Multiobjective K-connected Deployment and Power Assignment in WSNs using constraint handling

Andreas Konstantinidis*, Kun Yang*, Qingfu Zhang* and Fernando Gordejuela-Sanchez†
**School of Computer Science and Electronic Engineering*, University of Essex,
{akonst, kunyang, qzhang}@essex.ac.uk
†*Centre for Wireless Network Design,* University of Bedfordshire, fernando.sanchez@beds.ac.uk

***Abstract*— The K-connected Deployment and Power Assignment Problem (DPAP) in WSNs aims at deciding both the sensor locations and transmit power levels, for maximizing both the network coverage and lifetime under K-connectivity constraints, in a single run. Recently, it is shown that the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) is a strong enough tool for dealing with unconstraint real life problems (such as DPAP), emphasizing the importance of incorporating problem specific knowledge for increasing its efficiency. Since the K-connected DPAP requires constraint handling, several techniques are investigated and compared, including a DPAP-specific Repair Heuristic (RH) that transforms an infeasible network design into a feasible one and maintains the MOEA/D's efficiency simultaneously. This is achieved by alternating between two repair strategies, which favor one objective each. Simulation results have shown that the MOEA/D-RH performs better than the popular constrained NSGA-II in several network instances.**

I. Introduction

The Deployment and Power Assignment Problem [1], [2], [3] (DPAP) in WSNs [4] aims at deciding both optimal sensor (a) locations (deployment [5]) and (b) transmit power levels (power assignment [6]) for maximizing the (i) coverage and (ii) lifetime objectives in a single run. DPAP is typical for applications which invoke a limited number of expensive sensors, where their operation is affected by their position and communication. In that case, the application might favor the use of a centralized or even an offline algorithm to compute the decision variables, prior deployment. In WSNs, connectivity is crucial for most applications [7], [8], since a possible partition of the network into disjoint parts may cause undesirable effects, such as decreasing the coverage and consequently the amount of information forwarded to the interested users. A natural generalization of connectivity is K-connectivity or K-fault tolerance [9], [10]. Fault tolerance is a central challenge in WSN design, since the failure of the battery constrained sensors is very common in most applications. A WSN design is usually self-healing when each sensor sustains K-1 faulty neighbors (i.e. K-fault tolerant WSN design). However, most studies [7], [8], [9], [10] focus at deciding either (a) or (b), for maximizing (i) or (ii) individually, or by constraining one and optimizing the other while maintaining connectivity (i.e. K=1) and/or designing K-fault tolerant WSNs. This often results in ignoring and losing "better" solutions, since the WSN coverage and lifetime are conflicting objectives and warrant a trade-off. Moreover, the two decision variables highly influence both objectives and constraints, and should be optimized simultaneously [1], [2]. Thus, we have considered it important and challenging to investigate the multiobjective K-connected DPAP in WSNs.

In [1], [2], [3], we have demonstrated that the Multi-Objective Evolutionary Algorithm based on Decomposition [11] (MOEA/D) is a strong tool to tackle unconstraint real-world problems (e.g. DPAP) emphasizing the importance of incorporating WSN knowledge for increasing its efficiency. However, the addition of constraints in DPAP necessitates constraint handling and render the tailoring of the existing DPAP-specific MOEA/D, to match the abundance of the constraints and objectives of the K-connected DPAP for WSN design in its full practical complexity, as a major challenge. MOEAs with constraint handling focus at obtaining a set of feasible Pareto optimal solutions, i.e. Pareto Front (PF) [12], providing the trade-off between two or more conflicting objectives. Feasible are the solutions that satisfy all constraints, and infeasible are those that do not. In the literature, there are several constraint handling techniques [13], including the use of a Penalty Function (PenF), adopting the rules of the Superiority of Feasible solutions (SoFs) and using a Repair Heuristic (RH) [14]. For the latter, Coello [13] has effectually declared that "*any heuristic which would guide the repair process, and the success of this approach relies mainly on the ability of the user to come up with such a heuristic*." Hence, RH is a good choice when an infeasible solution can be easily transformed to a feasible one without harming the optimization process and is carefully designed with problem domain knowledge.

