An Effective Metaheuristic Approach to Node Localization in Wireless Sensor Networks

Massimo Vecchio, Roberto López–Valcarce
Departamento de Teoría de la Señal y las Comunicaciones,
University of Vigo,
C/ Maxwell s/n, 36310 Vigo - SPAIN
Email: {massimo,valcarce}@gts.uvigo.es

Francesco Marcelloni
Dipartimento di Ingegneria dell'Informazione,
University of Pisa,
Largo Lucio Lazzarino 1, 56122 Pisa - ITALY
Email: f.marcelloni@iet.unipi.it

***Abstract*—Given a small percentage of nodes whose actual positions are known, the problem of estimating the locations of the remaining nodes of a wireless sensor network has attracted a large interest in the last years. The localization task is based on the noisy estimates of the distances between pairs of nodes in range of each other. The problem is particularly hard when the network connectivity is not sufficiently high, the most attractive case in real applications. In this paper, we propose to solve the localization problem by using a two–objective evolutionary algorithm which takes concurrently into account during the evolutionary process both the localization accuracy and certain topological constraints induced by the network connectivity. The proposed method is tested with different network configurations and sensor setups, and compared in terms of normalized localization error with two approaches based on semi–definite programming. The results show that, in all the experiments, our approach achieves considerable accuracies, thus manifesting its effectiveness and stability, and outperforms the compared approaches.**

I. Introduction

A Wireless Sensor Network (WSN) may consist of hundreds or even thousands of low–cost nodes communicating among themselves for applications like environment monitoring, precision agriculture, vehicle tracking, logistics, etc. [1]. Generally speaking, tiny nodes are deployed in an area to be monitored, spanning a potentially large geographical region. The small size and low cost of sensor nodes impose several practical limitations: as they mount small and cheap memory and microprocessor units, tasks such as large data storing and complex computations become unfeasible. At the same time, since nodes are usually battery–powered, network lifetime usually constitutes an important issue [2], [3].

In the aforementioned applications, knowledge about the location of sensor nodes may play a key role (we refer the interested reader to [4] and the references therein). Although in principle the use of a Global Positioning System (GPS) could enable such "*location awareness*", this solution is not always viable in practice. The first reason is merely economic, as the cost of GPS receivers is not negligible. The second reason is related to the power consumption of a standard GPS receiver, which is generally not affordable by battery–powered nodes. Last but not least, a technology–related reason arises in indoor and underground WSN deployments: in these situations, in fact, communication with satellites may be compromised.

The work of M. Vecchio and R. López-Valcarce is supported by the Spanish Government and the European Regional Development Fund (ERDF) under projects DYNACS (TEC2010-21245-C02-02/TCM) and COMONSENS (CONSOLIDER-INGENIO 2010 CSD2008-00010).

These limitations have motivated alternative approaches to the problem, as reviewed in [5]–[7]. Among these, *fine–grained* localization techniques arise as a flexible option. In these schemes, only a few nodes of the network (termed *anchor nodes*) are endowed with their exact positions through GPS or manual placement, while all nodes are able to estimate their distances to nearby nodes by using some measurement technique. These distance–related techniques include Received Signal Strength (RSS) measurements, Time of Arrival (ToA), Time Difference of Arrival (TDoA), etc. (for a review of these techniques the reader is referred to [6], [7]). Thus, assuming that the coordinates of anchor nodes are known, and exploiting pairwise distance measurements among the nodes, the fine–grained localization problem is to determine the positions of all non–anchor nodes. This task has proved to be rather difficult, due to the following reasons: *i)* determining the locations of the nodes from a set of pairwise distance estimates is a non-convex optimization problem; *ii)* the measurements available to nodes are invariably corrupted by noise; and finally, *iii)* even if the distance estimates were perfectly accurate, sufficient conditions for the solution to be unique are not easily identified [8]. We will briefly discuss these issues in the following.

Assuming a statistical characterization of measurement noise (which will usually depend on the kind of measurement technique [6]), Maximum Likelihood (ML) estimation is the natural approach to the localization problem. However, as previously mentioned, the ML formulation results in a multivariable nonconvex optimization problem. Three different approaches to this task can be found in the literature, namely stochastic optimization, multidimensional scaling, and convex relaxation. The first approach attempts to avoid local maxima of the likelihood function by resorting to global optimization methods, such as *e.g.* simulated annealing [9]. Multidimensional scaling [10], [11] is a *connectivity–based* technique that, in addition to distance measurements, exploits knowledge about the topology of the network; this information imposes additional constraints on the problem, since nodes within communication range of each other cannot be arbitrarily far apart. The third approach relax the original nonconvex ML formulation in order to obtain a Semi–Definite Programming (SDP) or a Second–Order Cone Programming (SOCP) problems. Global solutions to these relaxed, convex problems can be then obtained with moderate computational effort [12] [13] and constitute approximate solutions to the original nonconvex problem. In [13] it was shown that the solutions obtained by SOCP relaxation are less accurate than those obtained by SDP relaxation. Moreover, in order to improve the computational efficiency of the original SDP approach proposed in [12], further relaxations have been recently proposed. For instance, in [14], [15] examples of such relaxations were presented, which are able to handle much larger–sized problems, at the expense of an increase in localization error with respect to the original SDP approach. Basically, all of these methods aim to relax the original problem at the modeling level, so as to maintain the sparsity of the graph by limiting the amount of selected edges connected to each sensor or anchor node.

