Pesquisa Operacional (2026) 46: e299058 p.1-21
doi: 10.1590/0101-7438.2026.046.00299058
© 2026 Brazilian Operations Research Society
Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope
ARTICLES

A NEW HYBRID GENETIC ALGORITHM FOR MAXIMIZING AREA COVERAGE IN WIRELESS SENSOR NETWORKS

Nabil Boumedine[1*] and Sadek Bouroubi[2]

Received July 21, 2025 / Accepted October 28, 2025

ABSTRACT. Wireless sensor networks are becoming increasingly important in many fields, such as management and security. They enable the collection and transmission of large volumes of data from a specific area to a data center for processing and analysis. One of the most challenging problems in sensor deployment is determining the optimal placement of each sensor to maximize the coverage area. It has been demonstrated that this problem is NP-hard. Due to its huge complexity, various metaheuristics have been proposed to solve the coverage problem in wireless sensor networks. In this paper, we propose an efficient hybrid genetic algorithm to maximize the coverage area in wireless sensor networks. The proposed approach combines a genetic algorithm with an enhanced simulated annealing algorithm. The efficiency of the proposed algorithm has been tested in 15 benchmark instances taken from the literature and compared with state-of-the-art algorithms.

Keywords: wireless sensor networks, maximum coverage problem, simulated annealing, genetic algorithm, monte carlo method, overlapping fitenss.

1 INTRODUCTION

A wireless sensor network (WSN) is a system of sensors distributed throughout a given space that can communicate with each other wirelessly to monitor physical or environmental states (such as temperature, humidity, pressure, movement, military mission and so on (Gage (1992); Barroca et al. (2013)). For many reasons, wireless sensor networks (WSNs) are one of the most powerful technologies of the 21st century (Lopez-Ardao et al. (2021)). Due to their exceptional flexibility, low cost, and ability to function in remote or difficult-to-access areas, WSNs have emerged as a revolutionary technology for data collection, automation, monitoring, and control (Zhou et al. (2008)). These networks consist of small sensor nodes that can be deployed in large

*Corresponding author
[1]L'IFORCE Laboratory, UMBB, Department of Mathematics, Independence Avenue, 35000, Boumerdes, Algeria – E-mail: boumedine_nabil.fs@univ-boumerdes.dz – https://orcid.org/0000-0002-1466-310X
[2]L'IFORCE Laboratory, USTHB, Department of Operational Research, Bp 32 El Alia, 16111 Algiers, Algeria – E-mail: sbouroubi@usthb.dz – https://orcid.org/0000-0002-0691-6189

quantities and operate autonomously, collecting data and transmitting it to a central base station for further processing and analysis. Wireless sensor networks (WSNs) have become an important topic of academic research, mainly due to the multiple challenges associated with their design and implementation. These challenges are largely related to the stringent constraints involved in these systems, including limited energy resources, restricted computing and memory capacities, and limited bandwidth. In addition, the specific and often restrictive requirements of the targeted applications add an additional layer of complexity. In response to these growing difficulties, methods derived from operations research and optimization are increasingly being used to design efficient solutions (Gogu et al. (2012)). In WSNs, coverage optimization focuses on maximizing the area or region that sensors can monitor while also efficiently using resources such as energy and processing (Akyildiz (2002); Gogu et al. (2012)). This is a key challenge, as sensors typically have limited battery and computing resources, and network efficiency is essentially dependent on maximizing coverage without exceeding the capacity of these resources (Yick et al. (2008); Gogu et al. (2012)). Optimal sensor deployment to maximize coverage area in wireless networks is an interesting and complex optimization problem because of the associated constraints and the limited number of sensors (Gogu et al. (2012)). Several models are designed to solve this problem, each based on different criteria. This study is based on a model that aims to maximize the coverage of an area in wireless sensor networks (WSNs). More specifically, this model seeks to determine the optimal deployment of a set of sensors, each of which has particular sensing ranges to obtain the greatest possible coverage. In this model, we define the coverage area of a given sensor simply as the area of a disk centered on the sensor, the radius of which corresponds to its sensing range. Yourim Yoon et al. were the first to present this problem and prove that it was NP-hard (Yoon & Kim (2013)). In this paper, we introduce a new and effective method called HGA-ISA, which combines the Genetic Algorithm (GA) with an Improved Simulated Annealing (ISA) technique to solve the problem of maximizing the coverage area in Wireless Sensor Networks (WSNs). The objective is to determine the optimal deployment (i.e., sensor placement) to ensure maximum area coverage. The proposed algorithm integrates multiple search strategies to maintain a balance between global exploration and local exploitation, thus improving both the quality of the solutions and the convergence rate of the algorithm.

The remainder of this paper is organized as follows: the following section introduces the network model and provides a formal definition and mathematical formulation of the maximum coverage problem in Wireless Sensor Networks (WSNs). Section 3 presents related work, including various methods and fitness functions proposed in the literature to address this problem. In Section 4, we introduce the proposed HGA-ISA algorithm, providing a detailed description of each integrated search strategy. Section 5 details the experimental evaluation, where we compare HGA-ISA with five state-of-the-art algorithms using a set of fifteen benchmark instances, followed by a thorough analysis of the results. Finally, Section 6 presents our conclusions and outlines potential directions for future research.

2 NETWORK MODEL AND PROBLEM DEFINITION

In the present paper, we address the problem of maximum area coverage in heterogeneous sensor networks, as described in (Yoon & Kim (2013)). This model divides a given number n of sensors with different detection ranges $r_i, i = 1,\ldots,n$ into k types. Sensors with the same detection range are considered to be of the same type. Sensors must be installed in a given 2D domain A. The objective is to find locations for all sensors (that is, $\{(x_i,y_i), i = 1,\ldots,n\}$ that maximize the coverage area over A. The following tables show the different notation used in our mathematical formulation for this problem:

Table 1 – Notations.

Symbol	Definition
n	The number of sensors
k	The number of sensor types
n_j	The number of sensors of the same type j, $j = 1,\ldots,k$, with $\sum_{j=1}^{k} n_j = n$
r_j	The sensing radius of sensors of the same type j, $j = 1,\ldots,k$
$A_{W\times H}$	H, W: the height and width of the 2D domain A, respectively

The aim is to find the best location in A for each sensor to ensure maximum coverage of the area A (denoted COV(A)). The objective function of this optimization problem can be expressed mathematically using the following formula (Yoon & Kim (2013)):

$$\text{COV(A)} = \text{area}\left(\bigcup_{j=1}^{k}\bigcup_{i=1}^{n_j} c_{r_j}(x_i^j, y_i^j)\right) \quad \rightarrow \max \tag{2}$$

where, $c_{r_j}(x_i^j, y_i^j)$ corresponds to a circle with center at (x_i^j, y_i^j) and radius r_j, and $area(.)$ is the area coverage generated by all sensors placed in area A.
In equation (2), the term $\bigcup_{i=1}^{k}\bigcup_{j=1}^{n_i} c_{r_j}(x_i^j, y_i^j)$ represents the total area covered by all n sensors. However, our primary concern is the coverage within the area A. Therefore, the effective area coverage is determined by the intersection of the total coverage and A.