In this paper, the K-connected DPAP in WSNs is defined and formulated as a constrained Multiobjective Optimization Problem (MOP). Then, several constraint handling techniques are designed and/or adopted by MOEA/D for tackling the proposed MOP. Namely, 1) a PenF is defined for DPAP and used for directing the search into the feasible regions of the search space by penalizing the infeasible solutions. 2) The proposed DPAP-specific RH is designed for transforming an infeasible solution into a feasible one and maintaining MOEA/D's efficiency simultaneously, by alternating between two repair strategies, which favor one objective each. 3) The rules of the SoFs are adopted, initially proposed by Deb for handling constraints with the popular Non-dominated Sorting Genetic Algorithm II (NSGA-II) [12], which favor feasible solutions over infeasible. Finally, MOEA/D and NSGA-II are compared in several constrained DPAP network instances.

II. Problem Definition

A. System model and assumptions

Consider a 2-D static WSN formed by: a rectangular sensing area A, N homogeneous sensors and a static sink H with unlimited energy, placed at the center of A. We assume a perfect medium access control and we adopt the simple but relevant path loss communication model [2]. 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 $d_{i,j}$ 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 assigned P_i and the locations $L_i = (x_i, y_i)$ are the DPAP's decision variables and are considered fixed for the whole network lifetime, where (x_i, y_i) are sensor i's coordinates and $i = 1, \ldots, N$. The residual energy of sensor i, at time t, is calculated as follows:

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

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 $x \times y$ uniform consecutive grids to make the coverage problem more computationally manageable. 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 [7],

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

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

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

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

where sensor j is usually considered K-connected [9], if it directly communicates with H or if it sustains K neighbors with positive advance [15] towards H, considering the many-to-one communication nature of WSNs.

B. Problem formulation

The K-connected DPAP in WSNs can be formulated as a constrained MOP,

Given:

- A: 2-D plane of area size $x \times y$.
- N: 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:

- L_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** K-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:

$$Cv(X) = [\sum_{x'=0}^{x} \sum_{y'=0}^{y} g(x', y')]/(x \times y) \tag{4}$$

where, $x \times y$ is the total grids of A and $g(x', y')$ is calculated using Equation (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 follows:

Algorithm: Lifetime Evaluation

```
***Step 0:*** Set $t := 1$; $E_j(0) := E, \forall j \in \{1, ..., N\}$;
***Step 1:*** **For** all sensors $j$ at each time interval $t$ **do**
    ***Step 1.1:***Update $E_j(t)$ according to Equation (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** $\exists\, j \in \{1, ..., N\}$ such that $E_j(t) = 0$ **then** stop
    and set:
        $L(X) := t;$        (5)
    **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. [2]) and routing algorithms in Step 1.3 (e.g. geographical-based [15] routing algorithms).

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

$$Cn(X) = |CS|/N \tag{6}$$

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

III. Constraint Handling Techniques: The proposed Repair Heuristic (RH) in MOEA/D

A. MOEA/D [11]: an overview

The MOP can be decomposed into m subproblems using any technique that constructs aggregation functions, e.g. the

Weighted Sum Approach [11]. Then, a subproblem i with a weight coefficient λ^i can be defined as:

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

The dense-to-spread encoding representation is adopted [2]. The Internal Population, IP (stores the best solutions found for each subproblem i during the search) is randomly initialized. A new solution O is generated by the genetic operators (e.g. [2]) and a local heuristic (e.g. [3]) is applied to each O to produce X. In the update phase [11], the IP, the neighborhood of X (i.e. the solutions of the T closest subproblems of i in terms of their weight coefficients $\{\lambda^1, \cdots, \lambda^m\}$) and the external population (EP) (stores all the non-dominated solutions found so far during the search) are updated with X. The search stops after a predefined number of generations, gen_{max}. MOEA/D proceeds as in Algorithm 1.

Algorithm 1 MOEA/D framework

```
Input:
• network parameters ($A$, $N$, $E$, $R_s$);
• $m$ : population size and number of subproblems;
• $T$: neighborhood size;
• uniform spread of weight coefficients $\lambda^1, ..., \lambda^m$;
• the maximum number of generations, $gen_{max}$;
Output: the external population, $EP$.