In the following we will consider the most accurate convex relaxation approach, *i.e.* the original full–SDP formulation proposed in [12] (we will refer to this method as FSDP). It should be mentioned that FSDP may still incur in significant estimation errors, and that a regularized version (referred to as FSDPr) and a gradient–descent refinement technique have been proposed in [8] and [16], respectively, in order to improve its performance. FSDPr adds a regularization term to the objective function in order to reduce the tendency of SDP solutions to have points crowded together, which occurs after the last step of projecting the high–rank SDP solution back onto the two–dimensional plane. Thus, the regularization term suggested in [8] penalizes small node separations. The main issue in FSDPr is the choice of the regularization value: if it is too low, then the effects of regularization are negligible; on the other hand, if this value is too high, then the regularization term will overwhelm the error term, making the SDP either unfeasible or yielding a solution whose points are too far apart. Although there is no way to obtain a good regularization value *a priori*, in [8] the authors proposed a method for its computation based on the solution provided by FSDP without regularization. Finally, the goal of gradient–based refinements is to improve the final estimation given by a localization algorithm. Since gradient–based methods generally do not deliver a global optimal solution when the problem is not convex, this technique can be applied as a fine–tuning phase once an approximation to the global solution has been found [8], [16]. Thus, the technique can be applied to any localization method.

In this paper we propose to tackle the localization problem by using a Multi–Objective Evolutionary Algorithm (MOEA). In particular, we adopt two objectives: the first objective, denoted *CF*, is given by the original nonconvex cost (the sum of squared differences between the estimated and the corresponding measured inter–node distances). The second objective, denoted *CV*, exploits the connectivity–based *a priori* information about the network, and is especially useful in order to alleviate *localizability* issues.

The proposed approach is tested with a variety of network topologies, percentages of anchor nodes, and connectivity ranges, and compared in terms of normalized localization error with FSDP and FSDPr, without and with a gradient–based refinement stage. It will be shown that the proposed evolutionary method consistently produces more accurate estimates of the sensor locations than convex relaxation–based approaches. The improvement is more significant for network topologies with lower connectivity, for which the localization problem becomes more difficult. Thus, the proposed approach constitutes a good candidate for WSN applications which do not demand highly–scalable or real–time solutions but do require high localization accuracies.

The remainder of this paper is organized as follows. In Section II, we present the problem formulation. Section III introduces our multi–objective evolutionary approach to the problem. The experimental results of our performance analysis are presented in Section IV. Finally, in Section V we draw some conclusions.

II. Problem formulation

The goal of this section is to introduce the system model as well as the objective functions for the evolutionary algorithm and the performance metric. We also discuss certain geometrical constraints which can be defined on each non–anchor node, exploiting the *a priori* information (*i.e.* connectivity) about the network topology.

A. System model

We consider a WSN with n nodes deployed in $T = [0, 1] \times [0, 1] \subset \mathbb{R}^2$. Among these, nodes 1 through m, with $m < n$, are anchor nodes whose coordinates $\mathbf{p}_i = (x_i, y_i) \in \mathbb{R}^2$, $i = 1,\ldots, m$, are known. We assume that if two sensor nodes, say i and j, are within communication range of each other, then their inter–node distance d_{ij} can be estimated by using some measurement technique (see Section I). Distance measurements d_{ij} are modeled as

$$d_{ij} = r_{ij} + e_{ij} \tag{1}$$

where $r_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\|$ is the actual distance between nodes i and j ($\|\cdot\|$ denotes the Euclidean norm). Similarly to [8], [9], [12], [16], we assume that measurement errors e_{ij} follow a zero–mean Gaussian distribution with variance σ^2. It is also assumed that the random variables e_{ij} and e_{kl} are statistically independent for $(i, j) \neq (k, l)$.