In this study, we adopt the Boolean disk coverage model used in (Yoon & Kim (2013)). Let $d(x_s, y)$ represent the Euclidean distance between a sensor s located at x and a point y, and let r_s represent the sensing range of the sensor s. The Boolean coverage function between x_s and y is defined as:

$$f(d(x_s,y)) = \begin{cases} 1, & \text{if } d(x_s,y) \leq r_s, \\ 0, & \text{otherwise.} \end{cases}$$

For example, Figure 1 illustrates two possible configurations for 17 sensors (sensors of the same color are of the same type and range). In the first configuration (Figure 1.*a*), there is no intersection between the circles and no part of any circle extends beyond the designated zone. In

this case, we can easily calculate the optimal coverage by:

$$COV(A) = \sum_{i=1}^{n} \pi r_i^2.$$

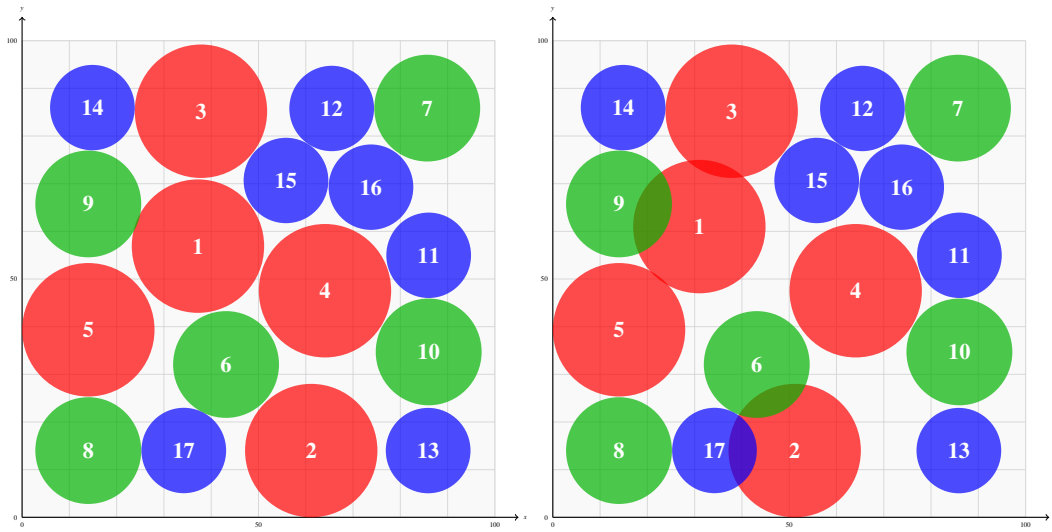


Figure 1 – Illustration of two configurations for 17 sensors: *a*) Optimal configuration. *b*) Non-optimal configuration.

In contrast, the second configuration (Figure 1.*b*) presents a more complex scenario, as the circles overlap and intersect the boundaries of the area, making it more challenging to determine the exact coverage. As a result, several fitness functions have been proposed in the literature to evaluate the solutions that we present in the next section.

3 RELATED WORKS

Since the first formulation of Maximum Coverage Sensor Deployment Problem (MCSDP) in (Yoon & Kim (2013)), various heuristics, metaheuristics, and fitness functions have been developed to address this challenge. The first genetic algorithm applied to this problem was introduced in (Yoon & Kim (2013)). In this study, the authors present an efficient genetic algorithm that employs the BLX-α crossover operator and the Gaussian mutation to generate new solutions (ie offspring) in each generation. Because the overlap of circles with different radii complicates the derivation of an exact expression for the area, the authors resort to using the Monte Carlo method as a fitness function to evaluate the candidate solutions. In (Ly et al. (2015)), the authors introduced a novel genetic algorithm that incorporates several enhancements over the standard version. One key improvement was the introduction of a new concept, overlapping, in the fitness function, designed to reduce execution time associated with the Monte Carlo method. In addi-

tion, they employed an effective heuristic technique to generate high-quality solutions for the initial population and implemented dynamic mutation. The experimental results demonstrated that their proposed algorithm outperformed existing approaches in terms of computational efficiency, solution quality, and stability. In (Hanh et al. (2018)), the authors propose two variants of the PSO algorithm, namely Particle Swarm Optimization (PSO) and Democratic PSO (DPSO), which are designed for faster convergence speeds. The results demonstrate that these new algorithms outperform the best existing genetic algorithm from previous research, both in terms of execution speed and overall performance. In (Binh et al. (2018)), the Cuckoo Search and Chaotic Flower Pollination algorithms were used to solve this problem. Both algorithms are based on the Lévy flight distribution to generate new solutions. The combination of their simplicity and their ability to effectively escape local minima makes these metaheuristics both powerful and stable. In (Hanh et al. (2019)), the authors introduced a novel genetic algorithm known as MIGA, which offers several significant contributions and improvements. These include a new initialization heuristic, an improved and more stable fitness function to evaluate candidate solutions, a hybridization of two distinct crossover operators, and the incorporation of a local search mechanism based on the Virtual Force Algorithm. In a more recent study (Yoon & Kim (2021)), the authors introduced an efficient method for estimating coverage, grounded in a novel theoretical analysis. They proposed a high-performing memetic algorithm (MA) enhanced by a new local search strategy, specifically designed to improve search efficiency. Experimental results showed that their approach consistently outperformed state-of-the-art algorithms in both computational time and solution quality.

4 THE PROPOSED METHOD

In this section, we present our proposed algorithm for solving the problem of maximum coverage area. The balance between diversification and intensification mechanisms is a big part of how well metaheuristics work. Achieving an optimal balance allows the algorithm to explore the solution space efficiently (Lee & Choi (2011); Yang et al. (2014)). Based on this theory, we developed an efficient hybrid algorithm that combines the standard genetic algorithm with an enhanced simulated annealing algorithm.