Step 0-Setup: Set $EP := \emptyset$; $gen := 0$; $IP_{gen} := \emptyset$;
Step 1-Decomposition: Initialize $m$ subproblems.
Step 2-Initialization: Randomly generate an initial internal
population $IP_0 = \{Y^1, \cdots, Y^m\}$;
Step 3: For each subproblem $i = 1$ to $m$ do
    Step 3.1-Genetic Operators [2]: Generate a new solution $O$ by using the weight-based selection, the window
    crossover and the adaptive mutation operators.
    Step 3.2-Local heuristics: Apply an improvement [3]
    and/or repair heuristic to $O$ to produce $X$.
    Step 3.3-Update Populations: Update $IP_{gen}$, $EP$ and the
    $T$ closest neighbors of subproblem $i$ with $X$.
Step 4-Stopping criterion: If stopping criterion is satisfied, i.e.
$gen = gen_{max}$, then stop and output $EP$, otherwise $gen =$
$gen + 1$, go to Step 3.
```

A major advantage of MOEA/D, compared to other MOEAs, is that each solution in the population is associated with a scalar subproblem. Thus, in [1], [2], [3], we have shown that MOEA/D can easily adopt different single objective methods for optimizing each scalar subproblem i, accordingly. This is achieved by designing problem specific genetic operators/heuristics rising by each subproblem i's objective preference (i.e λ^i) and requirements. The λ^i parameter is used as a guide to the operators/heuristics for adjusting the degree of network coverage and lifetime. In this paper, the main focus is at constraint handling and specifically at the repair heuristic, which aims at designing feasible and high quality WSN topologies of different objective preferences, at the same time.

B. Constraint Handling Techniques

The following constraint handling techniques were designed and/or adopted for tackling the K-connected DPAP,

1) Penalty Function [13], PenF: transforms a constrained MOP into an unconstrained one by subtracting a certain value (known as penalty measured by a penalty function) from the scalar fitness value, based on the amount of constraint violation. It aims at favoring the feasible solutions over infeasible ones during the selection process. The amount of violation is measured based on the percentage of sensors in the network that violate the K-connectivity constraint. The penalty of a solution X is measured as follows:

$$pn(X) = 1 - Cn(X),$$

where $Cn(X)$ is calculated using Equation (6).

A constrained subproblem i can then be transformed into an unconstrained one as follows:

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

2) The superiority of feasible solutions [12], SoFs: A comparison between two solutions X and Y of a subproblem i is performed based on the following rules:

- If X is feasible and Y is not feasible then select X.
- If both X and Y are feasible then select the one with the highest scalar fitness.
- If both X and Y are infeasible then select the one with the least constraint violation, i.e. the least number of sensors that violate the K-connectivity constraint.

It aims at favoring the good feasible, or least infeasible solutions to be copied in the next generation.

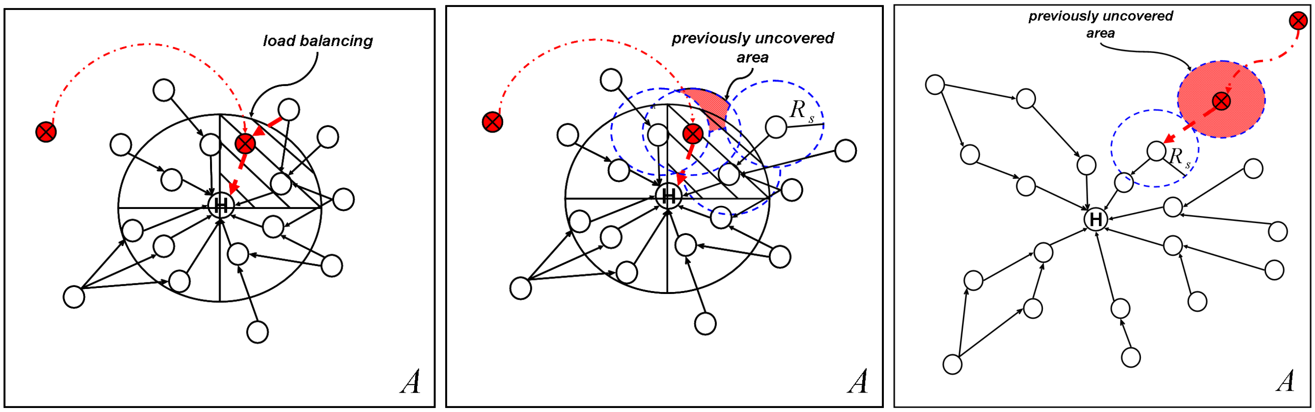
3) DPAP-specific Repair Heuristic, RH: aims at transforming an infeasible solution X to a feasible solution Z in such a way that:

- the feasible solution is similar to the infeasible to support the exploration behavior of the MOEA/D.
- the origin of infeasibility is used to support the exploitation behavior of the MOEA/D.

Algorithm 2 The DPAP-specific Repair Heuristic (RH)

