G Model
ASOC-1131; No. of Pages 18

ARTICLE IN PRESS

Applied Soft Computing xxx (2011) xxx–xxx

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Multi-objective energy-efficient dense deployment in Wireless Sensor Networks using a hybrid problem-specific MOEA/D

Andreas Konstantinidis[a,b,*], Kun Yang[c]

[a] Department of Computer Science, University of Cyprus, CY-1678 Nicosia, Cyprus
[b] Department of Computer Science and Engineering, Frederick University, CY-1036, Nicosia, Cyprus
[c] School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK

ARTICLE INFO

Article history:
Received 15 September 2010
Received in revised form 9 February 2011
Accepted 22 February 2011
Available online xxx

Keywords:
Wireless sensor networks
Energy efficiency
Dense deployment
Multi-objective optimization
Evolutionary algorithms
Decomposition
Heuristics
Problem-specific knowledge

ABSTRACT

An energy-efficient Wireless Sensor Network (WSN) design often requires the decision of optimal locations (deployment) and power assignments of the sensors to be densely deployed in an area of interest. In the literature, no attempts have been made on optimizing both decision variables for maximizing the network coverage and lifetime objectives, while maintaining the connectivity constraint, at the same time. In this paper, the Dense Deployment and Power Assignment Problem (d-DPAP) in Wireless Sensor Networks (WSNs) is defined, and a Multi-objective Evolutionary Algorithm based on Decomposition (MOEA/D) hybridized with a problem-specific Generalized Subproblem-dependent Heuristic (GSH), is proposed. In our method, the d-DPAP is decomposed into a number of scalar subproblems. The subproblems are optimized in parallel, by using neighbourhood information and problem-specific knowledge. The proposed GSH probabilistically alternates between six d-DPAP specific strategies, which are designed based on various WSN concepts and according to the subproblems objective preferences. Simulation results have shown that the proposed hybrid problem-specific MOEA/D performs better than the general-purpose MOEA/D and NSGA-II in several WSN instances, providing a diverse set of high-quality near-optimal network designs to facilitate the decision making process. The behavior of the MOEA/D-GSH in the objective space is also discussed.

1. Introduction

A critical issue in energy-efficient Wireless Sensor Networks (WSNs) [1] design is the decision of locations (deployment) and transmit power levels of the sensors to be deployed in an area of interest [2], for maximizing the network coverage and lifetime objectives while maintaining connectivity. In most studies [3–6], the coverage and lifetime objectives are optimized individually, concentrating in deciding either optimal locations assuming fixed transmit power levels, or optimal transmit power levels assuming random deployment. The Deployment and Power Assignment Problem (DPAP) [7] aims at maximizing both objectives while maintaining connectivity by deciding optimal locations and transmit power levels simultaneously, in a single run.

Maximizing the coverage and lifetime objectives individually was the main focus of several studies in the past. Many practitioners, such as Meguerdichian et al. [8], have pointed out that the coverage objective is a measure of the quality of service (QoS) that is provided by a particular network design. Several researchers [3,4] have proven the NP-hardness of various deployment problems that focus in determining optimal sensor placements to cover a grid area [4] or minimize the cost or prolong the network lifetime [9]. Furthermore, in sensor network applications where the number of sensors is large and the desired area is small, the sensors that are close to the sink often burden most of the traffic load and deplete their energy supply first [5,6]. In those cases, the transmit power levels of the sensors is often increased to provide load balancing [10]. However, operating at high transmit power levels often results on a premature exhaustion of the sensors initial power supply, which is a scarce resource for the energy-constraint sensor devices [10]. Thus, several studies have focused in assigning energy efficient transmit power levels to the sensors to maximize the network lifetime under certain energy constraints [11], this is commonly known as the power/range assignment problem in WSNs [12] and it is proven NP-hard by [13]. The same problem, while maintaining connectivity [14,15] is proven NP Complete by Cheng et al. in [16]. Few studies have tackled the two problems, i.e. deployment and power assignment in WSNs, simultaneously [5,6,10]. However, those approaches optimize the lifetime and coverage objectives individually and sequentially, or by constraining one and optimizing the other. This often results in ignoring and losing "better" solutions since WSN coverage and lifetime are conflicting objectives [17]. Therefore, there is not a single solution to

* Corresponding author. Tel.: +357 22 892682; fax: +357 22 892701.
E-mail addresses: akonstan@cs.ucy.ac.cy, com.ca@fit.ac.cy (A. Konstantinidis), kunyang@essex.ac.uk (K. Yang).

1568-4946/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2011.02.031

maximize both objectives simultaneously and a decision maker [18] needs an optimal trade-off of candidate solutions.

Because so many different aspects are involved, the respective DPAP is a proper object for Multi-Objective Optimization (MOO) [19]. In a Multi-objective Optimization Problem (MOP), a candidate trade-off solution is often called non-dominated or Pareto optimal. The set of all Pareto optimal or non-dominated solutions in the search space, also called Pareto Set (PS), is often mapped to a Pareto Front (PF) in the objective space [20]. Multi-objective Evolutionary Algorithms (MOEAs) could obtain such an approximate PF in a single run [18]. This is mainly due to the fact that MOEAs accommodate different forms of operators to iteratively generate a population of solutions. In the literature, several general purpose MOEA frameworks are used for dealing with MOPs in WSNs [17,21–24] such as the Non-dominated Sorting Genetic Algorithm-II [25] (NSGA-II). However, all the aforementioned studies treat a WSN MOP as a "black box" [26], i.e. without using problem-specific knowledge, which may have undesirable effects, such as forcing the evolutionary process into unnecessary searches and destructive mating, negatively affecting their overall performance [27]. This can be considered as a main drawback of the generic MOEAs when dealing with real life problems (such as DPAP) [7].

The incorporation of problem-specific knowledge in EAs to strategically generate new solutions and their hybridization with problem-specific (local search) techniques has been proven beneficial [14,28], in the past. These methods (also known as Memetic Algorithms [29]) are powerful in Single Objective Optimization (SOO) [30]. Therefore, it is worthwhile to study how problem-specific techniques can be used to improve the performance of MOEAs. Not much effort has been given along this direction because it is not so simple as in SOO and designing problem-specific operators for a MOP as a whole is a difficult task [31]. The Multi-objective Evolutionary Algorithm based on Decomposition (MOEA/D) [32] alleviates this difficulty by decomposing the MOP into a set of single objective subproblems, which are tackled in parallel by using ideas from SOO problem-specific techniques [7].

In our previous works, we have studied DPAPs with large-scale [33] sensing areas, in which a number of sensors is *sparsely* deployed to maximize the network lifetime and coverage objectives [7,27]. The incorporation of problem-specific knowledge in MOEA/D's genetic operators [7] as well as its hybridization with problem-specific techniques [34] have shown promising results. In [35], we have introduced a constrained DPAP that aims at maximizing the same objectives while maintaining the K-connectivity (fault tolerance) constraint. In the latter, we have proposed a problem-specific Repair Heuristic (RH) to deal with the infeasible solutions and direct the search into promising feasible regions of the objective space. More recently [36], we have successfully utilized our problem specific MOEA/D-based genetic operators to the Mobile agent-based Sensor Network Routing problem.

In this paper, we deal with a constraint DPAP for small-scale dense WSN topologies, coined d-DPAP, (e.g. for security [37], military and environmental applications [38]) considering a more realistic energy model, compared to the DPAP tackled for example in [7]. The goal of the d-DPAP is to assign locations and transmit power levels to a high number of sensors to be deployed in a small sensing area and maximize the WSN coverage and lifetime while maintaining connectivity, at the same time. Note that, the higher WSN density increases the problem's difficulty and introduces new challenges to be tackled by the proposed approach, such as load balancing [10] and minimizing the coverage holes [39]. We propose, a Generalized Subproblem-dependent Heuristic (GSH) that probabilistically alternates between six low-level problem-specific strategies designed based on the objective preferences and properties of different subproblems. The six low-level improvement strategies follow different network concepts, such as managing the sensors transmit power levels, decreasing the sensing range overlaps, providing load balancing and decreasing the networks coverage holes. GSH probabilistically orchestrates the six improvement strategies and aims at improving the overall performance of MOEA/D without violating the connectivity constraints. Our major aim is to show how problem domain knowledge can be extracted and incorporated to a MOEA based on Decomposition (e.g. MOEA/D [32] used here, MOGLS [40], etc.) to improve its performance as well as demonstrate the effectiveness and efficiency of this class of MOEAs compared to the popular MOEAs based on Pareto dominance class (e.g. NSGA-II [25], MPAES [41], SPEA-II [42], SPGA-II [43]). To do that, in our simulation studies we have shown that the proposed hybrid problem-specific MOEA/D-GSH performs better than the general purpose MOEA/D as well as the state-of-the-art in MOEAs based on Pareto dominance, i.e. NSGA-II, in various WSN test instances.

2. The Dense Deployment and Power Assignment Problem (d-DPAP)

2.1. System model and assumptions

Consider a 2-D static WSN formed by: a high number of N homogeneous sensors that are densely deployed to a small rectangular sensing area A and a static sink H with unlimited energy, placed at the center of A. We assume a perfect medium access control, such as SMAC [44], which ensures that there are no collisions at any sensor during data communication and we adopt the simple but relevant path loss communication model as in [6]. In this model, the transmit power level that should be assigned to a sensor i to reach a sensor j is $P_i = \beta \times d_{ij}^{\alpha}$, where $\alpha \in [2, 6]$ is the path loss exponent and $\beta = 1$ is the transmission quality parameter. The energy loss due to channel transmission is d_{ij}^{α}, where d_{ij} is the Euclidean distance between sensors i and j. The communication range of each sensor i is $R_c^i = d_{ij}$, s.t. $R_c^i \leq R_{\max}$, where $R_{\max}$ is the maximum communication range that is determined by the maximum transmit power level that a sensor can be assigned, denoted as $P_{\max}$. The transmit power level P_i and the coordinates (x_i, y_i) are the d-DPAP's decision variables that are considered fixed for the whole network lifetime, for sensor $i = 1, \ldots, N$. The residual energy of sensor i, at time t, is calculated as follows:

$$E_i(t) = E_i(t-1) - [E_{tx}^i(t) + E_{rx}^i(t) + E_s], \quad (1)$$

where $E_{tx}^i(t) = k \times (r_i(t) + 1) \times (P_i \times amp + E_{ct})$, $E_{rx}^i(t) = k \times r_i(t) \times E_{ct}$ is the amount of energy consumed by sensor i for transmission and reception, respectively, E_s is the amount of energy consumed for sensing and processing k, which is the amount of data sensed and collected by a sensor with a fixed sensing range R_s, $r_i(t) + 1$ is the total traffic load that sensor i forwards towards H at t, $r_i(t)$ is the traffic load that i receives and relays and "+1" is the data packet generated by i to forward its own data information, amp is the power amplifier's energy consumption and E_{ct} is the energy consumption due to the transmitter and receiver electronics.

Furthermore, it is assumed that A is divided into G uniform consecutive grids to make the coverage problem more computationally manageable [3]. The size of the grids is several times smaller than $\pi \times R_s$ for a more accurate approximation within the sensing disk. A sensing model based on the definite range law approximation is considered [45],

$$g(x', y') = \begin{cases} 1 & \text{if} \quad \exists j \in \{1, \ldots, N\},\ d_{(x_j, y_j),(x', y')} \leq R_s, \\ 0 & \text{otherwise,} \end{cases} \quad (2)$$

is the monitoring status of a grid centered at (x', y') with 1 indicating that the grid is covered and 0 otherwise.

Please cite this article in press as: A. Konstantinidis, K. Yang, Multi-objective energy-efficient dense deployment in Wireless Sensor Networks using a hybrid problem-specific MOEA/D, Appl. Soft Comput. J. (2011), doi:10.1016/j.asoc.2011.02.031

Finally, the connectivity status of a sensor j is denoted as,

$$c_j = \begin{cases} 1 & \text{if } j \text{ is connected,} \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where sensor j is considered connected if it directly or through nearby sensors communicates with H.

2.2. d-DPAP formulation

The d-DPAP in WSNs can be formulated as a constrained MOP, **Given**:

- A: small 2-D plane of area size $x \times y$.
- N: high number of sensors to be deployed in A.
- E: initial power supply, the same for all sensors.
- R_s: sensing range, the same for all sensors.
- P_{max}: maximum transmission power level, the same for all sensors.

Decision variables of solution *X*:

- (x_j, y_j): the location of sensor j.
- P_j: the transmission power level of sensor j.

Objectives: Maximize coverage $Cv(X)$ and lifetime $L(X)$, **subject to** connectivity $Cn(X) = 1$.

The network coverage $Cv(X)$ is defined as the percentage of the covered grids over the total grids of A and is evaluated as follows:

$$Cv(X) = \frac{\left[\sum_{\text{all}(x',y')} g(x', y') \right]}{G}, \tag{4}$$

where G is the total grids of A and $g(x', y')$ is calculated using Eq. (2).

The network lifetime $L(X)$ is defined as the duration from the deployment of the network to the cycle t in which a sensor j depletes its energy supply E and is evaluated as as in Algorithm 1.

Algorithm 1. Lifetime evaluation

Step 0: Set $t := 1$; $E_j(0) := E, \forall j \in \{1, \ldots, N\}$;
Step 1: **For** all sensors j at each time interval t**do**
Step 1.1: Update $E_j(t)$ according to Eq. (1);
Step 1.2: Assign each incoming link of sensor j a weight equal to $E_j(t)$;
Step 1.3: Calculate the shortest path from j to H;
Step 2: **If** $\exists j \in \{1, \ldots, N\}$ such that $E_j(t) = 0$**then** stop and set:

$L(X) := t$;

Else $t = t + 1$, go to step 1;

The same algorithm can be easily modified to consider different energy models in Step 1.1 (e.g. [27]) and routing algorithms in Step 1.3 (e.g. geographical-based [15] routing algorithms).

The percentage of connected sensors in X can be measured as follows:

$$Cn(X) = |CS|/N, \tag{5}$$

where $CS = \{j \mid c_j = 1\}$, $Cn(X) = 1$ when all sensors are connected and c_j is calculated using Eq. (3).

In Multi-Objective Optimization (MOO) [20] we need the following definitions. We assume that we have n objectives $f_1, \ldots, f_n$ to maximize.

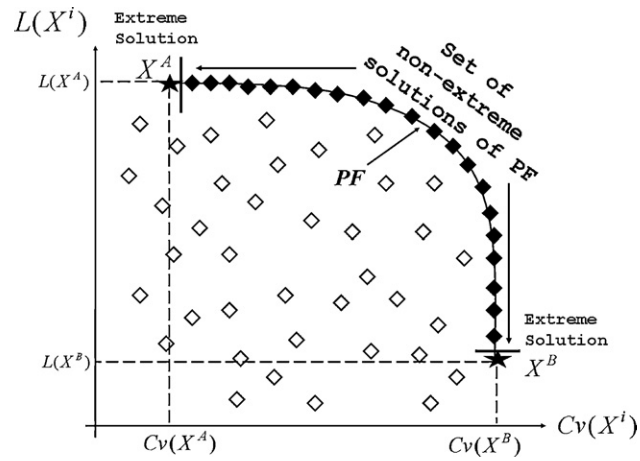

Fig. 1. The PF of d-DPAP, having X^A and X^B as the extreme solutions.