The proposed HGA-ISA algorithm starts with an initial population p containing m random generated solutions. In each generation t, a new offspring population$p(t)$ is created through a reproduction process that includes selection, crossover, and mutation operations. To enhance the overall quality of the solutions, we apply an Improved Simulated Annealing (ISA) algorithm at the end of each generation. Rather than refining every individual, we probabilistically select a subset of solutions from the current population $p(t)$, each with a selection probability of τ. The ISA algorithm then uses these selected individuals as starting points for a local focused search, exploring their neighborhoods to identify potentially better solutions. This selective application of local refinement improves the quality of the solution while maintaining a reasonable computational cost. In the ISA, the new solutions are generated using a Lévy flight distribution. Lévy flights, which are a type of random walk with heavy-tailed distributions, enable the algorithm to

make long-range moves in the solution space. This ability to take large steps helps the algorithm escape local minima and explore a wider portion of the solution space more effectively. By combining Lévy flights with simulated annealing (SA), the algorithm achieves a better balance between exploration (escaping local optima) and exploitation (refining promising solutions), which improves both search efficiency and the quality of the final solution (Loucera et al. (2017); Izci et al. (2022); Chen et al. (2023)). The best m solutions from $p(t-1)$ and $p(t)$ are selected to form the parent population for the $t+1$ generation. The proposed algorithm is outlined in Algorithm 1 and described in detail below.

4.1 Representation of Individuals and Initialization of the Population

Given n sensors, a feasible solution for the addressed problem can be represented by a set of coordinates $\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$ on the considered area $A=[0,W]\times[0,H]$, where the location of a given sensor i is defined by the coordinate (x_i,y_i). Our algorithm begins with a random initial solution, where the position of each sensor is randomly generated within area A.

$$(x_i,y_i)=U(H-r_i,W-r_i),\quad i=\overline{1:n},$$

where

1. r_i is the radius of the sensor i.
2. U represents a random value drawn from the uniform distribution in the interval $[H-r_i,W-r_i]$.

Figure 2 presents an illustrative example of the deployment of $n=7$ sensors of three types.

1. Type 1: $r_1=2$ (green color).
2. Type 2: $r_2=3$ (red color).
3. Type 3: $r_3=5$ (blue color).

4.2 Evaluation

To evaluate the quality of the generated solutions, we used the fitness function proposed in (Ly et al. (2015)). This is based on the idea that coverage is maximized when overlap is kept to a minimum, both between sensors and between sensors and the boundaries of the area being monitored. Compared with the Monte Carlo method, this approach is more efficient regarding computational complexity. In our study, the Monte Carlo method is only used in the final evaluation stage to estimate the coverage provided by the best solution. Consider the solution $O=(o_1,o_2,\ldots,o_n)$

Algorithm 1 Suggested hybrid HGA-ISA algorithm.

Input: A list of n sensors and their sensing ranges $\{r_1, r_2, \ldots, r_n\}$.

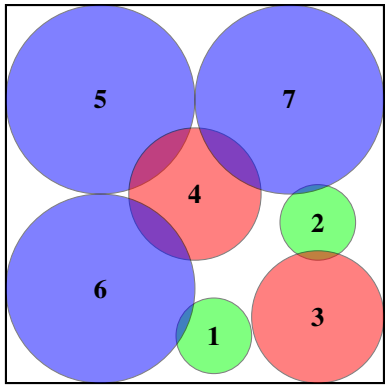
Output: The localization $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ of sensors in the area A.

Begin
Initialization: $P \leftarrow$ Generate a initial population of m random solutions.
Evaluate the solutions using the overlapping fitness.
while the stop criterion is not checked **do**
  Generate a new population P_{new} by the flowing steps:
  $k \leftarrow 0$, $P_{new} \leftarrow \emptyset$;
  while $k \leq m$ **do**
    $(p_1, p_2) \leftarrow$ Select two parents from P using the tournament selection technique.
    $u \leftarrow$ generate a random value in [0, 1].
    if $u < p_c$ **then**
      $(o_1, o_2) \leftarrow$ two solutions(offsprings) obtained by applying crossover operator to the selected parents.
    else
      $(o_1, o_2) \leftarrow$ two solutions(offsprings) obtained by applying mutation operator to the selected parents.
    $u_1 \leftarrow$ generate a random value in [0, 1].
    if $u_1 < \tau$ **then**
      $(o_1', o_2') \leftarrow$ Improve the quality of the obtained offsprings (o_1, o_2) using the suggested ISA given in algorithm 2.
    $P_{new} \leftarrow P_{new} \cup \{(o_1', o_2')\}$.
    $k \leftarrow k + 1$.
  end while
  $P \leftarrow$ the m best solution of $P \cup P_{new}$.
  Applying the suggested rotation operator to P.
end while
end

with $o_i = (x_i, y_i)$ the location of the sensor i in A. The quality of O is calculated as follows (Ly et al. (2015)):

$$\mathrm{OV}(O) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \mathrm{OS}(o_i, o_j) + \sum_{i=1}^{n} \sum_{K=1}^{4} \mathrm{OB}(o_i, b_k), \tag{1}$$

where



Sensor i	**Position :** (x_i, y_i)	r_i
1	(11, 2.5)	2
2	(16.5, 8.5)	2
3	(16.5, 3.5)	3
4	(10, 10)	3
5	(5, 15)	5
6	(5, 5)	5
7	(15,15)	5

Figure 2 – An illustrative example showing the deployment of $n = 7$ sensors, categorized into three types, within the area $[0,20] \times [0,20]$.

- The function $OS(s_i, s_j)$ determines the overlap between the coverage areas of two sensors i and j and is calculated as follows:

$$\mathrm{OS}(o_i, o_j) = \begin{cases} 0 & \text{if } d(o_i,o_j) \geq r_i + r_j, \\ \gamma_{i,j}\Delta_{i,j} - d(o_i,o_j) & \text{if } r_i - r_j \leq d(o_i,o_j) < r_i + r_j, \\ \beta_{i,j}\min(r_i, r_j) & \text{if } d(o_i,o_j) < |r_i - r_j|. \end{cases} \tag{2}$$

  With $d(o_i,o_j)$ denotes the Euclidean distance between o_i and o_j and $\Delta_{i,j} = r_i + r_j - d(o_i,o_j)$.

- The function $\mathrm{OB}(s_i, b_m)$ calculates the overlap between the coverage area of the sensor i and the boundary b_m, $b_m \in \{(0,0),(0,H),(0,W),(H,W)\}$ of the considered area $A = [0,H] \times [0,W]$.

$$\mathrm{OB}(o_i, b_m) = \begin{cases} r_i \cdot (r_i - d(o_i,b_m)) & \text{if } d(o_i,b_m) < r_i, \\ 0 & \text{else.} \end{cases} \tag{3}$$

In equation 2 $\gamma_{i,j}$ is given by:

$$\gamma_{i,j} = \frac{r_i + r_j}{\max(r_1, r_2, \ldots, r_n)} \cdot \frac{\min(r_i, r_j)}{\max(r_i, r_j)},$$

$\beta_{i,j}$ is a similar coefficient to γ but is used when the coverage of one sensor i is included in another sensor j, indicating a maximum overlap. Therefore, $\beta_{i,j}$ should be greater than the maximum value of $\gamma_{i,j}$. The fitness $f(O)$ of a given solution O is :

$$f(O) = \frac{1}{OV(O)}.$$

4.3 Selection

In this work, we used tournament selection to identify parents for the reproduction phase (i.e., crossover and mutation operations).

