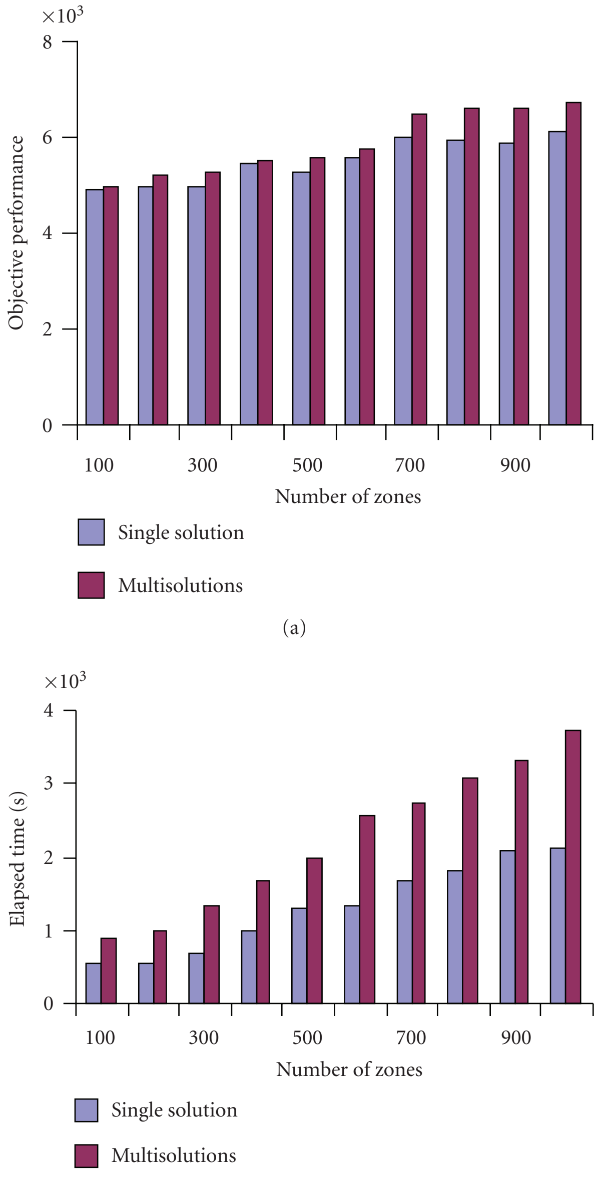
Two implementations are considered for the SA algorithm: single solution per iteration and multiple solutions per iteration. Multiple solutions per iteration algorithm direct the search into multiple search spaces. To compare these two implementations, a problem with 50 sensors, 12 time units and number of zones that range from 100 to 1000 zones is used. For the multiple solutions case, 10 solutions per iteration are generated. The starting and final temperatures are set to 2 and 1000, respectively. The temperature rate is assumed to be 0.99. As shown in Figure 13, based on the conducted experiments, the multiple solutions implementation seems to outperform the single solution. For example, for a problem with 1000 zones, the objective function of single solution implementation is 10% less than that of the multiple solutions implementation. This 10% improvement in the coverage performance is associated with about 40% increase in the running time.

5.4. Effect of deployment parameters

In this section, we study the effect of different deployment parameters on the performance of the genetic and simulated annealing algorithms. In the first set of experiments, we discuss the effect of the zones' weight variance on the performance of the developed algorithms. The second set of experiments examines the tradeoff between different sensors attributes such as reliability, lifespan, mobility, and state-switching. In addition, these parameters illustrate the effect of these attributes on the coverage performance of both GA and SA solutions. Throughout these experiments, the time-exchange crossover operator is used with 10 chromosomes per iteration. For SA, the starting and ending temperatures are assumed to be 2 and 50, respectively. The heat rate is set to 0.99. A sample of 300 iterations from both GA and SA is presented.

5.4.1. Effect of observations variance

A set of experiments is conducted to illustrate the effect of the observation weights' variance on the algorithms



(a)

(b)

Figure 13: A comparison between single and multiple solution(s) per iteration: (a) objective performance and (b) elapsed time.

performance. In these experiments, the results of the optimal solution mentioned in Section 3 are compared to the output of both GA and SA for small-scale problems due to the limitation of the optimal solution. A field of six zones is monitored for 12 time intervals using three sensors. The observation weights along the different zones are generated randomly using uniform distributions with different ranges. In addition, sensors parameters such as lifespan, state-switching, and mobility are generated using uniform distribution as mentioned in Section 5.2. Sensors' reliability is also generated uniformly within the range of $R(0, 1)$.