Definition 1. Pareto dominance

Suppose x and y are two decision variables, x is said to *dominate* y, denoted by $x \succ y$, if and only if $f_i(x) \geq f_i(y)$ for every $i \in \{1, 2, \ldots, n\}$ and $f_j(x) > f_j(y)$ for at least one index $j \in \{1, 2, \ldots, n\}$.

Definition 2. Pareto optimality $x^* \in \Omega$ is said to be *Pareto-optimal (or nondominated)* if there is no another $x \in \Omega$ so that x dominates x^*. The set of all Pareto-optimal solutions in the decision space is called the *Pareto-optimal set (PS)*. The image of the PS in the objective space is is called the *Pareto-optimal Front (PF)*.

Fig. 1 illustrates the PF in d-DPAP with $n = 2$ (i.e. lifetime and coverage). The Pareto optimal solutions in the PF (marked as solid diamonds) provide better lifetime and/or coverage than any other solution in the objective space. The remaining solutions (marked as open diamonds) are all dominated by at least one solution of the PF. Solutions X^A and X^B (marked as solid stars) are often called the extreme points of the PF [20], since they provide the highest lifetime and coverage, respectively, than any other solution in the objective space.

2.3. d-DPAP solution representation and ordering

In d-DPAP, a candidate solution X consists of N items. Its j th item has two parts, (x_j, y_j) and P_j, which represent the location and the transmit power level of sensor j, respectively. The items of a solution X are ordered as follows: the sensor locations in X are sorted based on their distance to H, where 1 is the closest and N is the farthest sensor location with respect to H, respectively. This results in having the locations of the sensors that are densely deployed around H at the beginning of each solution and the locations of the sensors that are spread away at the end, and is commonly known as the dense-to-spread ordering [7]. Then each sensor j is assigned a transmit power level P_j proportional to $R_c^j \leq R_{max}$, such that it reaches its closest neighbour sensor, e.g. k, where $k < j$.

3. The Multi-objective Evolutionary Algorithm based on Decomposition (MOEA/D)

3.1. Decomposition and d-DPAP analysis

MOEA/D initially decomposes the MOP into m single-objective subproblems. In [32] several decompositional techniques are studied for this purpose, including the Weighted Sum approach used here. In the multi-objective d-DPAP that considers two objectives, i.e. lifetime $L(X)$ and coverage $Cv(X)$, the i th single-objective opti-

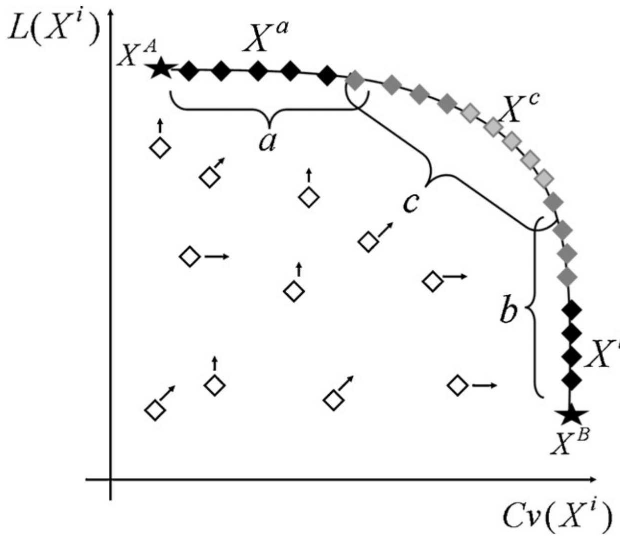

Fig. 2. Classifying the Pareto-optimal network designs of the d-DPAP. This will benefit the design of appropriate single-objective heuristics to direct the solutions towards the PF.

mization subproblem is defined as:

$$\max \quad g^i(X, \lambda^i) = \lambda^i L(X) + (1 - \lambda^i)Cv(X),$$

where λ^i is the weight coefficient of subproblem $i = 1, \ldots, m$. For the remainder of this paper, we consider a uniform spread of the weights λ^i, which remain fixed for each i for the whole evolution [46] and are determined as follows:

$$\lambda^i = 1 - \left(\frac{i}{m}\right),$$

for $i = 2, \ldots, m$ and $\lambda^1 = 1$.

In the aggregation approaches, such as MOEA/D [32], and MOGLS [40], the λ^i coefficient is mainly used for decomposing a MOP into single-objective subproblems by adding different weights to the objectives. In this paper, we have also given a problem-specific meaning to this parameter as follows: the λ^i weight coefficient of a subproblem i, can also be used to predict the objective preference of a particular design X^i and therefore its position in the objective space. For example, the *extreme solutions* X^A and X^B in Fig. 2 focus at optimizing one objective each. The extreme solution X^A aims at maximizing the lifetime objective ignoring coverage and the extreme solution X^B focuses at maximizing the coverage objective ignoring lifetime. The goal of d-DPAP, however, is to obtain a trade-off set of solutions between the extreme solutions, providing the interested users with a diverse set of network design choices (e.g. X^a, X^b, and X^c in Fig. 2) to facilitate decision making. However, the procedure of designing non-extreme topologies is complicated, since there is not a single objective method which can design all of them in a single run. Therefore, in order to effectively deal with the d-DPAP, one should design single-objective heuristics (by extracting knowledge from the problem domain and the objectives properties) to be strategically controlled by MOEA/D and optimize different areas of the objective space (e.g. *a*, *b*, and *c* in Fig. 2) accordingly. In d-DPAP, improvement heuristics considering the following problem-specific properties can be shown beneficial:

(a) decrease the sensors' transmit power levels to reduce their energy consumption and favour the lifetime objective without affecting the connectivity constraints,
(b) provide load balancing [10], often defined as the technique that spreads the traffic load between two or more sensors, so that no individual sensor becomes overwhelmed by too much traffic. This is often achieved by utilizing longer hops to increase the node-degree of each sensor (i.e. number of one-hop neighbours) and consequently balance the traffic load and/or avoid heavy loaded critical sensors,
(c) decrease the sensing range overlaps by spreading the sensors in the area and increasing their transmit power level to favour the coverage objective and maintain the connectivity,
(d) tackle the so-called coverage-hole problem [39], i.e. a small subarea surrounded by connected sensors remains uncovered, to favour the coverage objective.

Note that, this beneficial procedure cannot be utilized by any non-decompositional MOEA framework.

3.2. MOEA/D: an overview

A general MOEA/D approach usually proceeds as in Algorithm 2.

Algorithm 2. The MOEA/D general framework

Input: • network parameters (A, N, E, R_s, $P_{\max}$);
• m: population size and number of subproblems;
• T: neighbourhood size;
• uniform spread of weight vectors $(\lambda^1, 1 - \lambda^1), \ldots, (\lambda^m, 1 - \lambda^m)$;
• the maximum number of generations, $gen_{\max}$;
Output: the external population, *EP*.
Step 0 – Setup: Set $EP := \varnothing$; $gen := 0$; $IP_{gen} := \varnothing$;
Step 1 – Initialization: Uniformly randomly generate an initial internal population $IP_0 = \{X^1, \cdots, X^m\}$;
Step 2: **For** $i = 1, \ldots m$ **do**
Step 2.1 – Genetic operators: Generate a new solution Y using the genetic operators.
Step 2.2 – Repair/improvement heuristic: Apply a scalar repair/improvement heuristic on Y to produce Z.
Step 2.3 - Update populations: Use Z to update IP_{gen}, EP and the T closest neighbour solutions of Z.
Step 3 – Stopping criterion: **If** stopping criterion is satisfied, i.e. $gen = gen_{\max}$, **then** stop and output *EP*, **otherwise** $gen = gen + 1$, go to Step 2.

The internal population IP_{gen} of size m keeps the best solution found so far for each subproblem. The initial solutions of IP_0 are generated either randomly [7], which is also the case in this paper, or by a problem-specific heuristic [35]. Solution Y is generated by using a selection operator (e.g. the M-tournament selection [7]) to choose two parent solutions from the IP_{gen}, e.g. Pr_1, Pr_2, a crossover operator (e.g. the Adaptive Crossover [7]) to produce a new solution from Pr_1, Pr_2 and a mutation operator (e.g. the Adaptive Mutation [7]) to modify the new solution Y. Solution Z is produced by using a repair method(e.g. the Repair Heuristic [35]) and an improvement heuristic on Y. The T closest neighbour solutions of Z are the solutions of the T closest subproblems of i in terms of their weights $\{\lambda^1, \cdots, \lambda^m\}$. This is commonly known as the T neighbourhood of subproblem i. The external population *EP* stores all the non-dominated solutions found so far during the search.

In this paper, the focus is on the hybridization of the MOEA/D with an improvement heuristic at Step 2.2.

4. The Generalized Subproblem-dependent Heuristic

In this section, we initially design six low-level improvement strategies, each having different properties and providing different treatment to the solutions of the d-DPAP's search space. Then we classify the six improvement strategies based on the way that

Please cite this article in press as: A. Konstantinidis, K. Yang, Multi-objective energy-efficient dense deployment in Wireless Sensor Networks using a hybrid problem-specific MOEA/D, Appl. Soft Comput. J. (2011), doi:10.1016/j.asoc.2011.02.031

they treat a solution (i.e. improve a solution towards the direction of one objective with/without affecting the other). Finally, we study how they can be combined to efficiently and effectively improve a solution X of a subproblem i and we design the Generalized Subproblem-dependent Heuristic that is composed of the six low-level improvement strategies.

4.1. Improve-coverage/affecting Lifetime ($ImpCv_1(X)$): re-locating sensors for minimizing sensing range overlaps

$ImpCv_1(X)$ mainly favours the solutions of the subproblems that require high network coverage (i.e. some X^b and X^c in Fig. 2), as follows:

- The sensing range overlaps between the sensors are minimized by increasing the distance between them.
- The distance that the sensors are shifted is limited and increases as the distance between the sensors and H increases.
- The sensing range overlaps between the sensors and the area boundaries are minimized.

In $ImpCv_1(X)$, a sensor k at location (x_k, y_k) is shifted backwards from its one-hop neighbour j a distance *shift*, to decrease the sensing range overlap between them. The sensing range overlap between two sensors k, j denoted as $A_o(k, j)$, is equal to:

$$A_o(k, j) = R_s^2(q - sin(q)),$$

where $q = 2 \times acos(d_{kj}/2R_s)$. Therefore, by increasing d_{kj} the $A_o(k, j)$ between k and j decreases. Note that for $d_{kj} = 2 \times R_s$, $A_o(k, j) = 0$.

However, $ImpCv_1(X)$ may force the solutions of each subproblem i with low λ^i to converge towards a single solution, i.e. X^B. This is undesirable, since our primary aim is to obtain the trade-off between the objectives (i.e. the solutions between the extreme solutions X^A, X^B in Fig. 2). Hence, the new location (x'_k, y'_k) should be calculated so that the sensing range overlap between sensors k and j is decreased, giving also an adequate network lifetime. Let $\overline{r_k}$, $\overline{r_j}$ be the average traffic load of sensors k and j, respectively, and $a = 2$.

Firstly, the required distance d'_{kj} so that the sensors k and j deplete their energy supply approximately at the same time is calculated as follows:

$$d'_{kj} = \sqrt{\frac{P_j \times amp \times \overline{r_j}}{amp \times \overline{r_k}}}, \tag{6}$$

and therefore, the shift that sensor k should move backwards from j is:

$$\text{shift} = d'_{kj} - d_{kj}. \tag{7}$$

Consequently, the new location of sensor k is:

$$(x'_k, y'_k) = \frac{(x_k, y_k) \times d'_{k,j} - \text{shift} \times (x_j, y_j)}{(d'_{k,j} - \text{shift})}. \tag{8}$$

The new transmit power level of sensor k is:

$$P'_k = \left(\sqrt{(x'_k - x_j)^2 + (y'_k - y_j)^2}\right)^{\alpha}, \tag{9}$$

where $\alpha = 2$. The final location and power assignment of sensor k should satisfy $P_j \times amp \times \overline{r_j} = P'_k \times amp \times \overline{r_i}$ and $d'_{kj} > d_{kj}$ so that $A_o(k, j)' < A_o(k, j)$ and $(x'_k, y'_k) \in A$. The first part of $ImpCv(X)$ is illustrated in Fig. 3.

Thereinafter, $ImpCv_1(X)$ minimizes the sensing range overlaps between the sensors and the area boundaries. Assuming that the area is a rectangle, there are three different cases where a sensor violates the boundaries:

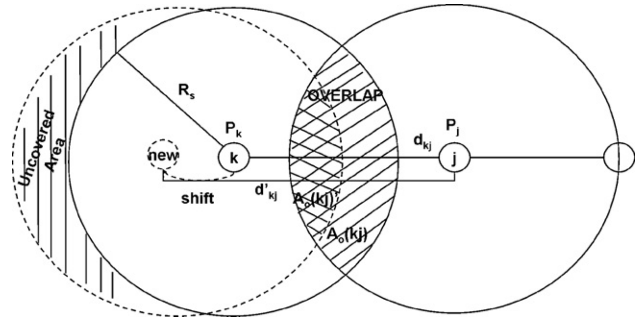

Fig. 3. An example of the first part of $ImpCv_1(X)$ heuristic.

- **Case #1**: Violation on x-axis: (a) left or (b) right bound.
- **Case #2**: Violation on y-axis: (a) lower or (b) upper bound.
- **Case #3**: Violation on both axes: (a) lower/left, (b) lower/right, (c) upper/left, and (d) upper/right.