This selection method is very popular because it is easy to implement and takes less time than roulette selection, which requires sorting the population by fitness and calculating the selection probability for each individual at each iteration (Goldberg & Deb (1991)).

It consists of selecting g random solutions from the m existing in the current population and Selecting the best individuals as parents (i.e., those with the highest fitness values), as illustrated in Figure 3.

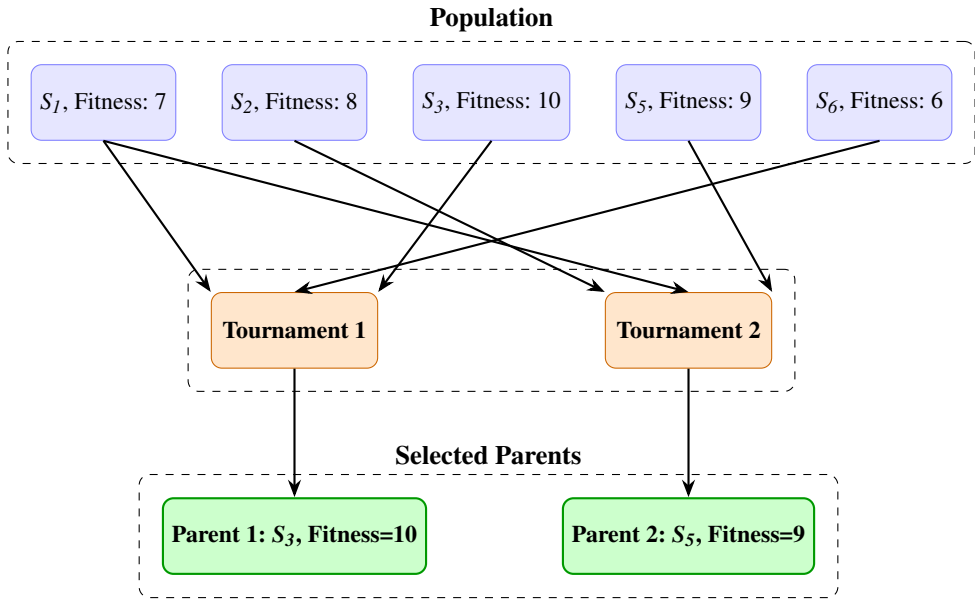


Figure 3 – An illustrative example of tournament selection: two selected parents for a population of six solutions and tournament size $g = 3$.

4.4 Crossover

We use the uniform crossover operator during the reproduction phase. In this type of crossover, the genes which represent the sensor positions in our problem are randomly selected from the two selected parents ,P_1, P_2 based on a fixed probability p. Specifically, for each gene position, a random number u is generated: if $u \geq p$, the corresponding sensor position in the first offspring O_1 is taken from the second parent P_2, and vice versa for the second offspring O_2 as we show in the illustrative example given in Table 2. This approach enhances genetic diversity by more effectively integrating traits from both parents into the offspring (Syswerda et al. (1989); Burjorjee (2013)).

Table 2 – An illustrative example of the uniform crossover operator applied to two parents solutions with a fixed probability $p = 0.5$.

Gene	P1	P2	u	O1	O2
1	(1,2)	(2,1)	0.7	(1,2)	(2,1)
2	(3,4)	(4,3)	0.3	(4,3)	(3,4)
3	(5,6)	(6,5)	0.6	(5,6)	(6,5)
4	(7,8)	(8,7)	0.4	(8,7)	(7,8)
5	(9,10)	(10,9)	0.8	(9,10)	(10,9)
6	(11,12)	(12,11)	0.2	(12,11)	(11,12)

4.5 Mutation Operator

To increase population diversity, we apply a mutation operator to each generation of the HGA-ISA. We randomly select a sensor i from a solution and move it to a new, randomly chosen position. This small move allows the algorithm to explore new areas of the solution space.

4.6 The improved simulated annealing algorithm based on Lévy flights

The improved simulated annealing (ISA) algorithm that we propose aims to optimize child solutions generated by genetic operators, namely crossover and mutation. These solutions are introduced as initial solutions in the ISA algorithm, which seeks to improve them iteratively until a predefined final temperature T_f is reached. In the suggested ISA, the temperature T gradually decreases over time based on a cooling rate α. At each iteration, the temperature is updated using the following formula:

$$T_{i+1} = \alpha \times T_i,$$

where α is a constant between 0 and 1. A value of α close to 1 results in slower cooling, allowing the algorithm to explore the search space more thoroughly. In contrast, a smaller α leads to faster cooling, which can speed up convergence but also increases the risk of premature convergence, potentially preventing it from reaching a better global solution. At each iteration, a new solution s' is generated from the current solution s by applying Lévy flights. This strategy enables the search space to be explored efficiently and limits the risk of stagnation in local optima. If the resulting solution s' is of higher quality than s (i.e., $f(s) \leq f(s')$), it is accepted directly. Otherwise, it can also be accepted with a probability determined by the Boltzmann distribution, thus promoting a good compromise between exploration and exploitation.In addition, to enhance the diversity of solutions explored and overcome stagnation situations, a perturbation operator has been introduced. This is activated when no improvement in the solution is observed after q successive iterations.The overall operation of the proposed algorithm is summarized in Algorithm 2 and will be described in more detail in the next section.

Algorithm 2 Improved Simulated Annealing Algorithm (ISA).

Input: An initail solution s.
Output: The best improvement achieved for s.

Begin
    Initialization:
    Evaluate $f(s)$ using the fitness function given in equation 2;
    $T_0 \leftarrow$ Initial temperature;
    $T_f \leftarrow$ Final temperature;
    $\alpha \leftarrow$ Cooling coefficient, where $\alpha \in [0,1]$;
    While $T_f < T_i$ **do:**
        Generate a new solution s' in using Lévy flights given in equation 4;
        Evaluate $f(s')$;
        $\Delta f = f(s) - f(s')$;
        If $(\Delta f < 0)$ **then**
            $s \leftarrow s'$;
            $f(s) \leftarrow f(s')$;
        Else
            $i \leftarrow i+1$;
            Generate a random number u in $[0,1]$;
            If $(u < e^{-\frac{\Delta f}{T_i}})$ **then**
                $s \leftarrow s'$;
                $f(s) \leftarrow f(s')$;
            End If
        End If
        If $(i = q)$ **then**
            Generate a new solution s' by applying the perturbation
            operator to s;
            $i \leftarrow 0$;
        End If
        $T_{i+1} \leftarrow \alpha \times T_i$;
    End While
End Algorithm

4.6.1 Lévy Flights

In the proposed ISA algorithm, at each iteration i, a new solution s_{new} is generated by updating the sensor positions of the current solution s_i using the following equation (Yang & Deb (2009)):

$$s_{new} = s_i + w \cdot \text{Lévy}(\beta), \quad 0 < \beta \leq 2, \tag{4}$$

where

1. The step size, $w > 0$, is often selected based on the characteristics of the optimization problem. In the literature, α is generally considered equal to 1.
2. Lévy(β) corresponds to the step length taken from the Lévy distribution.