A simple disk model is adopted for network connectivity: nodes i and j can communicate with each other if and only if $r_{ij} \leq R$, where R is the connectivity range. This model is commonly used in the literature, although empirical measurements on real WSNs have shown that it is only an approximation in practice. On the other hand, different connectivity models could be adopted by modifying the geometrical analysis in Section II-C. We refer to nodes j such that $r_{ij} \leq R$ as *first–level neighbors* of node i. Further, we refer to all nodes j which are not first–level neighbors of node i, but which share at least a first–level neighbor with node i, as *second–level neighbors* of node i. Let

$$\begin{aligned} N_i &= \{j \in 1 \ldots n, j \neq i : r_{ij} \leq R\} &(2)\\ \overline{N}_i &= \{j \in 1 \ldots n, j \neq i : r_{ij} > R\} &(3)\end{aligned}$$

be the set of the first–level neighbors of node i and its complement, respectively. We assume that sets N_i and $\overline{N}_i$ are known for all $i = 1, \ldots, n$. This is a reasonable assumption, since each node can easily determine which other nodes it can communicate with.

B. Objective functions and performance metric

Pursuing the goal of estimating the positions of the non–anchor nodes as accurately as possible, we propose to concurrently minimize two objective functions. Let $\hat{\mathbf{p}}_i = (\hat{x}_i, \hat{y}_i), i = m+1, \ldots, n$ be the estimated positions of the non–anchor nodes i. The first objective *CF* is defined as

$$CF = \sum_{i=m+1}^{n} \left(\sum_{j \in N_i} \left(\hat{d}_{ij} - d_{ij} \right)^2 \right), \qquad (4)$$

where $\hat{d}_{ij}$ is the estimated distance between nodes i and j computed as follows:

$$\hat{d}_{ij} = \begin{cases} \sqrt{(\hat{x}_i - x_j)^2 + (\hat{y}_i - y_j)^2}, & \text{if node } j \text{ is an anchor,} \\ \sqrt{(\hat{x}_i - \hat{x}_j)^2 + (\hat{y}_i - \hat{y}_j)^2}, & \text{otherwise.} \end{cases} \qquad (5)$$

Thus, CF is the sum of squared differences between the measured data and distances corresponding to the candidate geometry (as given by the estimated positions $\hat{\mathbf{p}}_i$, $i = m+1, \ldots, n$ of the non–anchor nodes and the positions of the anchor nodes).

Given a set of data comprised by the set of anchor nodes and the inter–node distance measurements, a network is said to be localizable if there is only one possible geometry compatible with the data. Localizability is a fundamental problem which can be studied within the framework of *rigid graph theory* [17]. If the network is not localizable, then multiple global minima will be present in the cost function, with only one of them corresponding to the actual geometry of the deployment. Thus, in settings which are close to not being localizable, any localization algorithm will become extremely sensitive to these false minima of *CF*, resulting in very large localization errors [18], [19].

The simplest effect leading to lack of localizability is the so–called *flip ambiguity* phenomenon, shown in Fig. 1. The neighbors of node i (*i.e.* nodes j, k, l and m) are almost collinear (double line in the figure), and thus, it is clear that if the location of node i is flipped with respect to this line to the new position denoted by i', then the new geometry so obtained is almost compatible with the original inter–node distance measurements (it would be fully compatible if nodes j, k, l and m were perfectly aligned). In order to combat this lack of localizability, connectivity considerations are helpful: observing Fig. 1, one notices that whereas the flipped position

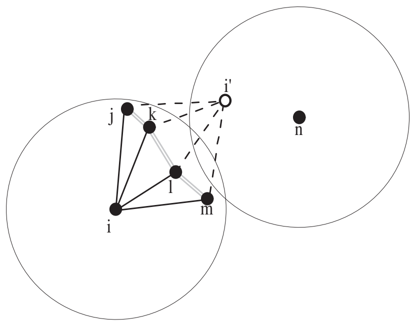


Fig. 1. The flip ambiguity problem.

i' is within the communication range of node n (shown by the circle in the figure), the actual position i is not. If the network is sufficiently dense, one would expect false minima of *CF* to violate some connectivity constraints of this sort. The number of these violations in a candidate topology constitutes our second objective function *CV*.

Formally, *CV* counts the number of connectivity constraints which are not satisfied by the candidate geometry, and is defined as

$$CV = \sum_{i=m+1}^{n} \left(\sum_{j \in N_i} \delta_{ij} + \sum_{j \in \overline{N}_i} (1 - \delta_{ij}) \right), \qquad (6)$$

where $\delta_{ij} = 1$ if $\hat{d}_{ij} > R$ and 0 otherwise.

In order to evaluate the accuracy of estimates, we consider the *normalized localization error* (NLE), defined as

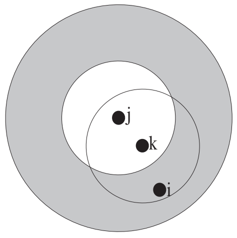
$$NLE = \frac{1}{R} \sqrt{\frac{1}{(n-m)} \sum_{i=m+1}^{n} \left((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right)} \times 100\%. \qquad (7)$$

Thus, assuming that the estimate is unbiased, NLE can be interpreted as the ratio of the standard deviation to the connectivity radius.