If a sensor k at location (x_k, y_k) violates any of the Cases #1, #2, #3 (illustrated in Fig. 4) then, is relocated in area $A = [0, x] \times [0, y$, as follows:

$$(x'_k, y'_k) = \begin{cases} (x', y_k) & \text{if Case \#1a, where } x' = R_s \\ (x', y_k) & \text{if Case \#1b, where } x' = x - R_s \\ (x_k, y') & \text{if Case \#2a, where } y' = R_s \\ (x_k, y') & \text{if Case \#2b, where } y' = y - R_s \\ (x', y') & \text{if Case \#3a, so that } d_{(0,0),(x',y')} = \sqrt{(0 - x')^2 + (0 - y')^2} \\ (x', y') & \text{if Case \#3b, so that } d_{(x,0),(x',y')} = \sqrt{(x - x')^2 + (0 - y')^2} \\ (x', y') & \text{if Case \#3c, so that } d_{(0,y),(x',y')} = \sqrt{(0 - x')^2 + (y - y')^2} \\ (x', y') & \text{if Case \#3d, so that } d_{(x,y),(x',y')} = \sqrt{(x - x')^2 + (y - y')^2} \end{cases} \tag{10}$$

Finally, sensor k is assigned a P_k, so that it reaches its closest neighbour. $ImpCv_1(X)$ proceeds as in Algorithm 3,

Algorithm 3. The $ImpCv_1(X)$ for a subproblem i

Input: Solution X;
Output: Improved solution Y;
Step 1: Order solution X by using the dense-to-spread ordering [7];
Step 2: Assign transmit power levels to solution X as in Section 2.3;
Step 3: Apply Algorithm 1 on solution X;
For $k = 1$ to N do
Step 4: **If** (x_j, y_j) is (x_k, y_k)'s next-hop neighbour location and $(x_j, y_j) \neq (x_H, y_H)$ **then**
  Step 4.1: Calculate the distance d'_{kj} using Eq. (6);
  Step 4.2: **If** $d'_{kj} > d_{kj}$ **then** calculate the *shift* using Eq. (7), **otherwise** stop;
  Step 4.3: Use Eq. (8) to find the new location $(x'_k, y'_k) \in A$ and calculate P'_k with Eq. (9). Replace $(x_k, y_k) \in X$ with (x'_k, y'_k) and set $P_k = P'_k$;
Step 5: **If** sensor k violates any of the Cases #1, #2, #3 **then**
  Step 5.1: Calculate a new location (x'_k, y'_k) using Eq. (10). Replace (x_k, y_k) with (x'_k, y'_k);
  Step 5.2: Set P_k, so that sensor k at (x_k, y_k) reaches its closest one-hop neighbour;
Step 6: Output $Y = X$;

In Step 1, the solution is ordered (i) to manipulate the current locations and power levels more easily and (ii) to start modifying the locations from inside of the area A to outside. In Step 2, the

Please cite this article in press as: A. Konstantinidis, K. Yang, Multi-objective energy-efficient dense deployment in Wireless Sensor Networks using a hybrid problem-specific MOEA/D, Appl. Soft Comput. J. (2011), doi:10.1016/j.asoc.2011.02.031

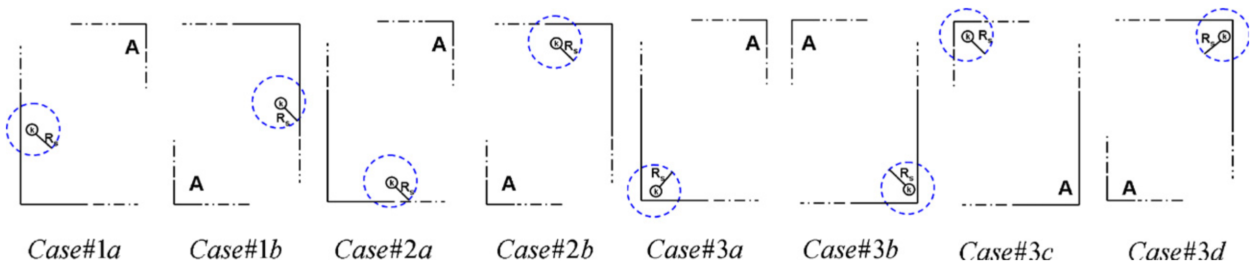

Fig. 4. The cases of boundary violations in the second part of $ImpCv_1(X)$.

power assignment of solution X is calculated. In Step 3, the algorithm that evaluates the lifetime is applied to solution X to get some network information about the current network topology (e.g. $r_j(t)$ of sensor j). In Step 4, when sensor k has many one-hop neighbours, then (x_j, y_j) is the one with the smallest d_{kj} and consequently the largest $A_o(k,j)$. The locations (x_k, y_k) of the sensors which are directly connected to H are not modified. In Steps 4 and 5, some sensor locations and transmit power levels are updated, accordingly.

Remark. This method focuses at improving the coverage objective having a negative impact on the lifetime objective, since the sensors' transmit power levels increase resulting to higher energy consumptions. Therefore, it mainly favours the solutions of area c and might favour some solutions of area b (i.e. X^b, X^c).

4.2. Improve-lifetime/affecting coverage ($ImpL_1(X)$): re-locating sensors for decreasing transmit power level

The goal of $ImpL_1(X)$ is to densely deploy the sensors around H and decrease their transmit power levels while they get closer to H, as follows:

A sensor j at location (x_j, y_j) is moving towards its one-hop neighbour h at location (x_h, y_h) a distance *shift*, which depends on:

- the energy consumption of sensor j at time t, i.e. $(r_j(t)+1) \times P_j \times amp$,
- the energy consumption of sensor k at location (x_k, y_k) at time t, which considers sensor j as its one-hop neighbour, i.e. $(r_k(t)+1) \times P_k \times amp$,

so that sensors j and k deplete their energy supply approximately at the same time, the energy consumption is given by Eq. (1).

Let $\overline{r_j}$ and $\overline{r_k}$ be the average traffic load of sensors j and k, respectively, during the network lifetime and $\alpha = 2$. Firstly, the required distance d'_{jh} between sensors j and h, so that sensors j and k deplete their energy supply approximately at the same time and therefore little residual energy is wasted at the end of a WSN lifetime, is calculated as follows:

$$d'_{jh} = \sqrt{\frac{d_{kj}^2 \times amp \times \overline{r_k}}{amp \times \overline{r_j}}}. \quad (11)$$

Consequently, the shift that sensor j should move towards h is:

$$\text{shift} = d_{jh} - d'_{jh} \quad (12)$$

Thereinafter, the new location of sensor j is:

$$(x'_j, y'_j) = (x_j, y_j) + \frac{\text{shift} \times [(x_h, y_h) - (x_j, y_j)]}{d_{jh}}. \quad (13)$$

The updated transmit power levels of sensor j and k are:

$$P'_j = \left(\sqrt{(x'_j - x_h)^2 + (y'_j - y_h)^2}\right)^{\alpha}, \quad P'_k = \left(\sqrt{(x'_j - x_k)^2 + (y'_j - y_k)^2}\right)^{\alpha}, \quad (14)$$

the final location and power assignment of sensor j should satisfy $d'_{jh} < d_{jh}$ so that the energy consumption of sensor j is decreased and shift $< d_{jH}$. $ImpL(X)$ proceeds as in Algorithm 4.

Algorithm 4. The $ImpL_1(X)$ for a subproblem i

Input: Solution X;
Output: Improved solution Y;
Step 1: Order solution X by using the dense-to-spread ordering [7];
Step 2: Assign transmit power levels to solution X as in Section 2.3.
Step 3: Apply Algorithm 1 on solution X;
Step 4: **For** $j = 1$ to N **do**
 Step 4.1: Calculate the distance d'_{jh} using Eq. (11);
 Step 4.2: **If** $d'_{jh} < d_{jh}$ **then** calculate the *shift* of j towards h using Eq. (12), **otherwise** stop;
 Step 4.3: **If** shift $< d_{jh}$ **then** calculate the new location (x'_j, y'_j) using Eq. (13),
 the transmit power levels P'_j and P'_k using Eq. (14) and update X;
Step 5: Output $Y = X$;

In Step 1, the solution is ordered (i) to manipulate the current locations and power levels more easily and (ii) to start modifying the locations from inside of the area A to outside. In Step 2, the power assignment of solution X is calculated. In Step 3, the algorithm that evaluates the lifetime is applied to solution X to get some network information about the current network topology (e.g. $r_j(t)$ of sensor j). In Step 4, some sensor locations and transmit power levels in X are updated, accordingly. The procedure and some special cases of $ImpL_1(X)$ are illustrated in Fig. 5.

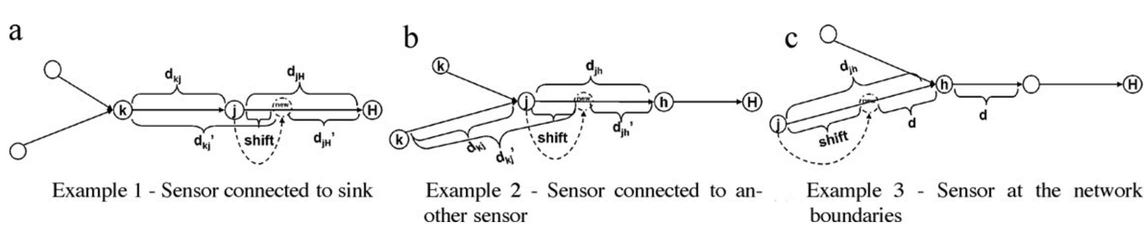

Fig. 5. Examples of the $ImpL_1(X)$ strategy.

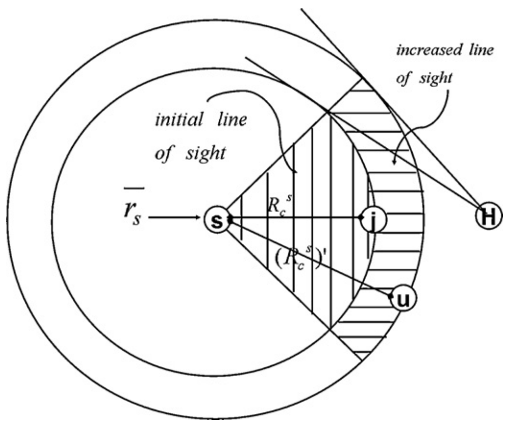

Fig. 6. The main concept of $Imp_1(X)$.

Remark. This method focuses at improving the lifetime objective having a negative impact on the coverage objective, since the sensors are shifted towards the center of the area. Therefore, this method mainly favours the solutions of area a and might favour some solutions of area b (i.e. X^a, X^b).

4.3. Improve-lifetime/without affecting coverage $Imp_1(X)$: increasing transmit power level for better load balancing

In the previous improvement heuristic, the concept that many short hops are more efficient than a single long hop is followed, connecting each sensor to its closest neighbour sensor, in terms of the Euclidean distance between each other [5,47]. In d-DPAP however, transmitting packets via multiple short hops may consume more energy [10,6] and may not be the optimal transmission structure for all solutions in the objective space.

For example, let sensor i be the source, sensor j the destination and sensor k an intermediate sensor. The total energy consumption of a long hop d_{ij} is equal to:

$$k \times amp \times (d_{ik} + d_{kj})^{\alpha} + 2kE_{ct} + E_s,$$

and the total energy consumption of two short hops is equal to:

$$k \times amp \times (d_{ik}^{\alpha} + (2d_{kj})^{\alpha}) + 4kE_{ct} + 2E_s.$$

Hence, if the following inequality is satisfied, the single long hop is more energy efficient than the two short hops:

$$(d_{ik} + d_{kj})^{\alpha} - (d_{ik}^{\alpha} + (2d_{kj})^{\alpha}) < \frac{2kE_{ct} + E_s}{k \times amp}.$$

On this account, the lifetime of the sensors that are directly connected to H and burden most of the network traffic load (i.e. critical sensors) becomes a vital issue. $Imp_1(X)$ improves the lifetime objective without affecting the coverage by increasing the network load balancing and the sensors' individual node-degree. This is achieved by increasing the sensors' transmit power level and consequently their line of sight. A sensor s at location $(x_s, y_s) \in X$ increases its transmit power level P_s (Fig. 6) to,

- avoid a critical sensor j at location (x_j, y_j), which is a positive advance neighbour of s (i.e. sensor j is closer to H than sensor s in terms of Euclidean distance) by directly communicating with H,
- increase the number (node-degree) of its positive advance neighbours u (i.e. sensors u are closer to H than sensor s) falling within its line of sight and therefore balance its individual traffic load more efficiently.

Sensor s, however, needs to ensure that the increase of its P_s will not cause an earlier depletion of its initial energy supply than sensor j. Let $\overline{r_j}$ and $\overline{r_s}$ be the average traffic load of sensors j and s, respectively, during the network lifetime. Thus, the maximum transmit power level, P_s', that can be assigned to sensor s is:

$$P_s' = \frac{P_j(\overline{r_j} + 1) + (1/amp) \times (\overline{r_j} - \overline{r_s})}{\overline{r_s} + 1}. \quad (15)$$

Therefore, sensor s should be assigned a transmit power level:

$$P_s'' = (R_c^s)^{\alpha}, \quad \text{where} \quad R_c^s = \begin{cases} d_{sH} & \text{if}(d_{sH})^{\alpha} \le P_s'; \\ d_{su} & \text{otherwise}; \end{cases} \quad (16)$$

where u is the farthest positive advance neighbour of s and $(d_{su})^{\alpha} \le P_s'$. $Imp_1(X)$ is outlined in Algorithm 5.

Algorithm 5. The $Imp_1(X)$ for a subproblem i

Input: A solution X;
Output: A solution Y;
Step 1: Order solution X by using the dense-to-spread ordering [7];
Step 2: Assign transmit power levels to solution X as in Section 2.3.
Step 3: Apply Algorithm 1 on solution X;
Step 4: **For** each location $(x_s, y_s) \in X$, where $s = 1, \ldots, N$ **do**
Step 4.1: Calculate P_s' with Eq. (15);
Step 4.2: Calculate P_s'' with Eq. (16);
Step 4.3: Replace $P_s \in X$ with P_s'';
Step 5: Output $Y = X$;

In Step 1, the solution is ordered to start modifying the transmit power levels of the sensors, which are close to the critical sensors, at the center of area A. In Step 2, the power assignment of solution X is calculated. In Step 3, the algorithm that evaluates the lifetime is applied to solution X to get some network information about the current network topology (e.g. $r_j(t)$ of sensor j). In Step 4, some sensor transmit power levels are updated accordingly.

Remark. $Imp_1(X)$ focuses at improving the lifetime without affecting the coverage objective, overall.

Another way of increasing the node-degree of sensor s, which may increase the network load balancing, is by adding more sensors u within its line of sight. However, in the d-DPAP the number of sensors is fixed and the node-degree of s can be only increased by relocating an existing sensor u within its line of sight. In that case, one should tackle the following issues: *"Which sensoru to relocate?"* and *"Where to relocate sensoru ?"*. The following two improvement strategies increase the network load balancing, favouring the network lifetime, by tackling the aforementioned issues.

4.4. Improve-lifetime/without affecting coverage ($Imp_2(X)$): re-locating sensors for better load balancing

$Imp_2(X)$ facilitates the network lifetime of a solution X without affecting the coverage by selecting and relocating a sensor s within another sensor's line of sight. Initially, $Imp_2(X)$ identifies a sensor $s \in \{1, \ldots, N\}$ that satisfies the following conditions:

- Its relocation does not violate the connectivity constraint (defined in Section 2.2) or partitions the network. Thus, sensor s is relocated iff,

$$\nexists u \in \{1, \cdots, N\} | s \in n_u, |n_u| = 1,$$

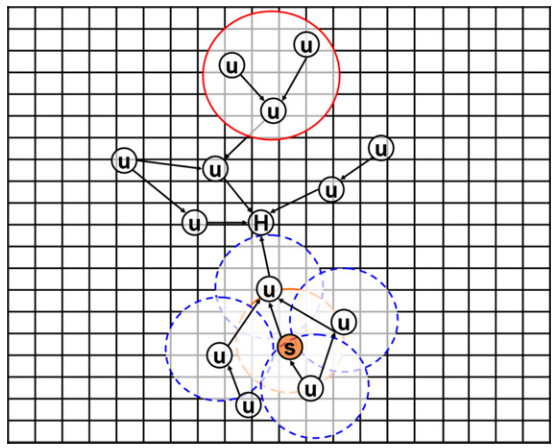

Fig. 7. Example of a network design X with sensors u not satisfying conditions $C1$, $C2$, $C3$ and the shadowed sensor s being eligible for relocation. The dashed circles indicate the sensors' R_s and the solid circle indicates an example of a possible network partition.

where n_u is the set of the positive advance neighbours of sensor u.