A random step length $L(\beta)$ can be generated from the Lévy distribution using Mantegna's algorithm (Mantegna (1994)), which defines the step as:

$$L(\beta) = \frac{u}{|z|^{1/\beta}},$$

where $u \sim \mathcal{N}(0, \sigma_u^2)$, $v \sim \mathcal{N}(0, \sigma_z^2)$, are two random numbers generated from normal distributions. The standard deviations $\sigma_u$and σ_v for the two distributions are given by the following formulas:

$$\sigma_u = \left[\frac{\Gamma(1+\beta) \cdot \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \cdot \beta \cdot 2^{(\beta-1)/2}} \right]^{1/\beta}, \quad \sigma_z = 1. \tag{5}$$

The main idea behind using lévy flights to generate new solutions in the ISA algorithm is that lévy flights have a good balance between exploration (through long jumps) and exploitation (through short steps). A Lévy flight is a random walk in which the step length follows a heavy-tailed probability distribution. In other words, most of the steps are small, but sometimes the algorithm takes a much larger step. Such long jumps significantly improve the ability of the ISA to escape from local optima and explore new regions in the search space.

4.6.2 Perturbation operator

We introduce this operator to increase diversity in the search process of the ISA algorithm. If the best solution remains unchanged after k consecutive iterations, we apply the perturbation operator. This operator randomly selects two sensors of different types (i.e., with different radii) and swaps their positions. Figure 4 presents an illustrative example in which the sensor located at $(3;4)$ is exchanged with the one at $(5;1)$.

4.7 Rotation operator

This operator is used to introduce more diversity into the search process.

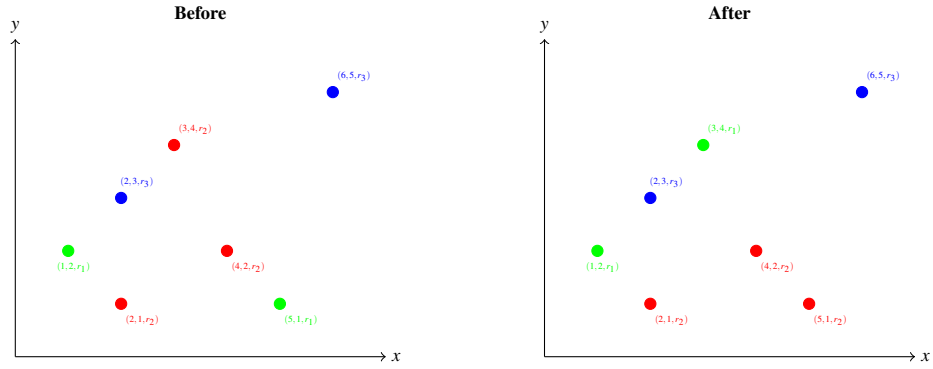


Figure 4 – Perturbation operator applied to two selected sensors with different radii highlighted in red and green.

As illustrated in Figure 5, the rotation operation modifies the sensors' positions without changing the quality of the solution. However, this operator plays a vital role in the crossover and mutation operators (Boumedine & Bouroubi (2022)). For example, consider that s_1 and s_2 are two selected solutions and that $s_1^{'}$ is the solution obtained by applying a rotation from $\beta^{\circ} \in \{90^{\circ}, 180^{\circ}, 270^{\circ}\}$ to s_1. The combination of $s_1^{'}$ and s_2 produces solutions that are different from those generated by the combination of s_1 and s_2.

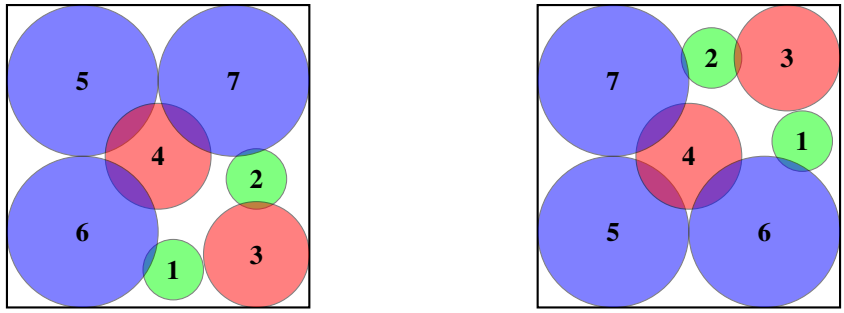
5 EXPERIMENTAL RESULTS

To evaluate the performance of our proposed algorithm, HGA-ISA, we implemented it in Python on an Acer- Intel Core i5 processor with 6 GB of RAM. The benchmark data sets used in this study are taken from (Yoon & Kim (2013); Ly et al. (2015)) and are detailed in Table 3.

These datasets include:

- A**:** The dimensions of the considered surveillance region, which is fixed at $A = 100 \times 100$ for all instances.
- **Sensor Types and Radii:** Three sensor types (n_1, n_2, n_3) with corresponding sensing radii (r_1, r_2, r_3), defined as:
  - $r_2 = 0.8 \times r_1$
  - $r_3 = 0.8 \times r_2$
- **Sensor Tightness Ratio (α):** This is defined as the ratio of the total coverage area achieved by all sensor nodes to the area (A).

To compare the proposed algorithm with state-of-the-art methods, we employed Monte Carlo simulations to evaluate the final solution obtained for each instance. This evaluation involves



(a) Initial Solution

(b) After rotated by 90°

(c) After rotated by 180°

(d) After rotated by 270°

Figure 5 – An illustrative example of four equivalent solutions generated by rotating the initial solution by 90°, 180°, and 270° respectively.

generating L random points within the considered area and computing the coverage ratio, defined as the number of points covered by the deployed sensors divided by the total number L of generated points. This ratio is then multiplied by the surface area of the region (i.e., $W \times H$) to estimate the total coverage.

The parameter settings of the proposed algorithm are summarized in Table 4.

Based on 15 independent runs per instance (see Table 3), Table 5 presents the average coverage achieved by the proposed algorithm, along with results taken from five state-of-the-art algorithms: Genetic Algorithm (column GA (Yoon & Kim (2013))), an Improved Genetic Algorithm (column IGA (Ly et al. (2015))), two versions of Particle Swarm Optimization (columns PSO and DPSO (Hanh et al. (2016))), the Chaotic Flower Pollination Algorithm (column CFPA (Binh et al. (2018))), an improved Cuckoo Search Algorithm (column ICS Binh et al. (2018)), and the Memetic Algorithm (column MA (Yoon & Kim (2021))).