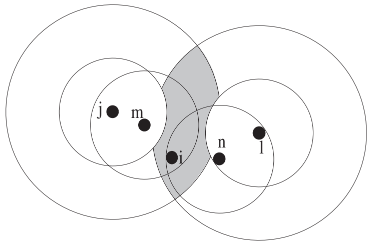
C. Geometrical constraints

The connectivity ranges and the positions of the anchor nodes determine subsets of the overall search space where each single non–anchor node can be positioned. These subsets depend on the type of non–anchor node, and can be defined by means of geometrical constraints. We adopt the following classification based on the position of a non–anchor node with respect to anchor nodes:

- *Class 1 node:* a non–anchor node which is first–level neighbor to at least one anchor node.
  If a node belongs to Class 1, then its position must lie within the intersection of the circles of radius R centered in the anchor nodes which it is neighbor to.
- *Class 2 node:* a non–anchor node which is second–level neighbor to at least one anchor node.
  If a node belongs to class 2, then its position must lie within the intersection of the annuli with inner and outer radii R and $2R$, respectively, centered in the anchor nodes



(a) node i is neighbor to node k, which in its turn is neighbor to anchor node j.



(b) node i is neighbor to nodes m and n, which in their turns are neighbors to anchor nodes j and l, respectively.

Fig. 2. Constraints imposed on *class 2 nodes*.

which it is second–level neighbor to. Fig. 2 shows two examples of class 2 nodes.

- *Class 3 node:* a non–anchor node which belongs to neither class 1 nor class 2.
  If a node is class 3, then its position must lie outside the union of the circles of radius R centered in all anchor nodes.

It is clear that, given a non–anchor node belonging to one of the three classes, it is possible to restrict the space where it can be located. This information can be exploited both in the generation of the initial population of the MOEA and, during the evolutionary process, to constrain the application of the mating operators. Thus, by avoiding the generation of solutions which certainly cannot be optimal (since they violate the geometrical constraints determined by the connectivity ranges and by the known anchor node positions), it is possible to speed up the execution of the evolutionary algorithm. Furthermore, these constraints help alleviate the localizability issues and in particular the flip ambiguity effect. As it turns out, this phenomenon is much more likely to occur if the candidate positions of non–anchor nodes are not constrained within the subspace corresponding to its membership class.

III. The Optimization Framework

MOEAs aim to search for optimal solutions to problems that incorporate multiple performance criteria. MOEAs generate a family of equally valid solutions, where each solution tends to satisfy a criterion to a higher extent than another. Different solutions are compared with each other by using the notion of Pareto dominance. A solution x associated with a performance vector $\mathbf{u}$ dominates a solution y associated with a performance vector $\mathbf{v}$ if and only if, $\forall i \in \{1, \ldots, I\}$ with I the number of criteria, u_i performs better than, or equal to, v_i and $\exists i \in \{1, \ldots, I\}$ such that u_i performs better than v_i , where u_i and v_i are the ith elements of vectors $\mathbf{u}$ and $\mathbf{v}$, respectively. A solution is said to be Pareto optimal if it is not dominated by any other possible solution. The set of Pareto–optimal solutions is denoted as Pareto front. Thus, the aim of a multi–objective search algorithm is to discover a family of solutions that are a good approximation of the Pareto front [20].

MOEAs have been investigated by several authors in recent years [21]. Although there exist a number of recently proposed MOEAs with different peculiarities, we have focused our attention on some of the most popular, namely the Strength Pareto Evolutionary Algorithm (SPEA) [22] and its evolution (SPEA2) [23], the Niched Pareto Genetic Algorithm (NPGA) [24], the different versions of the Pareto Archived Evolution Strategy (PAES) [25], and the Non–dominated Sorting Genetic Algorithm (NSGA) [26] and its evolution (NSGA–II) [27]. On the other hand, the main aim of this paper is not to compare the performances obtained by different MOEAs on the localization problem in WSNs, but rather to prove that this problem can be successfully tackled by using an MOEA. We have used the jMetal [28] implementation of PAES in our experiments.

In the following subsections, we describe the chromosome coding, the objective functions, the genetic operators and the PAES algorithm.

A. Chromosome coding and objective functions

In our optimization framework each chromosome encodes the positions of all non–anchor nodes in the network. Thus, each chromosome consists of $n - m$ pairs of real numbers, where each pair represents the coordinates $\hat{x}$ and $\hat{y}$ of a non–anchor node. The variation range of each coordinate is bounded by the geometrical constraints described in Section II-C. We enforce compliance with these constraints in the initial population. Further, whenever mutations are applied during the evolutionary process, only mutated individuals satisfying these constraints are generated.

Each chromosome is associated with a vector of two elements, which represent the values of the two objective functions CF and CV (Eqs. (4) and (6) in Section II-B).