- Its relocation does not uncover any previously covered area of the network. Thus, a sensor s is relocated iff,

$$\nexists g(x', y')^R = 1 | R = 1, d_{(x_s, y_s),(x', y')} \le R_s,$$

where $g(x', y')$ is the monitoring status of the grid centred at $(x', y') \in A$ and R is the number of sensors covering that grid.

- It is not directly connected to H,

$$d_{sH} > R_{\max}.$$

The locations of all sensors $s \in \{1, 2, \ldots, N\}$ that satisfy conditions $C1$, $C2$ and $C3$ are added in the set S. The remaining sensors u, which do not satisfy the conditions (as illustrated in Fig. 7) are considered ineligible for relocation and remain in their current position. Thereinafter, a sensor $s \in S$ is relocated as follows (Fig. 8).

Algorithm 6. The $Imp_2(X)$ for a subproblem i

Input: A solution X;
Output: A solution Y;
Step 1: Assign transmit power levels to solution X as in Section 2.3.
Step 2: Apply Algorithm 1 on solution X;
Step 3: Generate a set $S \subset \{1, \cdots, N\}$ with the locations of all sensors s that satisfy $C1$, $C2$, $C3$;
Step 4: Identify the location $(x_j, y_j) \in X$ of the critical sensor j, the location $(x_u, y_u) \in X$ of sensor u and uniformly randomly select a location $(x_s, y_s) \in S$ of sensor s;
Step 5: **If** $R_c^u + R_c^j > d_{u,j}$ and $R_c^u - R_c^j < d_{u,j}$ **then**
Step 5.1: Calculate (x'_s, y'_s) with Eq. (17);
Step 5.2: Replace $(x_s, y_s) \in X$ with (x'_s, y'_s) and set $P_s = d_{sh}^{\alpha}$;
Else
Step 5.3: Uniformly randomly generate (x'_s, y'_s) so that $d_{js} \le R_c^j$;
Step 5.4: Replace $(x_s, y_s) \in X$ with (x'_s, y'_s) and set $P_s = d_{sv}^{\alpha}$;
EndIf
Step 6: Output $Y = X$;

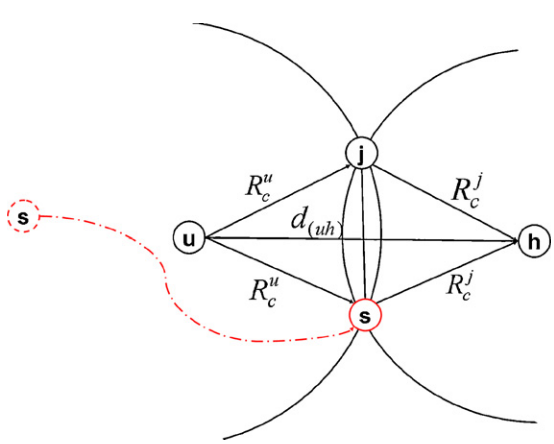

Fig. 8. The main concept of $Imp_2(X)$.

Let j be a critical sensor (i.e. the sensor that currently depletes its energy supply first), u the sensor with the highest $\overline{r_u}$ from all sensors that include j as their positive advance neighbour and h the farthest positive advance neighbour of j. Note that when j is directly connected to H then $h = H$. The new location (x'_s, y'_s) of s should meet the following:

$$\begin{aligned} (x'_s - x_u)^2 + (y'_s - y_u)^2 &= (R_c^u)^{\alpha}, \\ (x'_s - x_h)^2 + (y'_s - y_h)^2 &= (R_c^j)^{\alpha}, \end{aligned} \tag{17}$$

where $\alpha = 2$. The solution of Eq. (17) is two locations, one is the same as (x_j, y_j) and the other is (x'_s, y'_s). Set $P_s = d_{sh}^{\alpha}$. $Imp_2(X)$ is outlined in Algorithm 6.

In Step 1, the power assignment of solution X is calculated. In Step 2, the algorithm that evaluates the lifetime is applied to solution X to get some network information about the current network topology (e.g. $r_j(t)$ of sensor j). In Step 3, the set S with the locations of all sensors s that satisfy the conditions $C1$, $C2$ and $C3$ is generated. In Step 4, the locations of the critical sensor j, sensor u and sensor s are identified. In Step 5, a new location (x'_s, y'_s) is calculated and the corresponding transmit power is updated. However, a new location (x'_s, y'_s) does not exist if:

- $]R_c^u + R_c^j < d_{u,h}$ //circles are separated,
- $]R_c^u - R_c^j > d_{u,h}$ //circles are combined within each other

and only one pair exists if:

- $]R_c^u + R_c^j = d_{u,j}$ //circles are tangent.

In that case, a new location (x'_s, y'_s) is uniformly randomly generated within an R_c^j distance from $(x_j, y_j) \in X$ and a $P_s = d_{sv}^{\alpha}$ is set, such as sensor s is directly connected to its closest neighbour v, where $d_{sv} \le R_{\max}$.

Remark. $Imp_2(X)$ focuses at improving the lifetime without affecting the coverage objective, overall.

4.5. Improve-lifetime/affecting coverage ($ImpL_2(X)$): re-locating sensors for better load balancing

$ImpL_2(X)$ facilitates the network lifetime of a solution X and has a negative impact on coverage by selecting and relocating a sensor s within another sensor's line of sight. $ImpL_2(X)$ aims at (1) increasing the load balancing by increasing the number of sensors around a critical sensor j and (2) decreasing the overall traffic load forwarded towards j.

Initially, $ImpL_2(X)$ finds a sensor $s \in \{1, \ldots, N\}$ that satisfies the following condition:

- C4: There exists a path p_s, which is the longest (in terms of number of sensors) among all paths towards H. Sensor s is the source of the

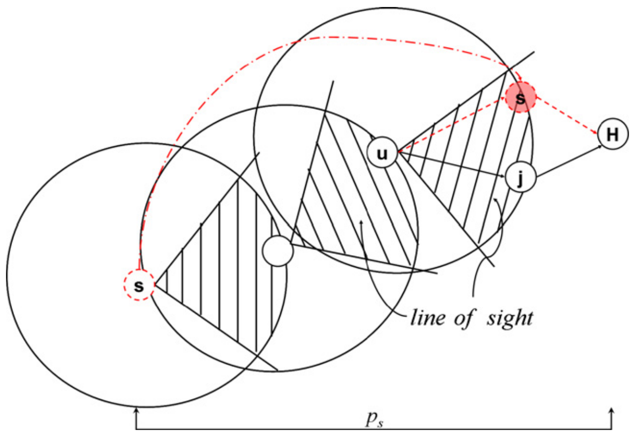

Fig. 9. The main concept of $ImpL_2(X)$.

path or a "leaf" of a tree that includes p_s (i.e. $\nexists u \in \{1, \ldots, N\} \mid s \in n_u$, where n_u is the set of the positive advance neighbours of sensor u), H is the destination and j is a critical intermediate sensor that burdens most of the traffic load in the path or tree.

The locations of all sensors $s \in \{1, 2, \ldots, N\}$ that satisfy condition $C4$ are added in the set S. Thereinafter, sensor s is relocated, i.e. a new location (x'_s, y'_s) is calculated as in $Imp_2(X)$, Eq. (17) in Section 4.4. $ImpL_2(X)$ (Fig. 9) proceeds as in Algorithm 7.

Algorithm 7. The $ImpL_2(X)$ for a subproblem i

Input: A solution X;
Output: A solution Y;
Step 1: Assign transmit power levels to solution X as in Section 2.3.
Step 2: Apply Algorithm 1 on solution X;
Step 3: Generate a set $S \subset \{1, \cdots, N\}$ with the locations of all sensors s that satisfy the condition $C4$;
Step 4: Identify the location $(x_j, y_j) \in X$ of the critical sensor j, the location $(x_u, y_u) \in X$ of sensor u and uniformly randomly select a location $(x_s, y_s) \in S$ of sensor s;
Step 5: **If** $R_c^u + R_c^j > d_{u,j}$ and $R_c^u - R_c^j < d_{u,j}$ **then**
 Step 5.1: Calculate (x'_s, y'_s) with Eq. (17);
 Step 5.2: Replace $(x_s, y_s) \in X$ with (x'_s, y'_s) and set $P_s = d_{sh}^{\alpha}$;
 Else
 Step 5.3: Uniformly randomly generate (x'_s, y'_s) so that $d_{js} \leq R_c^j$;
 Step 5.4: Replace $(x_s, y_s) \in X$ with (x'_s, y'_s) and set $P_s = d_{sv}^{\alpha}$;
 EndIf
Step 6: Output $Y = X$;

In Step 1, the power assignment of solution X is calculated. In Step 2, the algorithm that evaluates the lifetime is applied to solution X to get some network information about the current network topology (e.g. $r_j(t)$ of sensor j) as well as identify the longest path p_s of the network. In Step 3, the set S with the locations of all s that satisfy the condition $C4$ is generated. In Step 4, the locations of the critical sensor j, sensor u and sensor s are identified. In Step 5, a new location (x'_s, y'_s) is calculated and the corresponding transmit power P_s is updated (similarly to Algorithm 6). There are cases that (x'_s, y'_s) does not exist (please refer to Section 4.4 for more details).

Remark. This method focuses at improving the lifetime objective having a negative impact on the coverage objective, due to the re-location of sensors that are far away from H. Therefore, this method mainly favours the solutions of area a and might favour the solutions of area b (i.e. X^a, X^b).

4.6. Improve-coverage/affecting lifetime ($ImpCv_2(X)$): re-locating sensor for decreasing coverage holes

$ImpCv_2(X)$ aims at minimizing the network coverage holes for maximizing the network coverage. A coverage hole is defined as a set of consecutive uncovered grid locations, denoted as Ψ with size $|\Psi|$. $ImpCv_2(X)$ calculates the coverage holes of a particular network design X using Algorithm 8.

Algorithm 8. The coverage holes of a solution X

Input: A solution X;
Output: A set of locations Ψ;
Step 0: Set $\psi = 1$;
Step 1: **While** $\exists g(x', y') = 0 \mid (x', y') \in A$ and $(x', y') \notin \Psi_j$, $j = 1, \ldots, \psi$ **do**
 Step 1.1: Set $\Psi_\psi = \Psi_\psi \cup (x', y')$;
 Step 1.2: **While** $\exists (x'', y'') = 0 \mid (x'', y'') \in A$ **do**
 $\Psi_\psi = \Psi_\psi \cup (x'', y'')$,
 where (x'', y'') are all consecutive grids of each $(x', y') \in \Psi_\psi$ with $g(x'', y'') = 0$;
 Step 1.3: $\psi = \psi + 1$;
Step 2: Output Ψ_j, where $|\Psi_j| \geq |\Psi_k|$, for $k = 1, \ldots, \psi$;

In Step 1, the coverage holes Ψ_j are identified, where $j = 1, \ldots, \psi$. The loop continues until all uncovered grids are added in a coverage hole. In Step 1.1, a new coverage hole Ψ_j is identified and the location of its first uncovered grid is added. In Step 1.2, the locations of all the consecutive uncovered grids are iteratively added in the current coverage hole, Ψ_j. Step 2 outputs the largest coverage hole Ψ_j, i.e. the coverage hole with the highest number of locations, where $j = 1, \ldots, \psi$ and ψ is the total number of holes in X.

Algorithm 9. The $ImpCv_2(X)$ for a subproblem i

Input: A solution X;
Output: A solution Y;
Step 1: Generate a set $S \subset \{1, \cdots, N\}$ with the locations of all sensors s that satisfy the conditions $C1$, $C2$, $C3$;
Step 2: **While** $S \neq \emptyset$ and $Cv(X) \neq 1$ **do**
 Step 2.1: Find the largest coverage hole Ψ with Algorithm 8;
 Step 2.2: Uniformly randomly select a new location $(x'_s, y'_s) \in \Psi$, such that $\exists u \in \{1, \ldots, N\} \mid d_{su} \leq R_{max}$;
 Step 2.3: Replace $(x_s, y_s) \in X$ with (x'_s, y'_s) and set $P_s = d_{su}^{\alpha}$;
 Step 2.4: Remove (x_s, y_s) from S;
Step 3: Output $Y = X$;

Thereinafter, $ImpCv_2(X)$ identifies the locations of all sensors s that satisfy the constraints $C1$, $C2$ and $C3$, defined in Section 4.4, and add them in set S. A new location (x'_s, y'_s) is then uniformly randomly selected from Ψ, so that $\exists u \in \{1, \ldots, N\} \mid d_{su} \leq R_{max}$. $ImpCv_2(X)$ is outlined in Algorithm 9 and exemplified in Fig. 10.

In Step 1, the locations of all sensors that satisfy the constraints $C1$, $C2$ and $C3$ are identified and added in the set S. In Step 2, the coverage holes are iteratively minimized until there are no more sensor locations $(x_s, y_s) \in S$ that satisfy all conditions and area A is not fully covered (i.e. $Cv(X) \neq 1$). In each iteration, the largest coverage hole Ψ is found by Algorithm 8 and a new location (x'_s, y'_s) is uniformly randomly selected from Ψ and replaces $(x_s, y_s) \in X$. The corresponding transmit power level P_s is updated accordingly.

Remark. This method focuses at improving the coverage objective having a negative impact on the lifetime objective, since the re-location of sensors may overwhelm the sensors close to H. Therefore, it mainly favours the solutions of area c and might favour the solutions of area b (i.e. X^b, X^c).

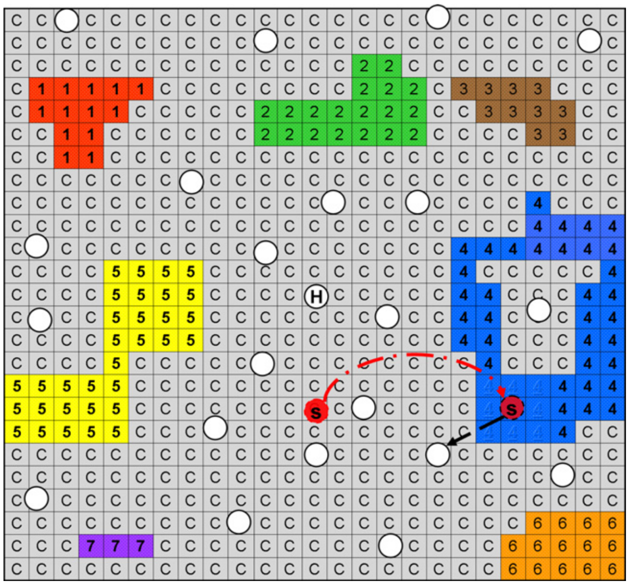

Fig. 10. The main concept of $ImpCv_2(X)$. The network design is divided into $\psi = 7$ coverage holes Ψ_j, where $j = 1, \ldots, 7$. The covered grids are denoted as c. The sensors are marked as circles. Sensor s satisfies all conditions $C1$, $C2$, $C3$ and is relocated within the largest coverage hole, i.e. Ψ_4, to cover a previously uncovered area (shown as embossed 4).

4.7. The Generalized Subproblem-dependent Heuristic (GSH)

In this section, we firstly classify the six improvement strategies into three sets, based on the way that they treat the solution X of a subproblem i. Then we design the Generalized Subproblem-dependent heuristic that probabilistically controls the six improvement strategies and tackles each subproblem according to its requirements and objective preference.