```
Input: A solution $X$;
Output: A feasible solution $Z$;
Step 0: Set $K$; $s$;
Step 1: if $Cn(X) = \begin{cases} 1 & \text{goto Step 2;} \\ 0 & \text{goto Step 5;} \end{cases}$
Step 2: Find the origin of infeasibility, e.g. sensor $j$;
Step 3:
    if $\lambda^i$
        $\geq 0.5$
            Step 3.1:Divide the circle $(r = R_{max})$
            centered at $L_H$ into $s$ equal sectors;
            Step 3.2:Find the sparsest sector;
            Step 3.3:Uniformly randomly generate a new
            location $L'_j$ in the sparsest sector, set $P_j = (d^{\alpha}_{jH})$;
        $< 0.5$
            Step 3.4:Find the $K^{th}$ closest location to $L_j$,
            e.g. $L_v \in X$;
            Step 3.5:Calculate a new location $L'_j$ using Eq. 7,
            set $P_j = (R^j_c)^{\alpha}$;
Step 4: If $\exists j|(x_j, y_j) \in X, c_j \neq 1$ then goto Step 2;
Step 5: Output $Z = X$;
```

To achieve this, the λ^i weight coefficient of each subproblem i is used as a guide to the RH for specifically repairing



(a) Repairing an infeasible solution with low λ^i, increasing lifetime.

(b) Repairing an infeasible solution with low λ^i, increasing coverage without affecting lifetime.

(c) Repairing an infeasible solution with high λ^i, increasing coverage.

Fig. 1. Examples of repairing an infeasible solution for subproblems with low and high λ^i coefficients.

an infeasible solution X based on the foresaid remarks and its objective preference. The RH (outlined in Algorithm 2), has the following characteristics:

- When λ^i is high, and subproblem i focuses at feasible solutions with high network lifetime, the RH
  1) divides the circle with radius $r = R_{max}$ centered at L_H into s equal sectors (e.g. $s = 4$).
  2) finds the sparsest sector (i.e. the sector with the lowest number of sensors).
  3) redeploys the origin of infeasibility, e.g. j at $L_j \in X$, to a random location L'_j within the sparsest sector, such that $R_c^j = d_{jH} \leq R_{max}$ and sets $P_j = (R_c^j)^\alpha$.

  In this case, while the RH is repairing an infeasible network design it might also provide the following benefits:
  - supports the network load balancing and prevents a premature energy exhaustion of the sensors that are already directly connected to H, increasing the network lifetime (Figure 1(a), for $K = 1$).
  - covers any previously uncovered area close to H, increasing the network coverage without decreasing the network lifetime (Figure 1(b), for $K = 1$).
- When λ^i is low, subproblem i focuses at feasible solutions with high network coverage. The RH,
  1) finds a sensor v, which is the K^{th} closest positive-advance neighbor of sensor j.
  2) Then, j is redeployed to a new L'_j as follows,

$$L'_j = L_j + (d_{ju} - R_c^j) \times (L_u - L_j)/d_{ju} \qquad (7)$$

$$\text{where, } u = \begin{cases} v & if \;\; c_j = 1, d_{jv} \leq d_{jH} \\ H & otherwise \end{cases},$$

$$R_c^j = \begin{cases} 2R_s & \text{if } R_{max} \geq 2R_s \\ R_{max} & otherwise \end{cases}$$