B. Genetic operators

PAES exploits only mutation during the evolutionary process. We have defined two mutation operators [29]. The first mutation operator, denoted *node mutation* operator, performs a uniform–like mutation: the position of each non–anchor sensor node is mutated with probability $P_U = 1/(n - m)$. Positions are randomly generated within the geometrical constraints imposed on the specific node location.

The second mutation operator, denoted *neighborhood mutation* operator, mutates, with probability $P_U = 1/(n - m)$, the position of each non–anchor sensor node within the geometrical constraints determined for the specific node, but unlike the first operator, it applies the same translation, which has brought the mutated node i from the pre–mutation position to the post–mutation position, to the neighbors of i with probability P_N. Fig. 3(a) shows an example of application of the neighborhood mutation operator. Let i be the sensor node to be mutated. In the figure, we denote with $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{p}}'_i$ the positions of i before and after the application of the mutation operator. The translation applied to node i for shifting this node from $\hat{\mathbf{p}}_i$ to $\hat{\mathbf{p}}'_i$ is also applied to the nodes k and m, which are randomly selected from the set $\{j, k, l, m\}$ of neighbors of i.

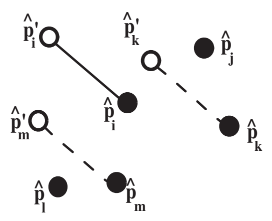
The neighborhood mutation was introduced to deal with particular topological configurations such as the one shown in Fig. 3(b). Here, the actual positions $\mathbf{p}_i$ and $\mathbf{p}_j$ of nodes i and j are marked with squares, while the estimated positions are marked with circles. We note that the distance $\|\mathbf{p}_i - \mathbf{p}_j\|$ between the actual positions is similar to the distance $\|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j\|$ between the estimated positions, thus resulting in a low contribution to CF. Let us suppose that node i is moved from position $\hat{\mathbf{p}}_i$ to position $\hat{\mathbf{p}}'_i$ by applying the node mutation operator. By analyzing the figure, we can realize that, though $\hat{\mathbf{p}}'_i$ is closer to $\mathbf{p}_i$ than $\hat{\mathbf{p}}_i$, the distance $\|\hat{\mathbf{p}}'_i - \hat{\mathbf{p}}_j\|$ between the estimated positions is much larger than that between the actual positions, thus resulting in a considerable increase of CF. This increase will probably lead to discarding the solution with i', even though this solution is certainly better than the one with i. On the other hand, applying the neighborhood mutation, node j would have been translated, with a certain probability, together with i into j' and i', respectively, as shown in Fig. 3(b), thus leaving the distances between estimated and actual positions unchanged. It follows that the solution with i' and j' has the same contribution to CF (as far as these two nodes are concerned) as the solution with i and j, and thus the previous problem is avoided. We experimentally verified that this mutation operator performs better when not all the neighbors are translated with the mutated node. Indeed, if the estimated positions of the neighbors of a mutated node are very close to the actual positions, then translating all of them would considerably worsen the solution. Thus, the translation is applied only to a randomly chosen subset of neighbors.

C. PAES

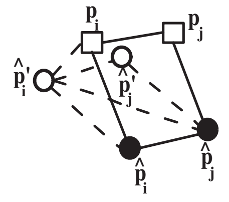
The PAES algorithm was introduced in [25] and probably represents the simplest possible nontrivial algorithm capable of generating diverse solutions in the Pareto optimal set. Further, PAES is characterized by a lower computational complexity than traditional niching methods [25], [30].

PAES consists of three parts: the candidate solution generator, the candidate solution acceptance and the non–dominated solution archive. The candidate solution generator maintains a single current solution c, and, at each iteration, produces a new solution m from c, by using a mutation operator. The candidate solution acceptance compares m with c. Three different cases can arise:

1) c dominates m: m is discarded;
2) m dominates c: m is inserted into the archive and possible solutions in the archive dominated by m are removed; m replaces c in the role of current solution;



(a) An example of application of the neighborhood mutation



(b) Impact of the application of the neighborhood mutation on the CF objective

Fig. 3. The behavior of the neighborhood mutation operator.

3) neither condition is satisfied: m is added to the archive only if it is dominated by no solution contained in the archive; m replaces c in the role of current solution only if m belongs to a region with a crowding degree smaller than, or equal to, the region of c.

The crowding degree is computed by firstly dividing the space where the solutions of the archive lie into a number ($numReg$) of equally sized regions and then by counting the solutions that belong to the regions. The number of these solutions determines the crowding degree of a region. This approach tends to prefer solutions which belong to poorly crowded regions, so as to guarantee a uniform distribution of the solutions along the Pareto front.