As illustrated in Figure 14, as the variance in the observation weights across the zones and time intervals increases, the approximation algorithms tend to produce solutions that are closer to the optimal ones. For example, using GA, a coverage performance of 75% is obtained when the zones' observation weights are set according to uniform distribution with a range of 25–125. This performance is improved to 90% when the range is increased to 200. These results are also confirmed using SA in which the performance of the algorithm is increasing with increasing the weight variance. For instance, 75% coverage is given when a variance range 50–150 is used, while the coverage percentage is increased to 85% when the variance range is increased to 200. As the variance increases the number of zone-time intervals with high observation weights decreases. In other words, important observations are concentrated in less number of zones and time intervals. In addition, the observation weights of these zone-time intervals represent high percentage of the observation weights of all considered zone and time intervals. Thus, these observations could be collected using less number of sensors' lifespan, which frees other sensors to collect other observations. Hence, the overall coverage performance is improved with the increase in the observation variance.

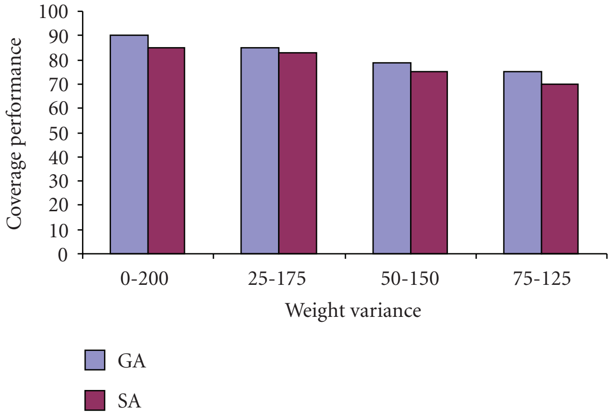


Figure 14: Effect of zones' weight variance on the coverage performance of GA and SA.

5.4.2. *Effect of state-switching and mobility*

In this section, we study the effect of sensors' state-switching and mobile capabilities on the coverage performance using GA and SA. Two sets of experiments are conducted in which 20 zones are monitored using five sensors for 12 units of time. Observation weights of the different zones are generated randomly based on uniform distribution $U(1, 200)$. In addition, sensors' reliability is generated uniformly within the range $(0, 1)$. In the first set of experiments, the effect of sensor state-switching limitation is examined. The effect of using sensors with mobile capabilities is assessed in the second set of experiments.

Figure 15 illustrates the coverage performance for sensors with different state-switching capabilities that range from 0% to 100% as ratio of the sensors lifespan. Based on the obtained results, using sensors with state-switching capabilities

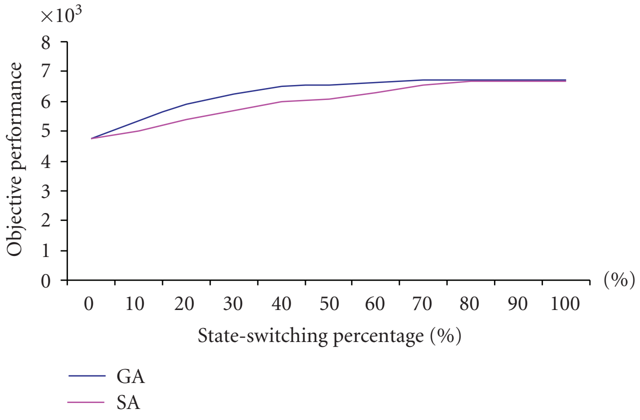


Figure 15: Effect of sensors' state-switching on the coverage performance of GA and SA.

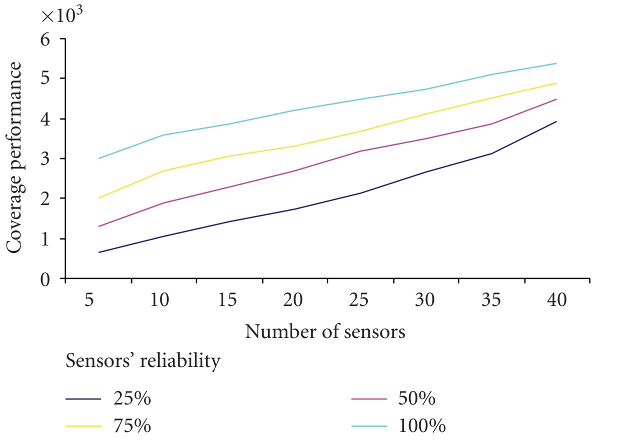


Figure 17: Effect of sensors' reliability with different numbers of sensors on the coverage performance of GA.