  3) Sets $P_j = (R_c^j)^\alpha$.

This results in low sensing range overlaps between the sensors that might increase the network coverage while repairing the infeasible solution (Figure 1(c), for $K = 1$). If there does not exist a L'_j that satisfies $c_j = 1$ and $d_{ju} < d_{jH}$, then j is directly connected to H to repair the infeasibility.

IV. Simulation Results and Discussion

The goals of our simulation studies are: 1) to demonstrate the difficulty in obtaining feasible solutions for the K-connected DPAP through a purely random process, 2) to test the strength of the three constraint handling techniques with MOEA/D at dealing with the K-connected DPAP and to show the superiority of the proposed repair heuristic (RH) that incorporates DPAP-specific knowledge. 3) To demonstrate the effectiveness of the proposed DPAP-specific MOEA/D-RH against the popular constrained NSGA-II (using the SoFs) in various WSN instances, giving the trade-off of the objectives and a variety of feasible network design choices.

TABLE I
Network Instances (NIn)

NIn	A (m^2)	N	Density (N/A)
1	2500(50 × 50)	25	0.01 (25/2500)
2	2500(50 × 50)	50	0.02 (50/2500)
3	2500(50 × 50)	63	0.025 (63/2500)
4	10000(100 × 100)	100	0.01 (100/10000)
5	10000(100 × 100)	150	0.02 (150/10000)
6	10000(100 × 100)	250	0.025 (250/10000)

Table I shows several network instances. In our simulation studies we have investigated sixteen network instances, however, only six are presented here due to the page limit. In all simulations, we have used the following network settings: $a = 2$, $R_s = 5m$, $R_{max} = 10m$, $amp = 100pJ/bit/m^2$, $E_s = E_{ct} = 50nJ/b$, $k = 250bytes$, $E = 5J$ and square grids with $1m$ side length. Moreover, we have used the following algorithm settings: $m = 120$, crossover rate $r_c = 1$, mutation rate $r_m = 0.1$, tournament size $s_t = 10$ and $gen_{max} = 250$, $T = 2$ as in [1], [2], [3]. In all cases, lifetime is normalized with the upper bound (L_{max}) defined in [3].

Initially, to get an estimate of how difficult is to generate feasible network designs in the proposed K-connected DPAP

TABLE II

SIMULATION RESULTS ON $S = 30000$ RANDOM NETWORK DESIGNS

NIn	K	Infeasible Sol.	ϱ	Disconnected Sensors Total	Disconnected Sensors Average
1	1	29235.0	0.0255	275337.0	9.175
	2	30000.0	0.0	511665.0	17.05
	3	30000.0	0.0	649890.0	21.675
	4	30000.0	0.0	717879.0	23.925
	5	30000.0	0.0	741642.0	24.725
2	1	12870.0	0.571	16766.0	1.7
	2	30000.0	0.0	110425.0	11.05
	3	30000.0	0.0	214441.0	21.45
	4	30000.0	0.0	321518.0	32.15
	5	30000.0	0.0	405060.0	40.5
3	1	6030.0	0.799	5412.0	0.567
	2	30000.0	0.0	82646.0	8.253
	3	30000.0	0.0	174914.0	17.514
	4	30000.0	0.0	293468.0	29.358
	5	30000.0	0.0	411594.0	41.139

TABLE III

SIMULATION RESULTS OF MOEA/D W/RH, W/SOFS AND W/PENF

NIn	K	Infeasible Sol.	ϱ	Disconnected Sensors Total	Disconnected Sensors Average
1	MOEA/D w/RH				
	1	27751.0	0.075	153857.0	5.12
	2	19665.0	0.345	52634.0	1.754
	3	20106.0	0.330	60080.0	2.0
	4	20448.0	0.318	70083.0	2.33
	5	21898.0	0.270	80357.0	2.67