As shown by the results, the proposed algorithm consistently performs better than state-of-the-art methods in most of the tested instances, with particularly notable improvements over the standard Genetic Algorithm. The results show that PSO, ICS, and MA are the most competitive algorithms compared to the proposed HGA-ISA in terms of performance.

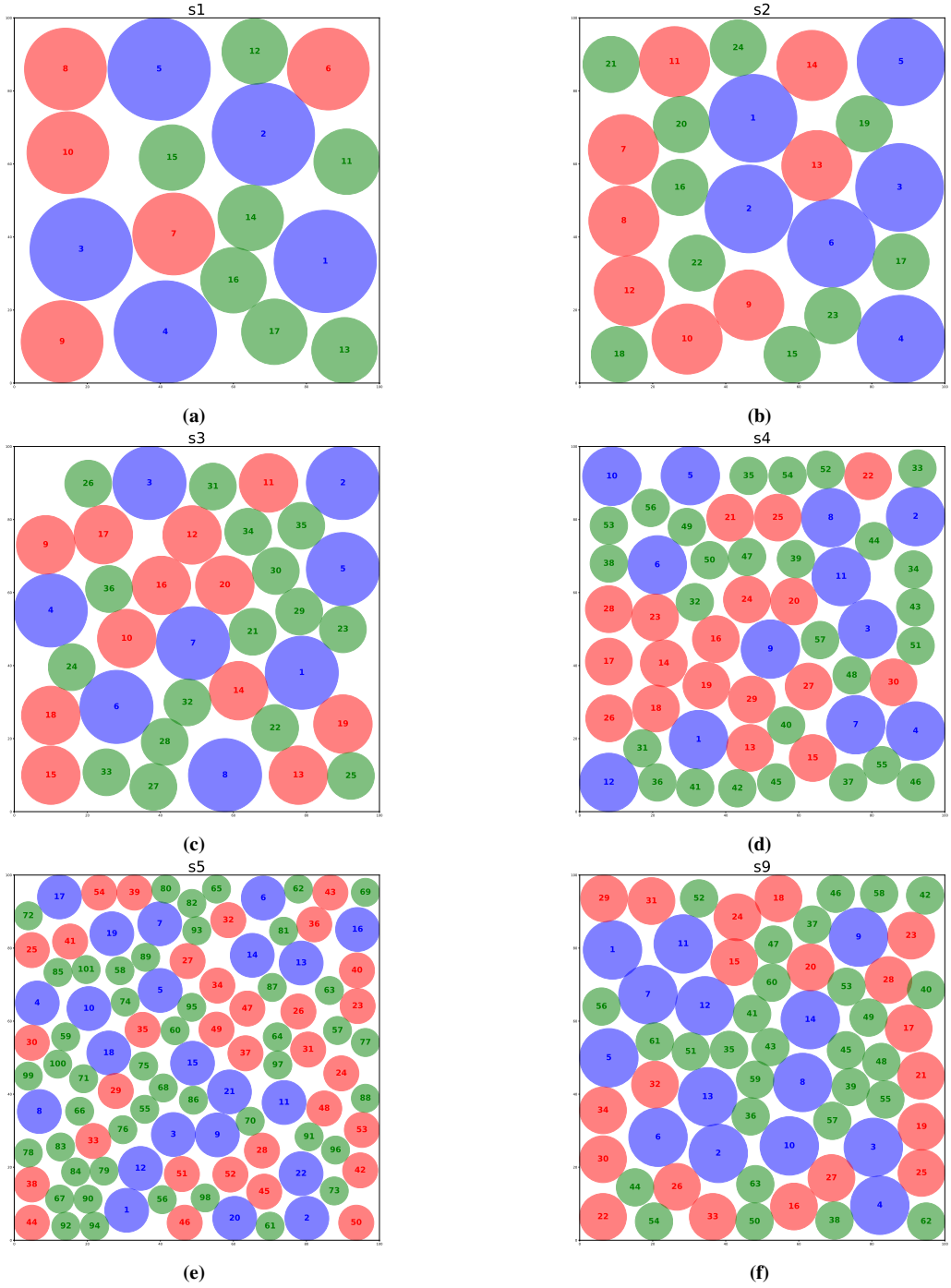


Figure 6 – The best solution obtained by the suggested HGA-ISA for the instances $S1$, $S2$, $S3$, $S4$, $S5$, and $S9$.

Table 3 – The set of benchmark instances used in our study to evaluate the performance of our proposed algorithm HGA-ISA.

Instances	r_1	n_1	r_2	n_2	r_3	n_3	n	α
$S1$	14.00	5	11.20	5	8.96	7	17	0.68
$S2$	12.00	6	9.60	8	7.68	10	24	0.69
$S3$	10.00	8	8.00	12	6.40	16	36	0.70
$S4$	8.00	12	6.40	18	5.12	27	57	0.70
$S5$	6.00	22	4.80	32	3.84	47	101	0.70
$S6$	14.00	5	11.20	6	8.96	10	21	0.80
$S7$	12.00	6	9.60	9	7.68	14	29	0.79
$S8$	10.00	9	8.00	13	6.40	19	41	0.79
$S9$	8.00	14	6.40	20	5.12	29	63	0.78
10	6.00	25	4.80	36	3.84	55	116	0.80
$S11$	14.00	6	11.20	7	8.96	10	23	0.90
$S12$	12.00	7	9.60	11	7.68	14	32	0.89
$S13$	10.00	11	8.00	14	6.40	21	46	0.90
$S14$	8.00	16	6.40	23	5.12	34	73	0.90
$S15$	6.00	28	4.80	41	3.84	61	130	0.90

Table 4 – Parameter settings of the proposed algorithm.

Parameter	**Value**
Population size	$m = 15$
Number of points generated in Monte Carlo method	$l = 1{,}00{,}000$
Perturbation operator threshold in ISA	$q = 100$
Step length of Lévy flights	$\beta = 1.5$
Number of selected solutions in tournament selection	$g = 6$
Uniform crossover ratio	$p = 0.3$
Initial temperature in ISA	$T = 1000$
Cooling rate in ISA	$\alpha = 0.98$
The application rate of ISA	$\tau = 0.25$
Maximum number of generations	200

To make the comparison with state-of-the-art algorithms more meaningful, we also implemented a Genetic Algorithm in (Ly et al. (2015)), and the cuckoo search algorithm presented in (Binh et al. (2018)). All three algorithms were run with the same initial population of 40 solutions. For CSA and GA, we used the same parameter values that were reported in (Ly et al. (2015)) and (Binh et al. (2018)). The Table 6 shows the average, execution time and standard deviation (std) values obtained from 30 independent runs.