The six improvement strategies are classified into the following three sets:

- S_I: Composed of the improvement strategies that benefit the solutions of all m subproblems (i.e. for all λ^i).
- S_{II}: Composed of the improvement strategies that benefit the solutions of the subproblems which prefer a high network lifetime (i.e. for high λ^i).
- S_{III}: Composed of the improvement strategies that benefit the solutions of the subproblems which prefer a high network coverage (i.e. for low λ^i).

Then, each improvement strategy is added in a set as follows:

- $Imp_1(X)$ (defined in Section 4.3) and $Imp_2(X)$ (defined in Section 4.4) might increase the network lifetime without affecting coverage (coverage remains the same in the worst case). Thus, they are classified as S_I strategies, since they may favour any solution of the objective space. The solutions of subproblems that favour a high network coverage, e.g. the solutions of area c (i.e. X^c), may also be favoured by these strategies, since an improved solution $X^{c'}$ dominates a solution X^c if $Cv(X^{c'}) = Cv(X^c)$ and $L(X^{c'}) > L(X^c)$.
- $ImpL_1(X)$ (defined in Section 4.2) and $ImpL_2(X)$ (defined in Section 4.5) are classified as S_{II} strategies, since they favour the lifetime objective and it is very likely to have a negative impact on coverage.
-]$ImpCv_1(X)$ (defined in Section 4.1) and $ImpCv_2(X)$ (defined in Section 4.6) are classified as S_{III} strategies, since they favour the coverage objective and it is very likely to have a negative impact on lifetime.

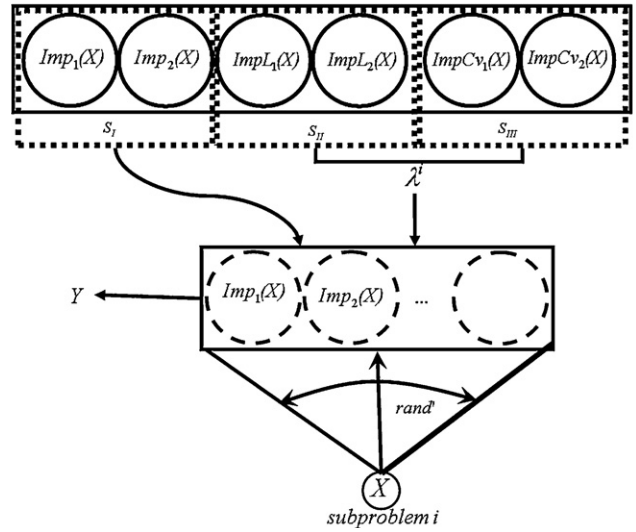

Fig. 11. The main concept of the Generalized Subproblem-dependent Heuristic (GSH)

Finally, the following sets of improvement strategies are designed,

$S_I = \{Imp_1(X), Imp_2(X)\}$,
$S_{II} = \{ImpL_1(X), ImpL_2(X)\}$,
$S_{III} = \{ImpCv_1(X), ImpCv_2(X)\}$,

for the MOEA/D-GSH to efficiently tackle the d-DPAP. The GSH aims at improving the solution of each subproblem i as outlined in Algorithm 10 (Fig. 11).

Algorithm 10. The Generalized Subproblem-dependent Heuristic (GSH) for subproblem i

Input: A solution X;
Output: A solution Y;
Step 0: Set $SS = \emptyset$; S_I; S_{II}; S_{III};
Step 1: Generate a uniform random number $rand \in [0, 1]$;
Step 2:

$$SS = \begin{cases} S_I \cup S_{II} & \text{if } rand \le \lambda^i, \\ S_I \cup S_{III} & \text{otherwise,} \end{cases}$$

Step 3: Uniformly randomly generate an integer $rand'$ from $\{1, 2, \ldots, |SS|\}$;
Step 4: Apply improvement strategy $rand'$ to X to obtain Y;

In Steps 1 and 2, the superset SS is initialized, including the low-level improvement strategies of S_I and either the improvement strategies of S_{II} or the improvement strategies of S_{III}, taking into consideration the objective preference of each subproblem (i.e. the weight coefficient λ^i). In Steps 3 and 4, an improvement strategy is uniformly randomly selected from the superset and is applied to X to obtain Y.

Note that more low-level improvement strategies can be designed and adopted by the proposed GSH. Besides, the improvement strategies can be further classified into more sets, considering more requirements of different areas of the objective space.

5. Performance metrics

This section briefly describes the metrics [25,48,49,31,50] adopted from the literature for comparing the performance of MOEAs. Comparing set of solutions obtained by different MOEAs is not straightforward, since there is not a single metric that can satisfy all requirements (e.g. convergence, diversity). Therefore, a set of performance metrics [25,26,49] is often used for this purpose, including the IGD-metric, the hypervolume measure, the Δ-metric, C-metric etc. In this paper, the d-DPAP network topologies are designed offline and there is no critical need of real-time decision making. The main focus of our experimental studies is to evaluate the performance of the MOEAs in approximating the PF. Thus, in the absence of the real PF of a d-DPAP instance, the following four metrics are adopted:

- **Δ-Metric**[25]: is a diversity metric that measures the extent of spread of the solutions in the Pareto-optimal set. In the case of two objectives, the Δ value of a set of candidate solutions A is defined as follows:

$$\Delta(A) = \frac{d_f + d_l + \sum |d_j - \overline{d}|}{d_f + d_l + |A|\overline{d}},$$

  where d_f and d_l are the extreme Pareto optimal solutions in the objective space, d_j is the distance between two neighbouring solutions and $\overline{d}$ is the mean of all the distribution. $\Delta(A)$=0 means a uniform spread of solutions in the objective space, therefore a lower $\Delta(A)$ is preferable.
- **Coverage (C)-metric**[48]: is a commonly used metric for comparing two sets of non-dominated solutions A and B, originally proposed by Zitzler and Thiele [48]. The $C(A, B)$ metric, which is often considered as a MOEA quality metric, calculates the ratio of the non-dominated solutions in B dominated by the non-dominated solutions in A, divided by the total number of non-dominated solutions in B. Hence,

$$C(A, B) = \frac{|\{x \in B | \exists y \in A : y \succ x\}|}{|B|}.$$

  Therefore, $C(A, B) = 1$ means that all non-dominated solutions in B are dominated by the non-dominated solutions in A. Note that $C(A, B) \neq 1 - C(B, A)$.
- **Non-dominated solutions($NDS(A)$)**[31,50]: a straightforward metric usually considered in cases of real-life discrete optimization problems such as d-DPAP, is the cardinality or the number of Non-dominated solutions in set A, i.e.

$$NDS(A) = |A|.$$

  In these cases, it is more desirable to obtain a high number of $NDS(A)$ in order to provide an adequate number of Pareto optimal choices. In contrast, and usually in cases of continuous optimization [32], a high number of NDS is not desirable, since the decision making procedure becomes more complicated and more time consuming. However, the NDS should be considered in combination with other metrics (e.g. Δ and C metrics), since it is usually desirable to have a high number of NDS when the solutions is of high quality (i.e. low C-metric) and spread (i.e. low Δ-metric) in the objective space.
- **Width($wdt_{f_i}(A)$)**[49]: the width of each objective $f_1(x), f_2(x)$ over a reference set of solutions A obtained by an algorithm is used in some cases as a performance metric as follows:

$$wdt_{f_i}(A) = \max\{f_i(x) | x \in A\} - \min\{f_i(x) | x \in EP^A\}.$$

  The wider a PF is, the better. However, the width metric should be taken into account in combination with the other metrics as well.

Table 1
Network instances.

Network instances	A (m^2)	N	Density (N/A)
NIn1	2500 (50 × 50)	25	0.01
NIn2	2500 (50 × 50)	50	0.02
NIn3	2500 (50 × 50)	63	0.025
NIn4	2500 (50 × 50)	38	0.015
NIn5	3500 (70 × 50)	35	0.01
NIn6	3500 (70 × 50)	53	0.015
NIn7	3500 (70 × 50)	70	0.02
NIn8	3500 (70 × 50)	89	0.025
NIn9	5000 (50 × 100)	50	0.01
NIn10	5000 (50 × 100)	75	0.015
NIn11	5000 (50 × 100)	100	0.02
NIn12	5000 (50 × 100)	125	0.025
NIn13	10,000 (100 × 100)	100	0.01
NIn14	10,000 (100 × 100)	150	0.015
NIn15	10,000 (100 × 100)	200	0.02
NIn16	10,000 (100 × 100)	250	0.025

- **CPU time**: Besides, it is always desirable to obtain high quality solutions within an acceptable CPU time.

Therefore, the set of metrics adopted and just described (i.e. the NDS and wdt in combination with the C and Δ metrics as well as the required CPU time) can be considered adequate for comparing the performance of the concerned algorithms and drawing insightful conclusions for their efficiency and effectiveness.

6. Experimental results and discussion

The goals of our experimental studies are to (a) study the effect of the proposed Generalized Subproblem-dependent improvement Heuristic (GSH) on the performance of the general purpose MOEA/D, (b) test the strength of the proposed specialized decompositional approach (i.e. MOEA/D hybridized with the problem-specific GSH) against the state-of-the-art in Pareto-dominance MOEAs, the NSGA-II [25], in various network instances and (c) study the behaviour of MOEA/D-GSH on d-DPAP in WSNs. Note that, the proposed GSH can be easily hybridized with other decompositional MOEAs, e.g. MOGLS [40], NSGA-II/MOGLS [51], etc., with minor changes in the algorithms' general purpose framework. However, this is out of the scope of this paper.

6.1. Network test instances and parameter settings

We examine 16 network test instances, denoted as NIn and designed using the popular Factorial design process [52], which represent a broad class of the small-scale and dense [53] d-DPAP WSN topologies. Their characteristics are shown in Table 1.

In our experimental studies, the parameters are set as in [7]. That is, max number of generations $gen_{max} = 250$, population size and number of subproblems $m = 120$, crossover rate $r_c = 0.9$, mutation rate $r_m = 0.5$, tournament size $M = 20$ and neighbourhood size $T = 2$. Moreover, in all simulation studies the following network parameters are set [45,54]: $R_s/R_{max} = 100/200$, $E = 5J$, $d_{min} = 100$ m, $a = 2$, $amp = 100$ pJ/bit/m^2 and square-grids of side length 10 m. The network lifetime and coverage are evaluated as in Section 2.2 and the lifetime objective is normalized by the $L(X^A)$ as in [34]. All algorithms were coded in Java programming language and run on an Intel/circledR Pentium 4 3.2 GHz Windows XP server with 1.5 GB RAM. To increase the fidelity of our experimental studies we have repeated each experiment of each algorithm for 30 independent runs, having the same number of function evaluations for fairness.

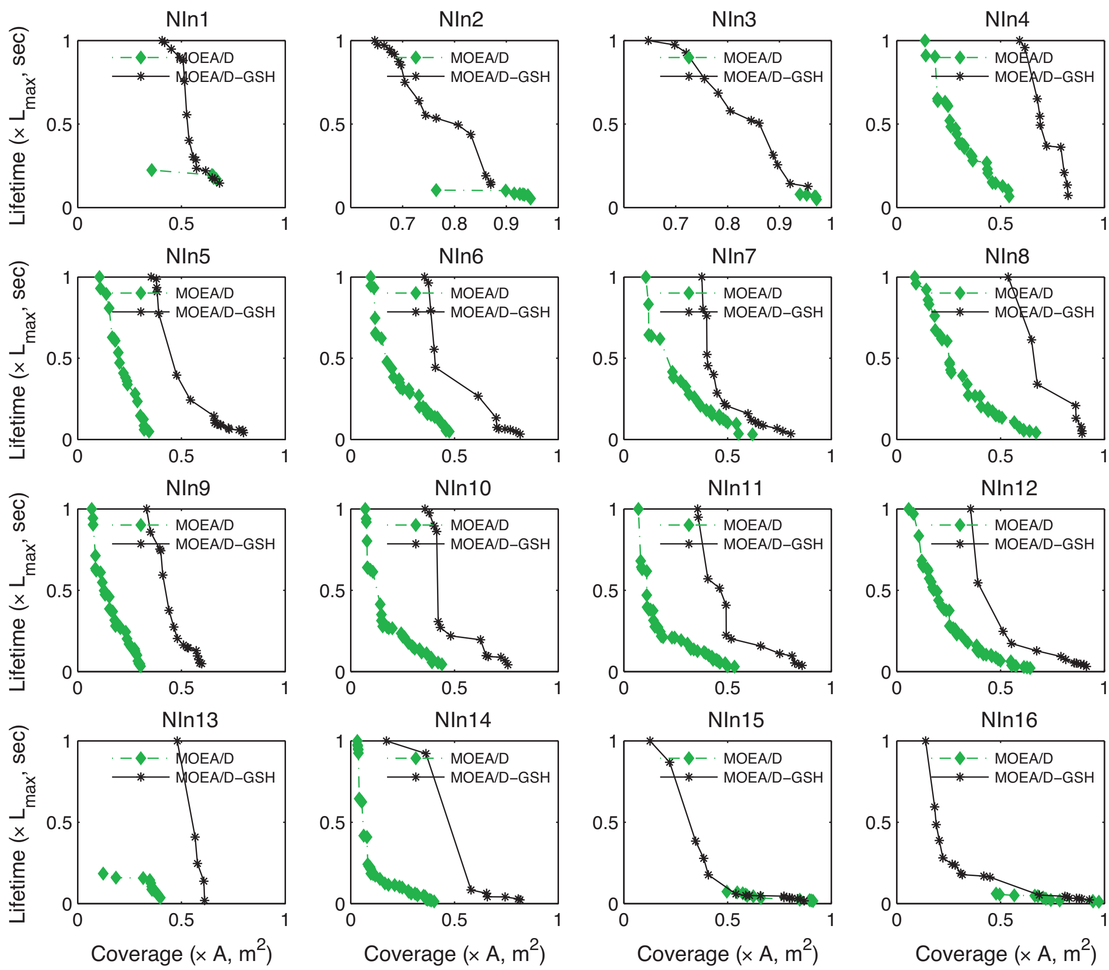

Fig. 12. MOEA/D-GSH vs. general-purpose MOEA/D in NIn1–16.

6.2. The hybridization of MOEA/D with the GSH

In Step 2.2 of the MOEA/D (Section 3.2), the solution of each subproblem is improved by the GSH, detailed in Section 4. In this subsection, we study the effect of the hybridization of the MOEA/D with the GSH. To do so, the performance of the MOEA/D with and without the improvement heuristic, i.e. MOEA/D-GSH and MOEA/D, respectively, is compared in NIn1, 2 and 3.

Fig. 12 shows the PFs obtained by each algorithm and Table 2 summarizes their statistical performance. From these results we can see that, MOEA/D-GSH outperforms MOEA/D in all network instances, producing very diverse non-dominated network designs. Specifically, MOEA/D-GSH provides a better average diversity $\Delta = 0.7968$ than the average $\Delta = 0.8565$ obtained by MOEA/D. Moreover, the non-dominated solutions obtained by MOEA/D-GSH dominate 79% of the non-dominated solutions obtained by MOEA/D, on average. This comes at the cost of a poorer compu-

Table 2
MOEA/D-GSH (denoted as G) vs. MOEA/D (denoted as M) in NIn1–16.