PAES terminates after a given number $maxEvals$ of evaluations. The candidate solution acceptance strategy generates an archive which contains only non–dominated solutions. On PAES termination, the archive (at most $archSize$). includes the set of solutions which are an approximation of the Pareto front. At the beginning, the archive is empty and the first current solution is randomly generated. For more details on PAES the reader should refer to [25], [30].

IV. Simulation results

A. Simulation setup

In this section we show the effectiveness of the proposed two–objective evolutionary algorithm in tackling the localization problem in WSNs. We have built different network topologies by randomly placing 200 nodes with a uniform distribution in $T = [0, 1] \times [0, 1] \subset \mathbb{R}^2$. We have varied the percentage of anchor nodes to 8%, 10% and 12% (thus each topology consists of 16, 20 and 24 anchor nodes and 184, 180 and 176 non–anchor nodes, respectively). Further, we have varied the connectivity range R in the interval $[0.11, 0.16]$

TABLE I
AVERAGE VALUES OF SEVERAL MAIN NETWORK INDICATORS FOR DIFFERENT CONNECTIVITY RANGES AND PERCENTAGES OF ANCHOR NODES.

R	node degree	anchor (%)	Cl.1 (%)	Cl.2 (%)	0 anch. (%)	3(or more) anch.(%)
0.11	6.86	8	42.28	31.68	57.72	2.28
		10	50.44	32.72	49.56	3.56
		12	56.02	30.68	43.98	5.97
0.12	8.09	8	51.03	31.85	48.97	2.12
		10	59.67	29.22	40.33	4.50
		12	66.48	25.80	33.52	5.85
0.13	9.41	8	58.04	32.72	41.96	2.17
		10	65.28	29.72	34.72	5.33
		12	69.72	26.14	30.28	9.43
0.14	10.84	8	56.52	30.43	43.48	5.98
		10	67.28	25.94	32.72	8.94
		12	75.06	20.68	24.94	14.20
0.15	12.28	8	62.01	29.35	37.99	7.66
		10	70.94	24.11	29.06	12.78
		12	75.74	21.42	24.26	17.16
0.16	14.16	8	67.83	26.20	32.17	11.09
		10	74.17	22.72	25.83	17.22
		12	81.36	16.76	18.64	24.55

TABLE II
PARAMETER SETUP OF PAES.

Parameter	Value
Size of non–dominated solution archive ($archSize$)	20
Number of regions ($numReg$)	5
Number of fitness evaluations ($maxEvals$)	4×10^5
Node mutation probability (P_M)	0.9
Node rigid translation probability (P_N)	0.3

with step 0.01. The distance measurements between neighboring nodes were generated according to the model (1). We assume that these distance estimates are derived from RSS measurements, which are commonly affected by log–normal shadowing with standard deviation of the errors proportional to the actual range r_{ij} [6]. Thus, the variance of e_{ij} is given by $\sigma^2 = \alpha^2 r_{ij}^2$. A value of $\alpha = 0.1$ was used in the simulations.

For each value of R, 10 random network topologies were generated. After this step, different scenarios were built by varying the percentage of anchor nodes. Thus, we were able to obtain a better control on the effects of both the different connectivity ranges and the different percentage of anchor nodes on the normalized localization error. Table I shows the average values of some network indicators, namely the node degree (considering anchor and non–anchor nodes), the percentage of non–anchor nodes classified in Class 1 and Class 2 (the percentage of nodes in Class 3 can be easily deduced from the first two), the percentage of non–anchor nodes with no anchor node in their neighborhoods and the percentage of non–anchor nodes having at least 3 anchor nodes in their neighborhoods. The analysis of Table I reveals that, when the connectivity range and the percentage of anchor nodes are low, the localization problem becomes very complex: indeed, with a connectivity range $R = 0.11$ and 8% of anchor–nodes, only a small fraction of non–anchor nodes can rely on 3 or more anchor neighbors (2.28%), while more than the half of them (57.72%) are in communication with no

```
 1: procedure FSDPr($s$)
 2:     $NRsol$ = FSDP_SOLVER($s$)
 3:     $\lambda^* =$ UPPERBOUND($s, NRsol$)
 4:     $\lambda = \lambda^*$
 5:     $tries = 0$
 6:     repeat
 7:         $[feasible, RegSol]$ = FSDPr_SOLVER($s, \lambda$)
 8:         $tries = tries + 1$
 9:         if $feasible$ then
10:             return $RegSol$
11:         end if
12:         $\lambda = \lambda/2$
13:     until $tries \leq MAX\_TRIES$
14:     return $NRsol$
15: end procedure
```

Fig. 4. The heuristic strategy used to tune λ in FSDPr.

anchor node. Moreover it is worth noting that, even when the connectivity range is increased to 0.16 and the percentage of anchor nodes to 12%, the average percentage of non–anchor nodes with no anchor neighbor is not negligible (18.64%), while the average percentage of non–anchor nodes with 3 or more anchor neighbors is still significantly low (24.55%).