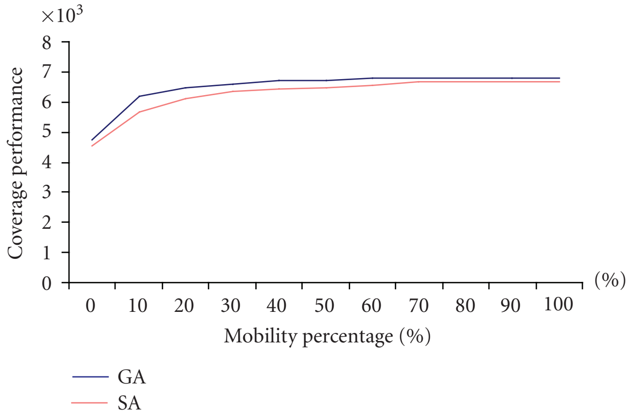


Figure 16: Effect of sensors' mobility on the coverage performance of GA and SA.

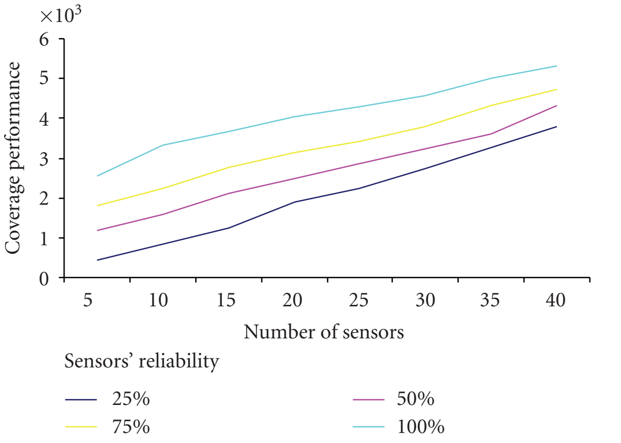


Figure 18: Effect of sensors' reliability with different numbers of sensors on the coverage performance of SA.

improves overall coverage performance in the monitored field. Sensors are able to save their lifespan to be used on high weights' zones at different time(s), and therefore enhance the overall coverage performance. For example, coverage performances of 5340 and 5000 coverage units are observed for 10% state-switching by GA and SA, respectively. These values are increased to about 6700 units using both GA and SA when 80% of the state-switching is used. Similarly, as illustrated in Figure 16, sensors with high mobility are able to move among the different zones to capture high-weights observations. When sensors' mobility is increased from 0% to 10%, the coverage performance improves by about 30 for the GA and by about 34 for the SA.

5.4.3. *Effect of lifespan and reliability*

In this section, we discuss the effect of sensors lifespan and reliability on the coverage performance. Using a problem configuration similar to the one presented in Section 5.4.2, Figures 17 and 18 illustrate the relationship between number of sensors and overall coverage performance for different levels of sensors reliability using GA and SA, respectively. As illustrated in the figures, the same coverage performance using less reliable sensors can be achieved only through increasing the number of used sensors. For instance, as shown in Figure 17, ten sensors with 100% reliability give the same coverage performance as that of 40 sensors with 25% reliability using GA. Similar results are obtained for the SA as shown in Figure 18.

The same pattern could also be observed when sensors with different lifespans are used. As the sensors lifespan decreases, more sensors will be needed to achieve a certain required coverage performance. As illustrated in Figure 19, to achieve a coverage performance of 3470 units using GA, five sensors are needed if the sensors' lifespan is equal to the monitoring horizon. This number jumps to about 35 sensors if the sensors lifespan is only 25% of the monitoring horizon. This result is also confirmed using SA as shown in Figure 20.