NIn	Δ (M)	Δ (G)	NDS(M)	NDS(G)	CPU(M)	CPU(G)	C(G,M)	C(M,G)
NIn1:	0.9209	**0.8127**	5	**15**	**0.15**	0.47	**0.2**	0.1
NIn2:	0.9475	**0.8432**	8	**17**	**0.26**	3.79	**0.1**	0.0
NIn3:	0.9867	**0.7001**	5	**12**	**0.32**	6.53	**0.4**	0.0
NIn4:	0.8262	**0.7561**	**30**	10	**0.18**	3.61	**1.0**	0.0
NIn5:	**0.8104**	0.8421	**20**	18	**0.20**	5.28	**1.0**	0.0
NIn6:	0.8228	**0.7541**	**30**	13	**0.22**	6.07	**1.0**	0.0
NIn7:	0.8157	**0.7704**	**24**	17	**0.27**	5.64	**1.0**	0.0
NIn8:	0.7637	**0.7483**	**34**	8	**1.10**	4.05	**1.0**	0.0
NIn9:	0.8350	**0.7854**	**32**	16	**0.87**	4.77	**1.0**	0.0
NIn10:	**0.8436**	0.8751	**31**	13	**0.31**	7.62	**1.0**	0.0
NIn11:	0.8726	**0.7875**	**39**	13	**0.82**	10.43	**1.0**	0.0
NIn12:	**0.8155**	0.8344	**47**	12	**8.65**	13.89	**1.0**	0.0
NIn13:	0.9288	**0.8120**	**13**	5	**0.88**	11.49	**1.0**	0.0
NIn14:	0.9200	**0.8863**	**38**	8	**0.84**	13.91	**1.0**	0.0
NIn15:	**0.9060**	0.9377	12	**16**	**1.59**	24.43	**0.3**	0.1
NIn16:	0.8914	**0.8064**	12	**18**	**2.95**	41.23	**0.6**	0.0

Note that in all cases the best performances are denoted in bold.

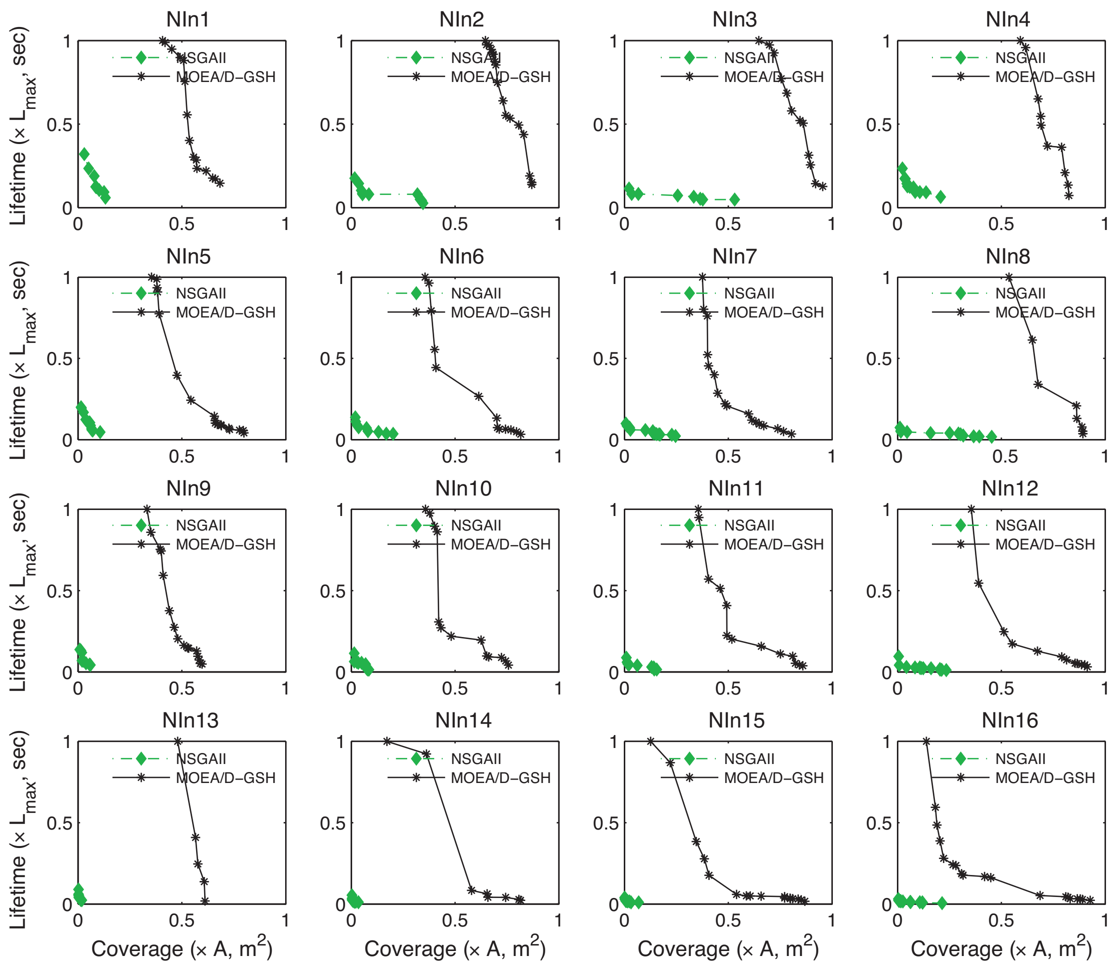

Fig. 13. MOEA/D-GSH vs. NSGA-II in NIn1–16.

tational effort. The average number of NDS obtained by the two approaches is the same.

The performance of the proposed MOEA/D-GSH approach is investigated against the popular NSGA-II [25]. NSGA-II maintains a population of solutions of size *m* at each generation *gen* and adopts evolutionary components for offspring reproduction as MOEA/D. The key characteristic of NSGA-II is that it uses a fast non-dominated sorting and a crowded distance estimation for comparing the quality of different solutions during selection and update. NSGA-II utilizes a tournament selection, a two-point crossover (2×) and a random mutation operators. The MOEAs are compared in all 16 network test instances summarized in Table 1, considering the same parameter settings for fairness. Note that, in this paper the same number of generations implies the same number of function evaluations. The reason is that in the proposed MOEA/D, the GSH is used just once for each subproblem i per generation, where $i = 1 \ldots m$. The new solution Z^i is then evaluated once. Solution Z^i replaces the current best solution X^i of subproblem i iff

Table 3
MOEA/D-GSH (denoted as M) vs. NSGA-II (denoted as N) in NIn1–16.

Net.	Δ		NDS		CPU		C		wdt_L		wdt_{Cv}	
Ins.:	(N)	(M)	(N)	(M)	(N)	(M)	(M,N)	(N,M)	(N)	(M)	(N)	(M)
NIn1:	0.9439	**0.8127**	8	**15**	**0.20**	0.47	**1.0**	0.0	0.261	**0.854**	0.102	**0.247**
NIn2:	0.9196	**0.8432**	10	**17**	**0.31**	3.79	**1.0**	0.0	0.149	**0.862**	**0.329**	0.224
NIn3:	0.8991	**0.7001**	8	**12**	**0.39**	6.53	**1.0**	0.0	0.067	**0.874**	**0.509**	0.308
NIn4:	0.9433	**0.7561**	**11**	10	**0.26**	3.61	**1.0**	0.0	0.170	**0.928**	0.184	**0.234**
NIn5:	0.9629	**0.8421**	12	**18**	**0.29**	5.28	**1.0**	0.0	0.154	**0.958**	0.092	**0.444**
NIn6:	0.9541	**0.7541**	9	**13**	**0.37**	6.07	**1.0**	0.0	0.102	**0.967**	0.182	**0.461**
NIn7:	0.9489	**0.7704**	13	**17**	**0.47**	5.64	**1.0**	0.0	0.077	**0.965**	0.239	**0.429**
NIn8:	0.9140	**0.7483**	**12**	8	**0.60**	4.05	**1.0**	0.0	0.057	**0.963**	**0.442**	0.356
NIn9:	0.9778	**0.7854**	8	**16**	**0.47**	4.77	**1.0**	0.0	0.095	**0.952**	0.051	**0.265**
NIn10:	0.9728	**0.8751**	9	**13**	**0.62**	7.62	**1.0**	0.0	0.100	**0.958**	0.070	**0.399**
NIn11:	0.9652	**0.7875**	9	**13**	**0.80**	10.43	**1.0**	0.0	0.073	**0.962**	0.146	**0.504**
NIn12:	0.9473	**0.8344**	**13**	12	**1.03**	13.89	**1.0**	0.0	0.084	**0.969**	0.229	**0.558**
NIn13:	0.9869	**0.8120**	**10**	5	**1.22**	11.49	**1.0**	0.0	0.066	**0.982**	0.015	**0.132**
NIn14:	0.9880	**0.8863**	**12**	8	**1.71**	13.91	**1.0**	0.0	0.043	**0.977**	0.032	**0.644**
NIn15:	0.9844	**0.9377**	12	**16**	**2.71**	24.43	**1.0**	0.0	0.030	**0.983**	0.067	**0.742**
NIn16:	0.9598	**0.8064**	13	**18**	**2.98**	41.23	**1.0**	0.0	0.026	**0.977**	0.211	**0.789**

Note that in all cases the best performances are denoted in bold.

Please cite this article in press as: A. Konstantinidis, K. Yang, Multi-objective energy-efficient dense deployment in Wireless Sensor Networks using a hybrid problem-specific MOEA/D, Appl. Soft Comput. J. (2011), doi:10.1016/j.asoc.2011.02.031

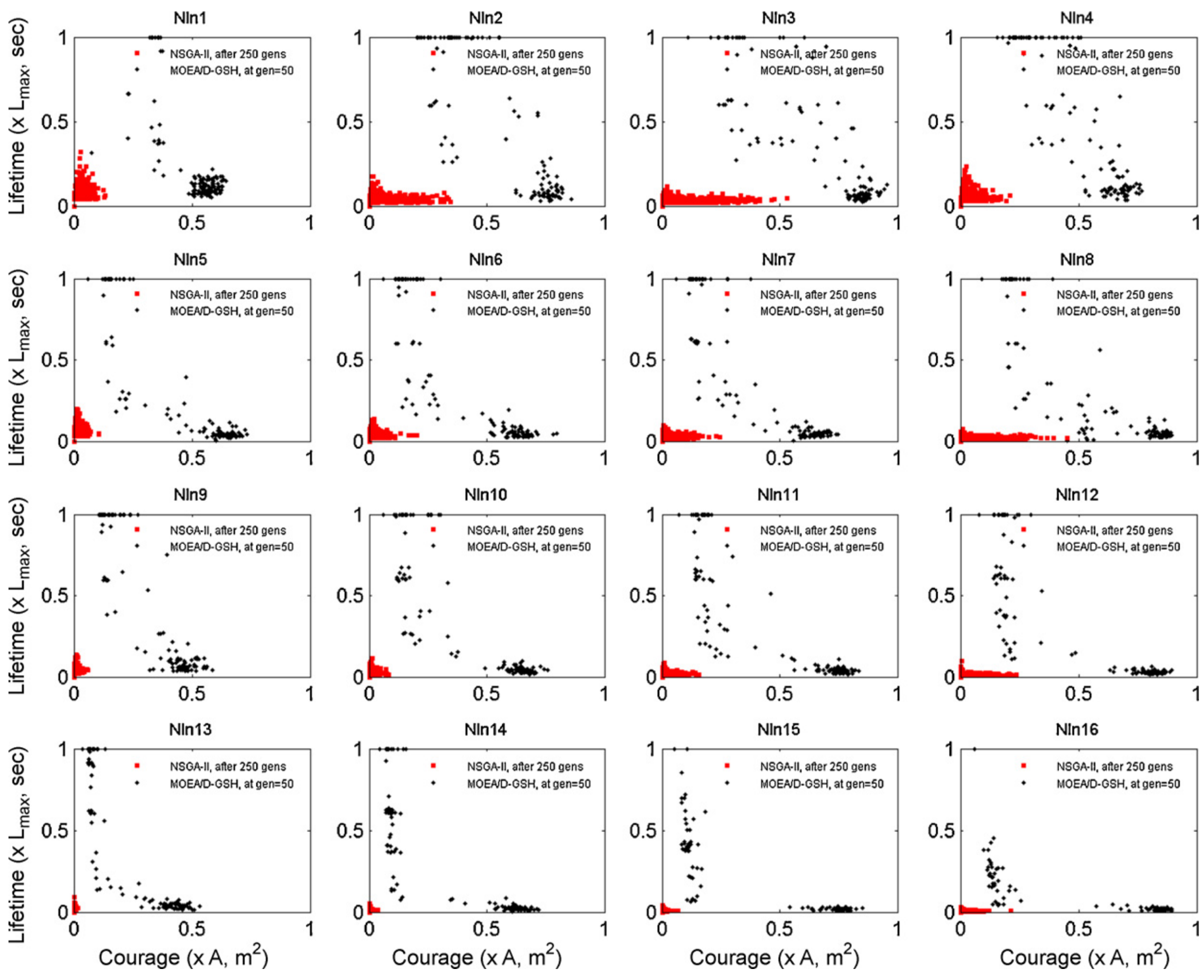

Fig. 14. The overall objective space searched by NSGA-II in 250 generations vs. the internal population of MOEA/D-GSH at the 50th generation in NIn1–16.

$g^i(Z^i, \lambda^i) > g^i(X^i, \lambda^i)$. Therefore, we count m function evaluations per generation. This results in $m \times gen_{max}$ function evaluations per run, which is also the same as the function evaluations used in both the general-purpose MOEA/D and NSGA-II.

The results in Fig. 13 show that MOEA/D-GSH outperforms NSGA-II in all network instances, giving non-dominated network designs of both higher coverage and lifetime, which dominate every network design in NSGA-II's PF and none is dominated. NSGA-II is not good in obtaining high quality network designs due to the unnecessary searches of its generic components in the infeasible regions of the search space and/or the areas where near-optimal network designs are difficult to be obtained. A statistical comparison between the two methods is summarized in Table 3. The results show the superiority of MOEA/D in terms of quality, diversity, width and number of NDS in the PF at the cost of a higher computational effort.

In the following the searching behavior of the two MOEAs is studied and compared in terms of convergence speed. Fig. 14 shows the overall searching ability of MOEA/D and NSGA-II in the objective space as well as a comparison in terms of the convergence speed. Note that 30,000 network topologies are designed by each MOEA in the whole evolution. Based on the results,

MOEA/D explores the objective space more efficiently, giving solutions in almost the whole range of the objective space, with respect to NSGA-II which generally obtains poor solutions. MOEA/D converges faster than NSGA-II in all network instances, since the set of solutions obtained by MOEA/D after 50 generations outperforms the set of solutions obtained by NSGA-II during the whole search. However, the deterministic nature of GSH may lead the proposed approach to a premature convergence (this will be discussed in more details shortly).

More insights on the conclusions just mentioned are provided by the following results. Fig. 15 shows the distribution of the solutions in various generations for both (a) MOEA/D and (b) NSGA-II in NIn3. The results show that MOEA/D obtains a variety of network designs in IP_{gen} from the beginning of the search, giving more focus on the network designs that prefer high network coverage (i.e. area c). In about the middle of the evolution, MOEA/D starts searching areas a and b of the objective space more effectively, obtaining solutions of higher quality. Finally, MOEA/D obtains non-dominated network designs of high quality across almost the whole range of the PF. In contrast, NSGA-II is trapped from the very beginning into network designs of poor lifetime and coverage.