Table 5 – The average results obtained from 15 independent runs: a comparative study between the proposed algorithm and seven state-of-the-art methods.

Ins.	GA	IGA	PSO	DPSO	CFPA	ICS	MA	HGA-ISA
*S*1	6779.40	6814.66	6815.14	6815.27	6813.34	6815.01	6814.65	**6816.24**
*S*2	6833.01	6882.98	6883.25	6884.79	6884.72	6883.26	6883.56	**6884.88**
*S*3	6906.43	6984.39	6986.82	6985.52	6986.14	6984.89	6984.89	**6886.26**
*S*4	6852.32	6952.54	6952.78	6953.53	6952.38	6952.82	6952.56	**6955.69**
*S*5	6799.29	6981.65	6985.59	6984.34	6981.11	6981.43	6981.63	**6981.66**
*S*6	7758.31	7923.84	7930.91	7915.18	7925.37	7902.49	**7954.47**	7928.37
*S*7	7706.98	7892.72	7885.19	7882.57	7886.59	7888.70	**7907.88**	7889.28
*S*8	7662.09	7869.23	**7884.60**	7872.44	7875.14	7876.51	7879.79	7884.42
*S*9	7532.39	7773.61	7778.36	7776.98	7776.96	7776.32	7773.80	**7778.12**
*S*10	7587.27	7975.01	7980.83	7978.79	7979.04	**7981.85**	7971.10	7978.57
*S*11	8474.15	8661.06	8589.88	8616.99	8637.19	8595.54	**8763.62**	8661.20
*S*12	8430.32	8648.65	8654.41	8645.71	8623.44	8629.43	**8739.38**	8654.47
*S*13	8444.49	8681.20	8706.77	8688.13	8675.48	8704.35	**8765.42**	8705.89
*S*14	8388.12	8730.00	8745.10	8725.22	8707.50	8742.22	**8781.06**	8742.69
*S*15	8258.97	8755.44	8760.38	8741.31	8722.98	8774.14	**8784.68**	8755.22

NB: Values in bold indicate the best obtained results for the correspondent instance.

Table 6 – Example results: average, standard deviation, and execution time for three algorithms.

Ins.	GA (Ly et al. (2015))			CSA (Binh et al. (2018))			HGA-ISA		
	Avg.	Std	Time	Avg.	Std	Time	Avg.	Std	Time
*S*1	6810.41	5.13	**193.55**	6812.31	2.33	213.55	**6814.64**	**0.892**	207.62
*S*2	6861.53	7.19	**241.12**	6879.07	1.43	**291.77**	**6884.40**	**0.27**	295.41
*S*3	6978.32	5.17	**301.33**	6984.89	3.81	**359.66**	**6984.61**	**1.01**	394.05
*S*4	6949.03	8.93	407.27	6943.09	1.79	**401.47**	**6954.59**	**1.48**	453.93
*S*5	6791.31	11.53	**574.13**	**6980.07**	**1.58**	**694.21**	6979.85	1.72	741.12
*S*6	7914.09	16.76	221.77	7884.98	11.13	**214.55**	**7939.89**	**8.08**	284.33
*S*7	7883.79	9.04	271.11	7876.38	5.01	**241.19**	**7896.95**	**4.28**	307.67
*S*8	7871.24	3.55	**300.92**	7878.63	1.52	361.86	**7884.53**	**1.12**	403.29
*S*9	7760.06	5.13	**403.04**	7777.01	1.41	**405.60**	**7777.94**	**1.32**	478.51
*S*10	7976.41	4.09	**715.23**	7971.07	1.53	**819.85**	**7978.64**	**1.15**	992.23
*S*11	8656.27	23.88	**112.41**	8597.72	18.61	151.07	**8662.05**	**17.34**	194.54
*S*12	8650.02	21.58	**218.97**	8651.22	14.19	**284.21**	**8655.39**	**13.52**	331.11
*S*13	8678.56	27.66	**267.22**	8704.35	22.46	**270.97**	**8721.65**	**19.14**	344.76
*S*14	8732.49	15.77	498.08	8745.03	13.04	**467.11**	**8747.27**	**12.55**	554.17
*S*15	8739.47	11.49	782.17	**8774.17**	**9.91**	**703.05**	8754.83	10.81	873.44

From this table, we can see that HGA-ISA consistently outperforms GA and CSA across the tested instances. It achieves the highest average solution quality, demonstrating its strong ability to reach near-optimal solutions, while also maintaining the lowest standard deviation, which reflects its robustness and stability over repeated runs. Although its execution time is higher than that of GA and CSA, this additional cost is offset by the significant improvements in solution quality and reliability. GA remains the fastest method but does so at the expense of accuracy and stability, whereas CSA achieves competitive results but without consistent superiority. These results show a clear trade-off between speed and solution quality, with HGA-ISA focusing on accuracy and robustness. Overall, its performance confirms the effectiveness of hybridization and advanced search strategies in solving the considered problem.

5.1 Assessment of the ISA algorithm and the rotation operator impact on the performance and efficiency of the proposed HGA-ISA method

To show the impact of incorporating ISA and the rotation operator (RO) into the proposed algorithm, we carried out a comparison between the standard genetic algorithm (GA) and the proposed HGA-ISA with and without the proposed rotation operator. All algorithms were initialized with the same population and parameter values to ensure an equitable evaluation. As shown in Table 7, the HGA-ISA algorithm outperformed the standard genetic algorithm in all the instances tested.

Table 7 – The average results obtained by GA, HGA-ISA with and without rotation operator (OR) from 15 independent runs for each instance.

Ins.	Size	GA without RO	GA + RO	HGA-ISA without RO	HGA-ISA + RO
$S1$	17	5200,11	5664,87	6513,16	**6816.24**
$S2$	24	5721,08	6204,41	6622,47	**6884.88**
$S3$	36	5209,26	6117,55	6598,19	**6886.26**
$S4$	57	6427,13	6642,24	6709,79	**6955.69**
$S5$	101	5991.57	6222,91	6714,11	**6981.66**
$S6$	21	6383.29	6873,89	7672,84	**7928.37**
$S7$	29	6004.08	6959,22	7702,12	**7889.28**
$S8$	41	7107,02	7397,19	7689,28	**7884.42**
$S9$	63	6451,83	6955,03	7581,63	**7778.12**
$S10$	116	6520.76	7100,11	1725,02	**7978.57**
$S11$	23	7502,17	8043,46	8479,92	**8661.20**
$S12$	32	7232,45	7527,05	8563,53	**8654.47**
$S13$	46	6687,41	7229,17	8602,07	**8705.89**
$S14$	73	7154,13	7875,79	8503,36	**8742.69**
$S15$	130	7093,33	7993,37	8413,48	**8755.22**

The performance gap becomes even more significant as the size of the problem increases. These results confirm that the incorporation of the improved simulated annealing mechanism provides an effective balance between global exploration and local exploitation, significantly improving the efficiency of the algorithm.