	MOEA/D w/SoFs				
	1	27941.0	0.069	197800.0	6.59
	2	30000.0	0.0	313096.0	10.43
	3	30000.0	0.0	458423.0	15.28
	4	30000.0	0.0	558941.0	18.63
	5	30000.0	0.0	625390.0	20.84
	MOEA/D w/PenF				
	1	27892.0	0.070	196318.0	6.54
	2	30000.0	0.0	159560.0	5.31
	3	30000.0	0.0	217785.0	7.25
	4	30000.0	0.0	242473.0	8.08
	5	30000.0	0.0	289071.0	9.63

through a purely random process, we have measured (i) the $\varrho = |F|/|S|$ metric, where $|F|$ is the number of feasible solutions and $|S|$ is the total number of solutions generated, (ii) the total number of infeasible solutions, (iii) the total and (iv) the average number of disconnected sensors of 30000 random trials in NIn1,2 and 3 with $K \in \{1, \ldots, 5\}$. Note that, disconnected sensors are those that are not K-connected.

The results of Table II show that the random process obtains feasible solutions only when $K = 1$. For $K = 2$ to 5, all 30000 network designs are infeasible in all network instances (i.e. NIn1-3). Moreover, when $K = 1$ and the density is low (i.e. NIn1, $N = 25$), there are only 2.55% feasible solutions, which means 29235 out of 30000 network designs are infeasible, having about $9/25$ sensors disconnected per network design (i.e. about 36%). When the density is high (e.g. NIn3, $N = 63$) this number decreases to about 0.99% (i.e. $0.567/63$ sensors, on average). This is the reason why, sometimes it is assumed [16] that a dense sensor deployment implies network connectivity. Table II, however, shows that even when the number of disconnected sensors is low, the $\varrho = 79.9\%$ indicates that a relatively high number of solutions is still infeasible, i.e. about 20.1% or $6030/30000$ solutions.

Thereinafter, the three constraint handling techniques were tested in NIn1 for $K \in \{1, \ldots, 5\}$ and $S = m \times gen_{max} = 30000$ in terms of the (i-iv) metrics, which are evaluated at the beginning of each generation for PenF and SoFs, and before repairing for RH. The results of Table III show that MOEA/D w/RH helps the evolutionary process to obtain feasible solutions for all Ks. In contrast, MOEA/D w/SoFs and w/PenH obtain infeasible solutions only when $K = 2$ to 5 (i.e. 30000 infeasible solutions and $\varrho = 0.0$). Besides, the number of disconnected sensors obtained by MOEA/D w/RH is lower than those obtained w/SoFs and w/PenH for all Ks.

The hybridization of MOEA/D with each constraint handling technique is compared in NIn1-3 for $K = 1$, in terms of the following performance metrics: $C(A, B)$ measures the solutions in an algorithm A's PF dominated by the solutions in an algorithm B's PF, the smaller $C(A, B)$ is the better algorithm A is. $\Delta(A)$ shows the diversity of the PF obtained by algorithm A, i.e. the spread/variety of the network design choices. $\Delta = 0$ is the maximum, which means that the solutions are evenly spread along the PF. $NDS(A)$ is the total number of non-dominated solutions obtained by algorithm A, the higher the NDS is the better algorithm A is. Finally, $CPU(A)$ measures the total computational effort of A.