In conclusion, MOEA/D-GSH is shown efficient and effective in dealing with the d-DPAP in WSNs with respect to NSGA-II. This is due to the fact that NSGA-II, and most MOEAs based on Pareto dominance, try to deal with the MOP as a whole and without any problem-domain knowledge (i.e. as a "black box"). This results in having difficulties to explore the search space efficiently and find feasible near-optimal solutions fast. The decompositional nature of MOEA/D, on the other hand, alleviates this difficulty by decomposing the MOP into many single objective subproblems, allowing the incorporation of single-objective problem-domain knowledge (e.g. the GSH proposed here) in a simple manner. This directs the search into good feasible regions of the objective

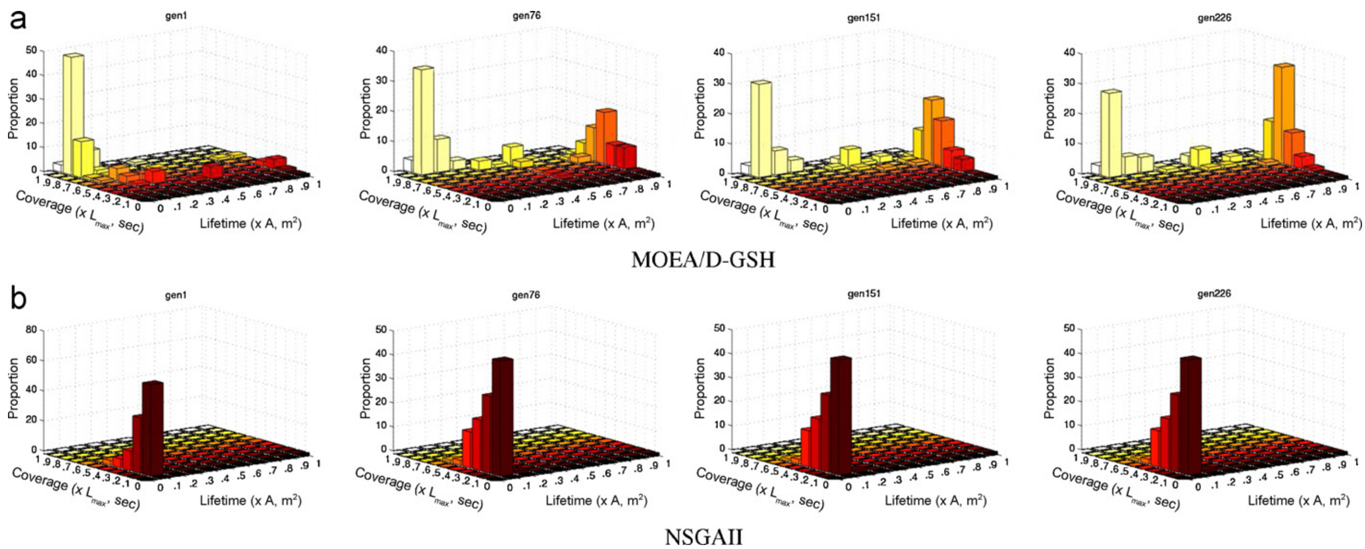

Fig. 15. The proportion of the solutions distribution in the objective space in several generations for NIn3.

space, improving the convergence and diversity of the proposed MOEA/D-GSH approach. Note that the underlying idea behind the proposed problem-specific heuristic might also shed some light on the design of MOEA/Ds (or other decompositional MOEAs, e.g. MOGLS [40], NSGA-II/MOGLS [51]) for other MOPs from different disciplines.

6.3. *More discussion on the behavior of MOEA/D-GSH in the objective space in terms of entropy*

Finally, we examine the behavior of the MOEA/D in the objective space of the d-DPAP in terms of diversity in the IP nd convergence using the Shannon's entropy [55]. Entropy shows the variation of

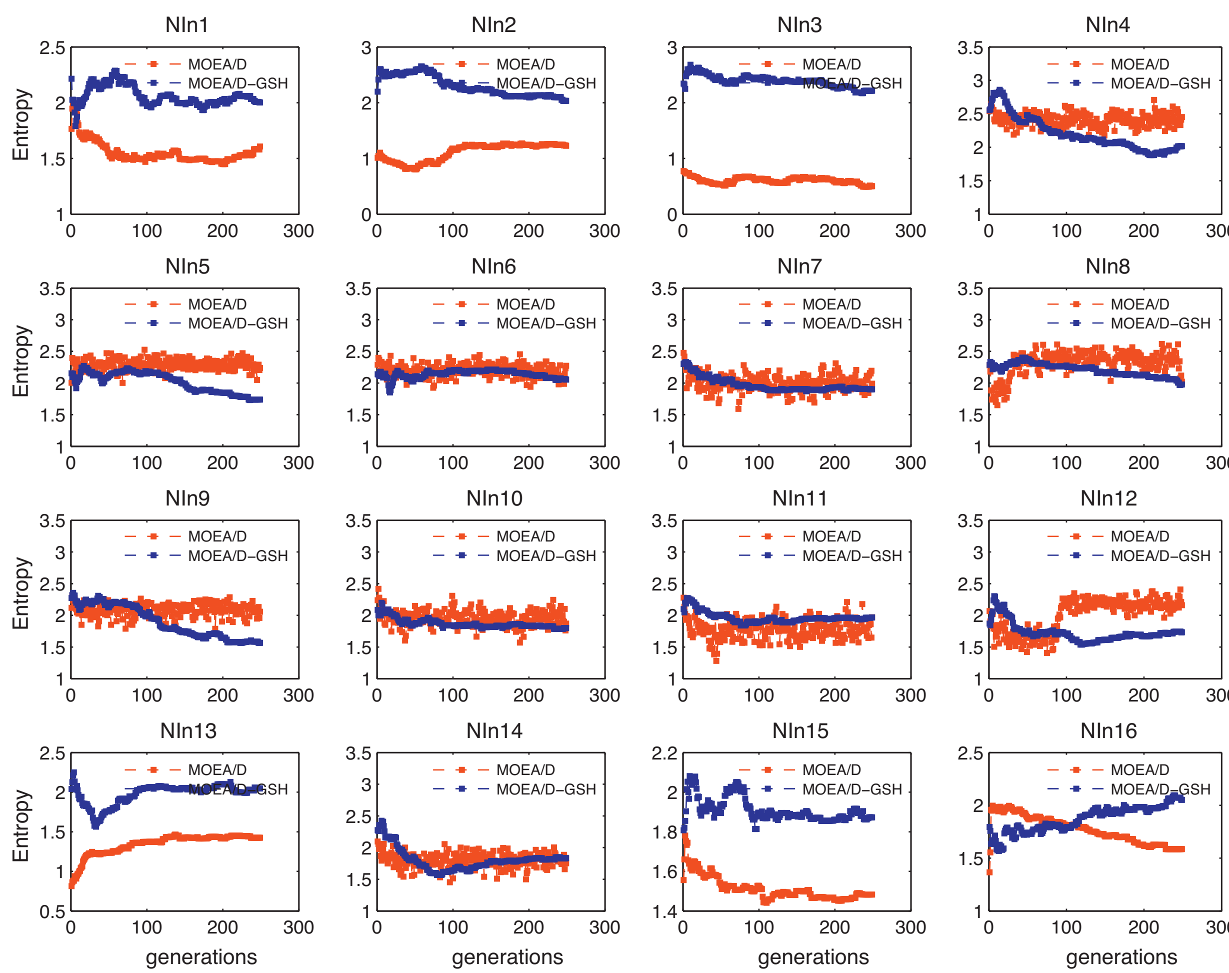

Fig. 16. MOEA/D-GSH vs. MOEA/D: entropy of the IP during the evolution in NIn1–16.

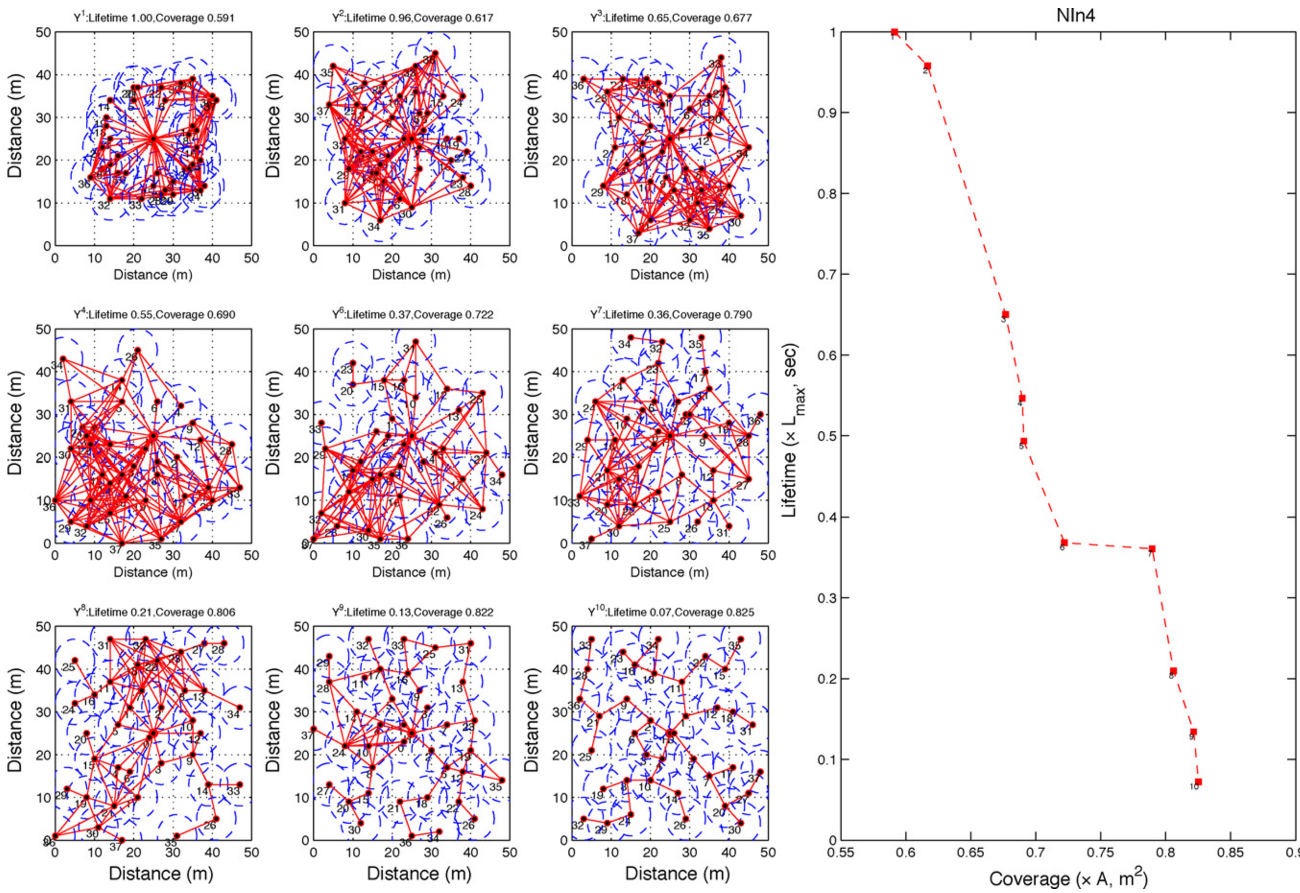

Fig. 17. Decoding non-dominated solutions of different areas of the PF, obtained by MOEA/D-GSH in NIn4.

the internal population between two consecutive generations for the results obtained by the MOEA/D in all 16 network instances, as follows:

$$\text{Entropy} = -\sum_{j=1}^{121} p(x_j)\log(p(x_j)), \qquad (18)$$

where $p(x_j)$ is the probability mass function that gives the probability of a variable to be equal to some value. In our case, the lifetime and coverage of each solution are rounded up, and $p(x_j)$ is estimated as the proportion of the population that has the same unique x_j=(lifetime,coverage) in each generation (note that the total number of unique values is 121).

The results of Fig. 16 show that the diversity of the IP of MOEA/D-GSH is high in most cases compared to MOEA/D. When the area size is small (i.e. NIn1–9), the MOEA/D-GSH initially obtains a diverse set of solutions, in which entropy increases in the first few iterations and then decreases or remain relatively fixed. When the area size increases, the entropy decreases at the beginning of the evolution and then remains relatively fixed (i.e. NIn10–11) or increases (NIn12–15). In that case, MOEA/D-GSH obtains a diverse set of solutions at the very beginning of the search and then requires a few iterations (about 100 generations) to explore the objective space and reach other feasible regions. Thus, one can say that the d-DPAP becomes harder as the area size increases. In NIn16 (i.e. the highest number of sensors and the largest area size studied in this section), the entropy increases almost linearly. On the other hand, the general purpose MOEA/D's entropy is low from the beginning until the end of the evolution when the area size is small, as well as relatively fixed (e.g NIn1–12) and similar in few network instances to the entropy of MOEA/D-GSH. When the area size is large (i.e. NIn13–16), however, it seems that the general purpose MOEA/D finds it difficult to explore the objective space having a low entropy that in most cases decreases during the evolution. Therefore, the problem-specific GSH may lead the proposed approach into a premature convergence only when simple WSN test instances are considered. In that case, MOEA/D-GSH may obtain a diverse high-quality PF, compared to the PF obtained by either NSGA-II or the general purpose MOEA/D, with fewer function evaluations and therefore less computational effort.