6 CONCLUSION

This study presents a novel hybrid genetic algorithm that incorporates Improved Simulated Annealing (HGA-ISA) to address the maximum coverage area problem in wireless sensor networks, one of the key challenges in WSN deployment. The HGA-ISA algorithm effectively combines global and local search strategies to enhance both solution quality and search efficiency. The integration of the improved Simulated Annealing technique strengthens the algorithm's intensification capability, while the introduction of a rotation operator helps maintain population diversity. Experimental evaluations on 15 benchmark instances demonstrate that the proposed method outperforms existing state-of-the-art algorithms. Future work may explore applying HGA-ISA to other optimization problems. Additionally, integrating this approach with other metaheuristics, such as Particle Swarm Optimization (PSO) or Harmony Search, could further improve its performance.

References

AKYILDIZ I. 2002. Wireless sensor networks: a survey. *Computer Networks (Elsevier) google schola*, **2**: 6–14.

BARROCA N, BORGES LM, VELEZ FJ, MONTEIRO F, GÓRSKI M & CASTRO-GOMES J. 2013. Wireless sensor networks for temperature and humidity monitoring within concrete structures. *Construction and Building materials*, **40**: 1156–1166.

BINH HTT, HANH NT, VAN QUAN L & DEY N. 2018. Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks. *Neural computing and applications*, **30**(7): 2305–2317.

BOUMEDINE N & BOUROUBI S. 2022. Protein folding in 3D lattice HP model using a combining cuckoo search with the Hill-Climbing algorithms. *Applied Soft Computing*, **119**: 108564.

BURJORJEE KM. 2013. Explaining optimization in genetic algorithms with uniform crossover. In: *Proceedings of the twelfth workshop on Foundations of genetic algorithms XII*. pp. 37–50.

CHEN C, LI Y, CAO G & ZHANG J. 2023. Research on dynamic scheduling model of plant protection UAV based on levy simulated annealing algorithm. *Sustainability*, **15**(3): 1772.

GAGE DW. 1992. Command control for many-robot systems. *Unmanned Systems Magazine*, **10**(4): 28–34.

GOGU A, NACE D, DILO A, MERATNIA N & ORTIZ JH. 2012. Review of optimization problems in wireless sensor networks. *Telecommunications Networks—Current Status and Future Trends*, pp. 153–180.

GOLDBERG DE & DEB K. 1991. A comparative analysis of selection schemes used in genetic algorithms. In: *Foundations of genetic algorithms*, vol. 1. pp. 69–93. Elsevier.

HANH NT, BINH HTT, HOAI NX & PALANISWAMI MS. 2019. An efficient genetic algorithm for maximizing area coverage in wireless sensor networks. *Information Sciences*, **488**: 58–75.

HANH NT, NAM NH & BINH HTT. 2016. Particle swarm optimization algorithms for maximizing area coverage in wireless sensor networks. In: *Proceedings of SAI Intelligent Systems Conference*. pp. 893–904. Springer.

HANH NT, NAM NH & BINH HTT. 2018. Particle swarm optimization algorithms for maximizing area coverage in wireless sensor networks. In: *Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016: Volume 2*. pp. 893–904. Springer.

IZCI D, EKINCI S & HEKIMOĞLU B. 2022. Fractional-order PID controller design for buck converter system via hybrid Lévy flight distribution and simulated annealing algorithm. *Arabian journal for science and engineering*, **47**(11): 13729–13747.

LEE SG & CHOI JH. 2011. Balance between intensification and diversification in ant colony optimization. *The Journal of the Korea Contents Association*, **11**(3): 100–107.

LOPEZ-ARDAO JC, RODRIGUEZ-RUBIO RF, SUAREZ-GONZALEZ A, RODRIGUEZ-PEREZ M & SOUSA-VIEIRA ME. 2021. Current trends on green wireless sensor networks. *Sensors*, **21**(13): 4281.

LOUCERA C, IGLESIAS A & GÁLVEZ A. 2017. Lévy Flight-Driven Simulated Annealing for B-spline Curve Fitting. In: *Nature-Inspired Algorithms and Applied Optimization*. pp. 149–169. Springer.

LY DTH, HANH NT, BINH HTT & NGHIA ND. 2015. An improved genetic algorithm for maximizing area coverage in wireless sensor networks. In: *Proceedings of the 6th International Symposium on Information and Communication Technology*. pp. 61–66.

MANTEGNA RN. 1994. Fast, accurate algorithm for numerical simulation of Levy stable stochastic processes. *Physical Review E*, **49**(5): 4677.

SYSWERDA G ET AL. 1989. Uniform crossover in genetic algorithms. In: *ICGA*, vol. 3. pp. 2–9.

YANG XS & DEB S. 2009. Cuckoo search via Lévy flights. In: *2009 World congress on nature & biologically inspired computing (NaBIC)*. pp. 210–214. Ieee.

YANG XS, DEB S & FONG S. 2014. Metaheuristic algorithms: optimal balance of intensification and diversification. *Applied Mathematics & Information Sciences*, **8**(3): 977.

YICK J, MUKHERJEE B & GHOSAL D. 2008. Wireless sensor network survey. *Computer networks*, **52**(12): 2292–2330.

YOON Y & KIM YH. 2013. An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks. *ieee transactions on cybernetics*, **43**(5): 1473–1483.

YOON Y & KIM YH. 2021. Maximizing the coverage of sensor deployments using a memetic algorithm and fast coverage estimation. *IEEE Transactions on Cybernetics*, **52**(7): 6531–6542.

ZHOU Y, FANG Y & ZHANG Y. 2008. Securing wireless sensor networks: a survey. *IEEE Communications Surveys & Tutorials*, **10**(3): 6–28.

How to cite

BOUMEDINE N & BOUROUBI S. 2026. A new hybrid genetic algorithm for maximizing area coverage in wireless sensor networks. *Pesquisa Operacional*, **46**: e299058. doi: 10.1590/0101-7438.2026.046.00299058.

Conflicts of interest

The authors declare no conflicts of interest.

Funding

The authors received no financial support for the research.

Authors contribution

NABIL BOUMEDINE: Conceptualization; Formal Analysis; Investigation; Methodology; Resources; Software; Validation; Visualization; Writing Original Draft Preparation; Writing Review & Editing. SADEK BOUROUBI: Project Administration; Data Curation; Supervision; Validation; Writing Review & Editing.

Data Availability

The data are available from the corresponding author upon reasonable request.

Editor responsible for the review

Editor-in-Chief: Annibal Parracho Sant'Anna.