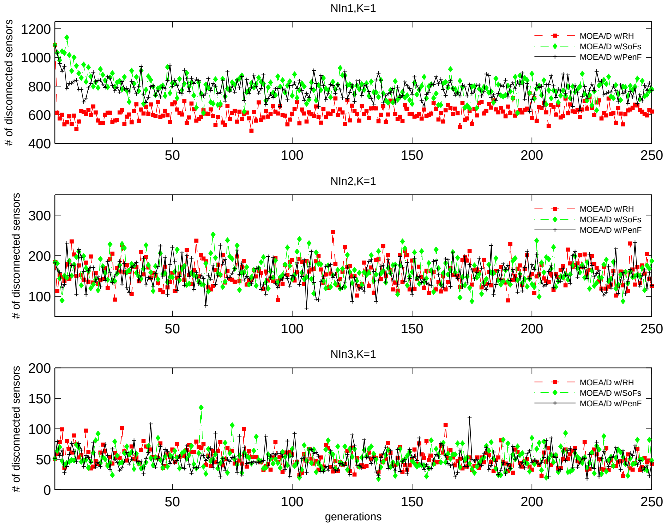


Fig. 2. The number of disconnected sensor per generation obtained by MOEA/D with RH, SoFs and PenF in NIn1-3, K=1.

Figure 2 examines the total number of disconnected sensors (ds) obtained by MOEA/D with each technique. In NIn1, all techniques begin with about 1100 ds. The latter is sharply decreased to about 600 ds after one generation when RH is adopted and is smoothly decreased to about 800 ds after about 20 generations when PenF and SoFs are adopted. This indicates that RH directs the search into the feasible regions of the search space more effectively. When the network becomes denser (NIn2,3) the number of ds decreases and the three techniques perform similarly. Nevertheless, the statistical results, summarized in Table IV, show that RH is more beneficial for MOEA/D's performance than PenF and SoFs. MOEA/D w/RH provides a better average Δ metric and about 0.7 more NDS. In terms of quality, the NDS obtained by MOEA/D w/RH

TABLE IV

SIMULATION RESULTS FOR MOEA/D WITH RH, SOFS AND PENF, $K = 1$

	Δ			NDS			CPU (hrs)			C			
NIn	RH	SoFs	PenF	RH	SoFs	PenF	RH	SoFs	PenF	RH,SoFs	SoFs,RH	RH,PenF	PenF,RH
1:	**0.9075**	0.9701	0.9627	**11**	7	7	0.1296	0.0976	**0.0815**	**0.0**	0.8571	**0.0**	0.7143
2:	0.9776	**0.9642**	0.9797	7	**8**	5	0.2611	0.1514	**0.1137**	**0.3333**	0.7143	0.5	**0.4**
3:	0.9799	0.9787	**0.9763**	**7**	**7**	5	0.3194	0.2001	**0.1817**	**0.0**	1.0	**0.0**	0.6
Av.:	**0.9550**	0.9710	0.9729	**8.3333**	7.3333	5.6667	0.2367	0.1497	**0.1257**	**0.1111**	0.8571	**0.1667**	0.5714

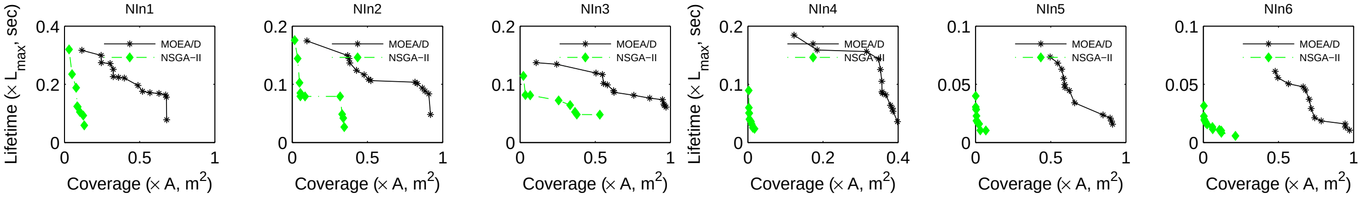


Fig. 3. Comparison of MOEA/D and NSGA-II in NIn1-6, K=1

dominate 85% and 57%, on average, of the NDS obtained by MOEA/D w/SoFs and MOEA/D w/PenF, respectively. Its superiority is at the cost of a slightly higher CPU effort.

TABLE V

SIMULATION RESULTS FOR MOEA/D (MD) AND NSGA-II (NG)

	Δ		NDS		CPU (hrs)		C	
NIn	Md	Ng	Md	Ng	Md	Ng	Md,Ng	Ng,Md
1:	**0.854**	0.943	**15**	8	**0.17**	0.2	**0**	1
2:	**0.873**	0.919	**14**	10	**0.26**	0.31	**0**	1
3:	**0.841**	0.899	**14**	8	**0.33**	0.39	**0**	1
4:	**0.928**	0.989	**13**	10	**0.88**	1.22	**0**	1
5:	**0.906**	0.984	12	12	**1.59**	1.71	**0**	1
6:	**0.891**	0.959	12	**13**	**2.94**	2.98	**0**	1
Av.:	**0.882**	0.948	**13.3**	10.1	**1.02**	1.135	**0**	1

Finally, MOEA/D w/RH is compared with NSGA-II in NIn1-6. NSGA-II adopts the SoFs as proposed by [12] and the same genetic operators as in [2], i.e. a tournament selection, a two-point crossover and a random mutation operator. The algorithm settings and the function evaluations are the same in both MOEAs for fairness. Table V and Figure 3 show the superiority of MOEA/D in all instances. Specifically, MOEA/D obtains a more diverse PF in less CPU time, which has three additional NDS, on average, and dominates all NDS of the PF obtained by NSGA-II and none is dominated.

V. CONCLUSIONS