6.4. Decoding the PF of the d-DPAP and a case study example in WSNs

In this subsection, we illustrate the decoding of the PF, obtained by the MOEA/D in NIn4 of d-DPAP, which can be used for decision making. Fig. 17 shows the following Pareto optimal network designs: Y^1, Y^2, Y^3, Y^4, Y^6, Y^7, Y^8, Y^9 and Y^{10}, which represent different non-dominated solutions in the objective space (please refer to Section 3.1 for details). Solution Y^1 represents the approximation towards the extreme solution X^A, having most sensors densely deployed around H. Solutions Y^2 and Y^3 can be classified as solutions of area a, having most sensors directly connected to H and the remaining sensors with high transmit power levels and high node-degree. Solutions Y^4, Y^6 and Y^7 can be classified as solutions of area b. These network designs are composed of both sensors which are deployed close to each other with high node-degree to benefit the network lifetime and sensors which are spread away to benefit the network coverage. Solutions Y^8 and Y^9 represent Pareto optimal solutions of area c, having the sensors spread in the area, with low node-degree and transmit power levels. Finally, solution Y^{10}

Please cite this article in press as: A. Konstantinidis, K. Yang, Multi-objective energy-efficient dense deployment in Wireless Sensor Networks using a hybrid problem-specific MOEA/D, Appl. Soft Comput. J. (2011), doi:10.1016/j.asoc.2011.02.031

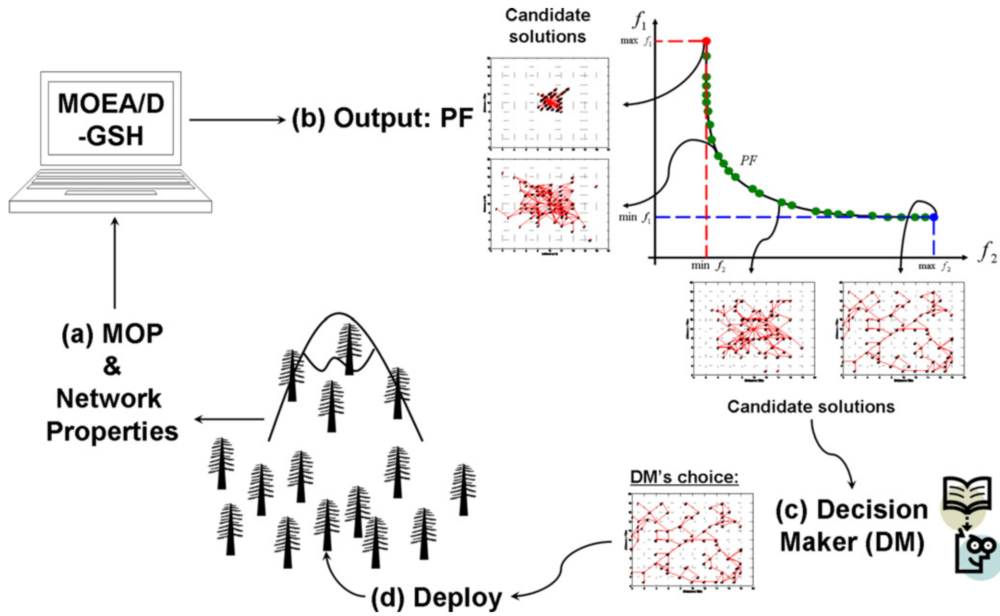

Fig. 18. MOEA/D-GSH as a sensor placement tool in a monitoring information system for fire detection applications in forests.

represents the approximation towards the extreme solution X^B of NIn4.

Case study: MOEA/D-GSH can be used as a sensor placement tool for fire-detection applications in forests. For example, MOEA/D-GSH could be embedded into a monitoring information system (e.g. Fire Management System (FMS),[1] FireWatch,[2] Koetter Fire Protection,[3] etc.) to enable real time detection of fires in forests. The system may be composed of several WSN technological components, including an efficient and effective placement strategy component, as well as a terrain analysis and digital terrain modeling component, a fire detection, prediction system component, etc. MOEA/D-GSH can be utilized in obtaining and providing a diverse set of Pareto-optimal network design choices (please refer to the decoding example discussed earlier in this section) to the network manager before making his/her final decision. Note that, the WSN system model can be easily modified to include various network parameters, e.g. terrain properties as well as irregular area shapes, and tackled by MOEA/D-GSH with minor changes in the algorithm's structure. The design choices that the MOEA/D-GSH will provide are, for example, long-lived topologies covering small sensitive parts of the forest, or sparse topologies covering almost the whole forest for short critical periods of time (e.g. summer), or balanced topologies etc. The MOEA/D-GSH as a sensor placement component in a fire detection system is illustrated in Fig. 18.

7. Conclusions and future research

In this work, a Generalized Subproblem-dependent Heuristic (GSH) is proposed and successfully hybridized with MOEA/D for tackling the multi-objective Dense Deployment and Power Assignment Problem in WSNs. Initially, the d-DPAP is decomposed into a number of scalar subproblems and discussed based on the subproblems objective preferences. Then, the GSH is introduced, i.e. a probabilistic mixture of six d-DPAP specific improvement strategies, each having different properties and directing the search into specific areas of the objective space. Finally, simulation results have shown that the hybridization of the proposed GSH with the MOEA/D obtains better results than the general MOEA/D and the popular NSGA-II in several WSN test instances. Moreover, the behavior of the MOEAs in the objective space is discussed using different metrics, such as the Shannon's entropy, giving important insights.

There is a number of avenues for further research. For example, it will be interesting to extend the GSH, using more problem-specific heuristics having different properties to explore different areas of the objective space. Besides, hybridizing the proposed GSH with other decompositional approaches, such as MOGLS, NSGA-II/MOGLS etc., and studying their behavior in the objective space as well as their performance compared to MOEA/D-GSH is also at the top of our list. Moreover, the DPAPs in WSNs include many features and issues (e.g. interference, delay), which are also important as those in the proposed d-DPAP. Thus, various multi-objective DPAPs can be defined and tackled by problem-specific MOEA/Ds, similarly to this work. Moreover, the underlying idea behind the proposed problem-specific heuristic might also shed some light on the design of MOEA/Ds for other MOPs from different disciplines.

[1] http://mms.geomatics.ucalgary.ca
[2] http://firewatch.cs.ucy.ac.cy
[3] http://www.koetterfire.com

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Communications Magazine (2002) 102–114.
[2] R. Tharmarasa, T. Kirubarajan, J. Peng, T. Lang, Optimization-based dynamic sensor management for distributed multitarget tracking, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 39 (5) (2009) 534–546.
[3] K. Chakrabarty, S.S. Iyengar, H. Qi, E. Cho, Grid coverage for surveillance and target location in distribuuted sensor networks, IEEE Transactions on Computers 51 (12) (2002) 1448–1453.
[4] Q. Wu, N.S.V. Rao, X. Du, S. Sitharama, V.K. Vaishnavi, On efficient deployment of sensors on planar grid, Elsevier Computer Communications 30 (2007) 2721–2734.
[5] P. Cheng, C.N. Chuah, X. Liu, Energy aware node placement in wireless sensor networks, IEEE Global Telecommunications Conference 5 (2004) 3210–3214.
[6] X. Liu, P. Mohaparta, On the deployment of wireless data back-haul networks, IEEE Transactions on Wireless Communication 6 (4) (2007) 1426–1435.
[7] A. Konstantinidis, K. Yang, Q. Zhang, D. Zeinalipour-Yazti, A multi-objective evolutionary algorithm for the deployment and power assignment problem in wireless sensor networks, New Network Paradigms, Elsevier Computer Networks 54 (2010) 960–976.
[8] S. Meguerdichian, F. Koushanfar, M. Potkonjak, M.B. Srivastava, Coverage problems in wireless ad-hoc sensor networks, IEEE INFOCOM 3 (2001) 1380–1387.

[9] M. Cardei, J. Wu, Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems, Chapter: Coverage in Wireless Sensor Networks, CRC Press, 2005, pp. 19-1–19-12.
[10] Y. Chen, C.-N. Chuah, Q. Zhao, Network configuration for optimal utilization efficiency of wireless sensor networks, Elsevier Ad Hoc Networks 6 (2008) 92–107.
[11] Q. Xue, A. Ganz, On the lifetime of large scale sensor networks, Elsevier Computer Communications 29 (2006) 502–510.
[12] P. Santi, Topology control in wireless ad hoc and sensor networks, ACM Computing Surveys 37 (2) (2005) 164–194.
[13] A.E.F. Clementi, P. Penna, R. Silvestri, Hardness results for the power range assignmet problem in packet radio networks, in: Proceedings of the International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, Springer-Verlag, 1999, pp. 197–208.
[14] A. Konstantinidis, K. Yang, H.-H. Chen, Q. Zhang, Energy aware topology control for wireless sensor networks using memetic algorithms, Elsevier Computer Communications 30 (14–15) (2007) 2753–2764.
[15] T. Melodia, D. Pompili, I.F. Akyildiz, On the interdependence of distributed topology control and geographical routing in ad hoc and sensor networks, IEEE Journal on Selected Areas in Communications 23 (3) (2005) 520–532.
[16] X. Cheng, B. Narahari, R. Simha, M. Cheng, D. Liu, Strong minimum energy topology in wireless sensor networks: Np-completeness and heuristics, IEEE Transactions on Mobile Computing 2 (3) (2003) 248–256.
[17] D.B. Jourdan, O.L. de Weck, Layout optimization for a wireless sensor network using a multi-objective genetic algorithm, IEEE Semiannual Vehicular Technology 5 (2004) 2466–2470.
[18] S. Chaudhuri, K. Deb, An interactive evolutionary multi-objective optimization and decision making procedure, Applied Soft Computing 10 (2) (2010) 496–511.
[19] K. Deb, An efficient constraint handling method for genetic algorithms, in: Computer Methods in Applied Mechanics and Engineering, 2000, pp. 311–338.
[20] K. Deb, Multi-objective Optimization Using Evolutionary Algorithms, Wiley and Sons, 2002.
[21] R. Rajagopalan, P.K. Varshney, C.K. Mohan, K.G. Mehrotra, Sensor placement for energy efficient target detection in wireless sensor networks: a multi-objective optimization approach, in: Conference on Information Sciences and Systems, Baltimore, Maryland, 2005.
[22] S.C. Oh, C.H. Tan, F.W. Kong, Y.S. Tan, K.H. Ng, G.W. Ng, K. Tai, Multiobjective optimization of sensor network deployment by a genetic algorithm, IEEE Congress on Evolutionary Computation, CEC 2007 (2007) 3917–3921.
[23] J. Jia, J. Chen, G. Chang, Z. Tan, Energy efficient coverage control in wireless sensor networks based on multi-objective genetic algorithm, Computers and Mathematics with Applications 57 (11–12) (2009) 1756–1766.
[24] J. Jia, J. Chen, G. Chang, Y. Wen, J. Song, Multi-objective optimization for coverage control in wireless sensor network with adjustable sensing radius, Computers and Mathematics with Applications 57 (11–12) (2009) 1767–1775.
[25] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.
[26] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G. da Fonseca, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 117–132.
[27] A. Konstantinidis, K. Yang, Q. Zhang, Problem-specific encoding and genetic operation for a Multi-Objective Deployment and Power Assignment Problem in wireless sensor networks, in: International Conference on Communications, ICC09, 2009.
[28] S. Salcedo-Sanz, J.A. Portilla-Figueras, E.G. Ortiz-Garcia, A.M. Pirez-Bellido, C. Thraves, A. Fernandez-Anta, X. Yao, Optimal switch location in mobile communication networks using hybrid genetic algorithms, Applied Soft Computing 8 (4) (2008) 1486–1497.
[29] P. Moscato, C. Cotta, A Gentle Introduction to Memetic Algorithms, vol. 57, Springer, 2003, pp. 105–144 (Chapter 5).
[30] K. Tang, Y. Mei, X. Yao, Memetic algorithm with extended neighborhood search for capacitated arc routing problems, IEEE Transactions on Evolutionary Computation 13 (5) (2009) 1151–1166.
[31] N. Weicker, G. Szabo, K. Weicker, P. Widmayer, Evolutionary multiobjective optimization for base station transmitter placement with frequency assignment, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 189–203.
[32] Q. Zhang, H. Li, MOEA/D: a multi-objective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 712–731.
[33] R. Tharmarasa, T. Kirubarajan, M. Hernandez, Large-scale optimal sensor array management for multitarget tracking, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 37 (5) (2007) 803–814.
[34] A. Konstantinidis, Q. Zhang, K. Yang, A subproblem-dependent heuristic in MOEA/D for the deployment and power assignment problem in wireless sensor networks, in: IEEE Congress on Evolutionary Computation, CEC09, 2009.
[35] A. Konstantinidis, K. Yang, Multi-objective k-connected deployment and power assignment in wsns using a problem-specific constrained evolutionary algorithm based on decomposition, Computer Communications 34 (2011) 83–98.
[36] A. Konstantinidis, C. Charalambous, A. Zhou, Q. Zhang, Multi-objective mobile agent-based sensor network routing using MOEA/D, in: IEEE Congress on Evolutionary Computation, CEC10, 2010.
[37] A.-S.K. Pathan, C.S. Hong, Serp: secure energy-efficient routing protocol for densely deployed wireless sensor networks, Annales des Télécommunications 63 (9–10) (2008) 529–541.
[38] S. Toumpis, L. Tassiulas, Packetostatics: deployement of massively dense sensor networks as an electrostatic problem, IEEE INFOCOM (2005) 2290–2301.
[39] N. Ahmed, S.S. Kanhere, S. Jha, The holes problem in wireless sensor networks: a survey, SIGMOBILE Mobile Computing Communication Reviews 9 (2) (2005) 4–18, http://doi.acm.org/10.1145/1072989.1072992.
[40] H. Ishibuchi, T. Murata, Multi-objective genetic local search algorithm and its application to flowshop scheduling, IEEE Transactions on Systems, Man and Cybernetics 28 (3) (1998) 392–403.
[41] J.D. Knowles, D. Corne, M-PAES: a memetic algorithm for multiobjective optimization, in: CEC'00 Congress on Evolutionary Computation, IEEE Press, Piscataway, NJ, 2000, pp. 103–108.
[42] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength pareto evolutionary algorithm for multiobjective optimization, in: K.C. Giannakoglou, D.T. Tsahalis, J. Périaux, K.D. Papailiou, T. Fogarty (Eds.), Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems, Athens, Greece, 2002, pp. 95–100.
[43] P.-C. Chang, S.-H. Chen, The development of a sub-population genetic algorithm ii (spga ii) for multi-objective combinatorial problems, Applied Soft Computing 9 (1) (2009) 173–181.
[44] W. Ye, J. Heidemann, D. Estrin, An energy-efficient mac protocol for wireless sensor networks, in: INFOCOM, 2002, pp. 1567–1576.
[45] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, C. Gill, Integrated coverage and connectivity configuration in wireless sensor networks, in: Proceedings of the 1st international conference on Embedded networked sensor systems, 2003, pp. 28–39.
[46] Y. Jin, T. Okabe, B. Sendhoff, Adapting weighted aggregation for multiobjective evolution strategies, in: Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization, EMO'01, Springer-Verlag, London, UK, 2001, pp. 96–110.
[47] C.-Y. Chang, K.-P. Shih, H.-R. Chang, H.-J. Liu, Energy-balanced deployment and topology control for wireless sensor networks, IEEE Global Telecommunications Conference 19 (2006) 1–5.
[48] E. Zitzler, L. Thiele, Multi-objective evolutionary algorithms: a comparative study and the strength Pareto approach, IEEE Transactions on Evolutionary Computation 3 (4) (1999) 257–271.
[49] H. Ishibuchi, T. Yoshida, T. Murata, Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 204–223.
[50] K. Kamyoung, M.T. Alan, N. Xiao, A multiobjective evolutionary algorithm for surveillance sensor placement, Environment and Planning B: Planning and Design 35 (2008) 935–948.
[51] H. Ishibuchi, T. Yoshida, Hybrid evolutionary multi-objective optimization algorithms, in: Design, Management and Applications, Frontiers in Artificial Intelligence and Applications, IOS Press, 2002, pp. 163–172.
[52] D.C. Montgomery, Design and Analysis of Experiments, John Wiley & Sons, 2001.
[53] G. Molina, E. Alba, Location discovery in wireless sensor networks using metaheuristics, Applied Soft Computing 11 (1) (2011) 1223–1240.
[54] M. Cardei, M.O. Pervaiz, I. Cardei, Energy-efficient range assignment in heterogeneous wireless sensor networks, in: International Conference on Wireless and Mobile Communications, 2006.
[55] T. Schurmann, Bias analysis in entropy estimation, Journal of Physics A: Mathematical and General 37 (2004) 295–301.