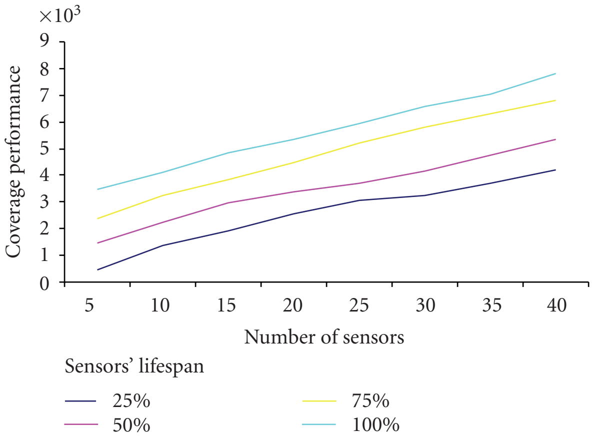


FIGURE 19: Tradeoff between sensors' lifespan and number of sensors using GA.

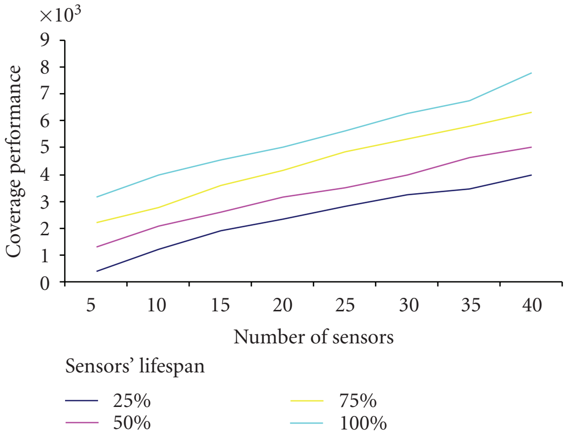


FIGURE 20: Tradeoff between sensors' lifespan and number of sensors using SA.

6. CONCLUSION AND FUTURE WORK

A modeling framework for the problem of deploying a set of heterogeneous sensors in a field with time-varying differential sensing requirements is presented. In this framework, the problem is formulated as mixed integer mathematical program with the objective to maximize coverage of a given field. A set of constraints is defined to ensure that sensors are not used beyond their capacity. Two metaheuristics are used to solve this problem. The first heuristic adopts a genetic algorithm (GA) approach while the second heuristic implements a simulated annealing (SA) algorithm. A set of experiments is used to illustrate capabilities of the developed models and to compare their performance. The results indicate that both GA and SA algorithms provide high-quality solutions in terms of objective function and running time. In addition, GA seems to yield better coverage performance than the SA. However, its running time was always higher in all considered experimental settings.

In this paper, sensors' lifetime, switching, and mobility capabilities are assumed to be deterministic. Their costs are also assumed to be given. For future research, these parameters could be related to a cost function that is based on the sensors' consumed energy. For example, the lifetime consumed energy may possibly associate with actual active duration of the sensor. The movement consumed energy of a sensor from one zone to another could be measured by the movement distance and the movement time. Furthermore, the state-switching energy could have a separate energy function for changing the sensor's state from on to off or vice versa.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their feedback, suggestions, and comments.

REFERENCES

[1] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA '02)*, pp. 88–97, Atlanta, Ga, USA, September 2002.

[2] M. Kuorilehto, M. Hännikäinen, and T. D. Hämäläinen, "A survey of application distribution in wireless sensor networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2005, no. 5, pp. 774–788, 2005.

[3] N. T. Nguyen, S. Venkatesh, G. West, and H. H. Bui, "Multiple camera coordination in a surveillance system," *Acta Automatica Sinica*, vol. 29, no. 3, pp. 408–422, 2003.

[4] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[5] M. Ilyas and I. Mahgoub, *Handbook of Sensor Networks*, CRC Press, Boca Raton, Fla, USA, 2005.

[6] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '01)*, vol. 4, pp. 2033–2036, Salt Lake, Utah, USA, May 2001.

[7] D. Li, K. D. Wong, Y. H. Hu, and A. M. Sayeed, "Detection, classification, and tracking of targets," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 17–29, 2002.

[8] V. Chvátal, "A combinatorial theorem in plane geometry," *Journal of Computorial Theory (B)*, vol. 18, pp. 39–41, 1975.

[9] J. O'Rourke, "Galleries need fewer mobile guards: a variation on Chvátal's theorem," *Geometriae Dedicata*, vol. 14, no. 3, pp. 273–283, 1983.

[10] K. Chakrabarty, S. S. Iyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Transactions on Computers*, vol. 51, no. 12, pp. 1448–1453, 2002.

[11] M. Cardei and J. Wu, "Coverage in wireless sensor networks," in *Handbook of Sensor Networks*, M. Ilyas and I. Mahgoub, Eds., CRC Press, Boca Raton, Fla, USA, 2004.