A K-connected DPAP in WSNs is defined and formulated as a constrained MOP and a MOEA/D is specialized. Several constraint handling techniques are designed and/or adopted by MOEA/D, including a DPAP-specific repair heuristic that incorporates WSN-knowledge for transforming an infeasible solution to a feasible one. The repair heuristic alternates between two repair strategies, each favoring one objective, for increasing the MOEA/D's performance at the same time. Simulation results have shown the difficulty in randomly obtaining feasible solutions for the proposed K-connected DPAP, the necessity of incorporating WSN-knowledge while handling constraints and the superiority of the DPAP-specific MOEA/D against the popular NSGA-II in several WSN instances.

REFERENCES

[1] A. Konstantinidis, K. Yang, and Q. Zhang, "An evolutionary algorithm to a Multi-Objective Deployment and Power Assignment Problem in wireless sensor networks," in *IEEE GlobeCom08*, vol. AH16, no. 1–3, 2008, pp. 475–481.

[2] ——, "Problem-specific encoding and genetic operation for a Multi-Objective Deployment and Power Assignment Problem in wireless sensor networks," in *IEEE ICC09*, 2009.

[3] A. Konstantinidis, Q. Zhang, and K. Yang, "A subproblem-dependent heuristic in MOEA/D for the deployment and power assignment problem in wireless sensor networks," in *IEEE CEC09*, 2009.

[4] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Comms Magazine*, pp. 102–114, 2002.

[5] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Elsevier Ad Hoc Networks*, vol. 6, no. 4, pp. 621–655, 2007.

[6] P. Santi, "Topology control in wireless ad hoc and sensor networks," *ACM Computing Surveys*, vol. 37, no. 2, pp. 164–194, 2005.

[7] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration in wireless sensor networks," in *Proc. of Embedded networked sensor systems*, 2003, pp. 28–39.

[8] W. Mo, D. Qiao, and Z. Wang, *Distributed Computing in Sensor Systems*. Springer, June 11 2006, ch. Lifetime Maximization of Sensor Networks Under Connectivity and k-Coverage Constraints, pp. 422–442.

[9] M. Hajiaghayi, N. Immorlica, and V. S. Mirrokni, "Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks," in *MobiCom '03*. ACM, 2003, pp. 300–312.

[10] X.-Y. Li, P.-J. Wan, Y. Wang, and C.-W. Yi, "Fault tolerant deployment and topology control in wireless networks," in *Proc. of MobiHoc03*. ACM Press, 2003, pp. 117–128.

[11] Q. Zhang and H. Li, "MOEA/D: A multi-objective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[13] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11-12, pp. 1245 – 1287, 2002.

[14] A. Konstantinidis, K. Yang, H.-H. Chen, and Q. Zhang, "Energy aware topology control for wireless sensor networks using memetic algorithms," *Computer Comm.*, vol. 30, no. 14-15, pp. 2753–2764, 2007.

[15] T. Melodia, D. Pompili, and 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*, vol. 23, no. 3, pp. 520–532, 2005.

[16] S. Toumpis and L. Tassiulas, "Optimal deployment of large wireless sensor networks," *IEEE Transactions on Information Theory*, vol. 52, no. 7, pp. 2935–2953, 2006.