For each scenario 15 trials of PAES were executed, with parameter values summarized in Table II.

Once the Pareto front approximation has been generated, a solution must be chosen. In our experiments, we verified that the variation interval of CF for the solutions on the final Pareto front approximation is quite small. Thus, we can assume that each solution on the Pareto front can be acceptable with respect to the CF objective. We have validated this hypothesis with a Wilcoxon test, by selecting from each final archive the solutions characterized by the minimum value of CV and the minimum value of CF (in practice, the solutions on the extremes of the Pareto front approximation). Since no statistical difference exists in terms of NLE among the solutions in the final Pareto front approximation, each solution can be actually selected in order to perform a comparison with the two SDP algorithms. For the sake of brevity, we take the solution characterized by the lowest value of CF.

Regarding the FSDPr algorithm, in order to select the optimum value of the regularization term (λ) we have adopted the following heuristic strategy (see Fig. 4). Given a scenario s, we first solve the non–regularized problem (line 2) and then exploit the non–regularized solution ($NRsol$) to compute the upper bound of λ (denoted λ^* in line 3), as suggested in [8]. λ^* is used as starting value in the main tuning loop (line 6-13), where the regularized problem is solved; if the current regularized solution is not feasible, then the current value of λ is divided by two and the new regularized problem is solved again, until a feasible solution is obtained or a maximum number of tries (MAX_TRIES) is reached. In the latter case, the regularized solution coincides with the non–regularized one (*i.e.* $\lambda = 0$). In our experiments we have fixed $MAX\_TRIES = 5$.

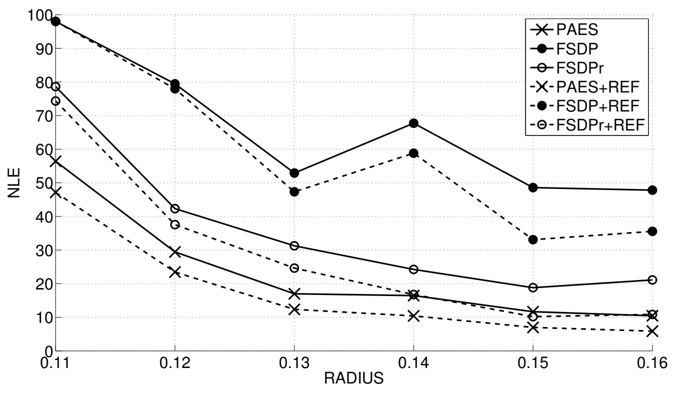


Fig. 5. Comparison among PAES, FSDP and FSDPr without (solid line) and with (dotted line) gradient refinement (REF) using 8% of the nodes as anchor nodes.

B. Performance analysis

In Figs. 5-7, we have plotted as solid lines the average NLE obtained by the FSDP, FSDPr and PAES algorithms versus the six values of radius R used in the experiments. Further, we have shown as dotted lines the average NLE obtained by applying the gradient refinement described in [8], [16] to the final solutions computed by the three algorithms.

The analysis of the figures highlights that, as expected, FSDPr slightly outperforms FSDP. Further, we observe that the gradient refinement is able to improve the estimation only when it is already sufficiently accurate. Indeed, if we consider the solutions generated by FSDP and FSDPr for the lowest connectivity ranges ($R = 0.11$ and $R = 0.12$ for FSDP, and $R = 0.11$ for FSDPr), we realize that the gradient method is unable to perturb the estimation out of the reached local minimum. On the contrary, when the solutions are characterised by a low NLE, the gradient method is able to improve them. In particular, the almost constant gap between the solid lines and the dotted lines for PAES suggests a stable improvement introduced by the refinement phase.

The NLEs obtained by PAES are comparable to those obtained by FSDPr+REF when the connectivity ranges are sufficiently high ($R \geq 0.14$), while they are slightly lower when $R < 0.14$. Further, in all the experiments, PAES+REF considerably outperforms FSDPr+REF. For example, when the percentage of anchor nodes is 8% and the connectivity range is 0.11, from Fig. 5 we can derive that PAES+REF provides estimations which are on average 36.57% more accurate than FSDPr+REF. This percentage increases to 45.75% when the connectivity range is equal to 0.16. Similar conclusions can be deduced by analysing Figs. 6-7, which show the results obtained by using the 10% and 12% of anchor nodes: PAES+REF generate solutions which are 57.84% and 53.72% (61.79% and 48.87%) more accurate when the percentage of anchor nodes is equal to 10% (12%) and the connectivity range is 0.11 and 0.16.