[12] V. Isler, S. Kannan, and K. Daniilidis, "Sampling based sensor-network deployment," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '04)*, vol. 2, pp. 1780–1785, Sendai, Japan, September-October 2004.

[13] X. Liu and P. Mahapatra, "On the deployment of wireless sensor nodes," in *Proceedings of the 3rd International Workshop on Measurement, Modeling, and Performance Analysis of Wireless Sensor Networks, in Conjunction with the 2nd Annual International Conference on Mobile and Ubiquitous Systems*, San Diego, Calif, USA, July 2005.

[14] W. Hu, C. Chou, S. Jha, and N. Bulusu, "Deploying long-lived and cost-effective hybrid sensor networks," in *Proceedings of the 1st Workshop on Broadband Advanced Sensor Networks (BaseNets '04)*, San Jose, Calif, USA, October 2004.

[15] J.-J. Lee, B. Krishnamachari, and C.-C. J. Kuo, "Impact of heterogeneous deployment on lifetime sensing coverage in sensor networks," in *Proceedings of the 1st Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON '04)*, pp. 367–376, Santa Clara, Calif, USA, October 2004.

[16] G. Wang, G. Cao, and T. F. La Porta, "Movement-assisted sensor deployment," *IEEE Transactions on Mobile Computing*, vol. 5, no. 6, pp. 640–652, 2006.

[17] S. Poduri and G. S. Sukhatme, "Constrained coverage for mobile sensor networks," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '04)*, pp. 165–171, New Orleans, La, USA, April-May 2004.

[18] A. Howard, M. J. Mataríc, and G. S. Sukhatme, "An incremental self-deployment algorithm for mobile sensor networks," *Autonomous Robots*, vol. 13, no. 2, pp. 113–126, 2002.

[19] S. S. Dhillon, K. Chakrabarty, and S. S. Iyengar, "Sensor placement for grid coverage under imprecise detections," in *Proceedings of the 5th International Conference on Information Fusion*, vol. 2, pp. 1581–1587, Washington, DC, USA, July 2002.

[20] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization in distributed sensor networks," *ACM Transactions on Embedded Computing Systems*, vol. 3, no. 1, pp. 61–91, 2004.

[21] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," in *Proceedings of the 22nd Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '03)*, vol. 2, pp. 1293–1303, San Francisco, Calif, USA, March-April 2003.

[22] J.-J. Lee, B. Krishnamachari, and C.-C. J. Kuo, "Node aging effect on connectivity of data gathering trees in sensor networks," in *Proceedings of the 60th IEEE Vehicular Technology Conference (VTC '04)*, pp. 4630–4634, Los Angeles, Calif, USA, September 2004.

[23] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '02)*, vol. 3, pp. 1567–1576, New York, NY, USA, June 2002.

[24] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *Proceedings of the Annual International Conference on Mobile Computing and Networking (MOBICOM '01)*, pp. 70–84, Rome, Italy, July 2001.

[25] D. Gage, "Command control for many-robot systems," in *Proceedings of the 19th Annual Technical Symposium and Exhibition of the Association for Unmanned Vehicle Systems (AUVS '92)*, Huntsville, Ala, USA, June 1992.

[26] C.-F. Huang and Y.-C. Tseng, "The coverage problem in a wireless sensor network," in *Proceedings of the 2nd ACM International Workshop on Wireless Sensor Networks and Applications (WSNA '03)*, pp. 115–121, San Diego, Calif, USA, September 2003.

[27] Z. Zhou, S. Das, and H. Gupta, "Connected K-coverage problem in sensor networks," in *Proceedings of the 13th International Conference on Computer Communications and Networks (ICCCN '04)*, pp. 373–378, Chicago, Ill, USA, October 2004.

[28] D. T. Lee and A. K. Lin, "Computational complexity of art gallery problems," *IEEE Transactions on Information Theory*, vol. 32, no. 2, pp. 276–282, 1986.

[29] A. GwiazdaTomaszDominik, *Genetic Algorithms Reference Volume I Crossover for Single-Objective Numerical Optimization Problems*, Tomasz Gwiazda, Lomianki, Poland, 2006.

[30] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Kluwer Academic, Boston, Mass, USA, 1989.

[31] L. Randy and S. Haupt, *Practical Genetic Algorithms*, John Wiley & Sons, New York, NY, USA, 2004.