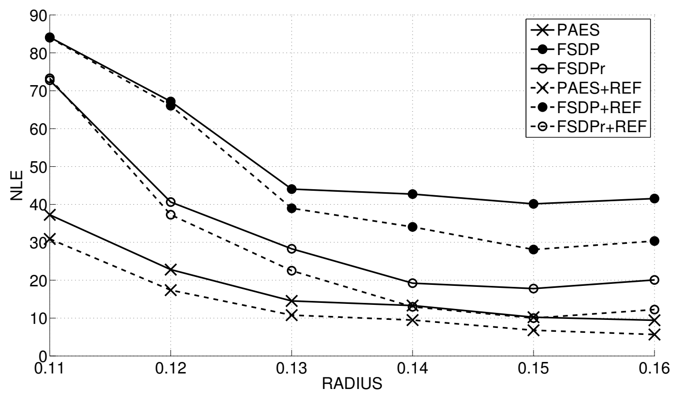


Fig. 6. Comparison among PAES, FSDP and FSDPr without (solid line) and with (dotted line) gradient refinement (REF) using 10% of the nodes as anchor nodes.

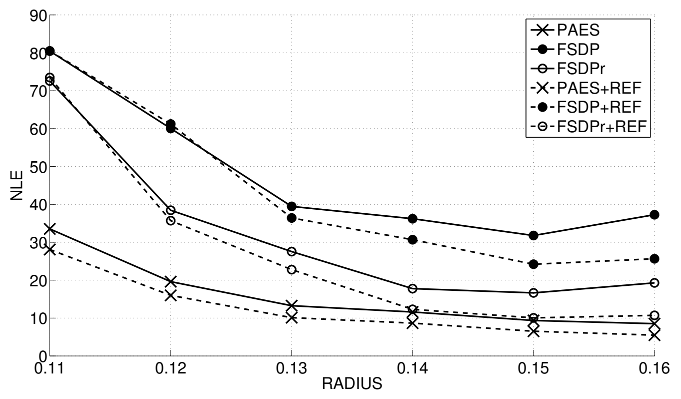


Fig. 7. Comparison among PAES, FSDP and FSDPr without (solid line) and with (dotted line) gradient refinement (REF) using 12% of the nodes as anchor nodes.

V. Conclusions

In this paper we have proposed a two–objective evolutionary algorithm able to accurately solve the fine–grained localization problem in WSNs. The problem is not new in the literature, since several techniques have been proposed in the last decade. The novelty of the approach relies on a better exploitation of the connectivity graph so as to define topological constraints to be used as a second objective function in a multi–objective optimization framework. The topological constraints define zones of the space where each sensor can or cannot be located, thus reducing the search space of the evolutionary algorithm and contextually the chance of ambiguously flipping node locations. Moreover we have discussed the possibility of using a standard gradient–based technique able to refine the final estimation produced by the evolutionary algorithm. We have shown that the proposed approach is able to solve the localization problem with high accuracy for a number of different topologies, connectivity ranges and percentages of anchor nodes. Further, we have discussed how our approach outperforms the standard SDP–based technique and its regularized version.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, pp. 393–422, 2002.

[2] S. Croce, F. Marcelloni, and M. Vecchio, "Reducing power consumption in wireless sensor networks using a novel approach to data aggregation," *The Computer Journal*, vol. 51, no. 2, pp. 227–239, 2008.

[3] F. Marcelloni and M. Vecchio, "An efficient lossless compression algorithm for tiny nodes of monitoring wireless sensor networks," *The Computer Journal*, vol. 52, no. 8, pp. 969–987, 2009.

[4] L. Hu and D. Evans, "Localization for mobile sensor networks," in *MobiCom '04: Proc. of the 10th Int. Conf. on Mobile Computing and Networking*, 2004, pp. 45–57.

[5] L. M. R. Peralta, "Collaborative localization in wireless sensor networks," in *SENSORCOMM 07: Proc. of the 2007 Int. Conf. on Sensor Technologies and Applications*, 2007, pp. 94–100.

[6] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, III, R. L. Moses, and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Mag.*, vol. 22, no. 4, pp. 54–69, 2005.

[7] G. Mao, B. Fidan, and B. D. O. Anderson, "Wireless sensor network localization techniques," *Computer Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[8] P. Biswas, T.-C. Liang, K.-C. Toh, Y. Ye, and T.-C. Wang, "Semidefinite programming approaches for sensor network localization with noisy distance measurements," *IEEE Trans. Autom. Sci. Eng.*, vol. 3, no. 4, pp. 360–371, 2006.

[9] A. A. Kannan, G. Mao, and B. Vucetic, "Simulated annealing based wireless sensor network localization with flip ambiguity mitigation," in *Proc. of the 63-rd IEEE Vehicular Technology Conference*, 2006, pp. 1022–1026.

[10] X. Ji and H. Zha, "Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling," in *Proc. of IEEE INFOCOM*, vol. 4, 2004, pp. 2652–2661.

[11] J. A. Costa, N. Patwari, and A. O. Hero, III, "Distributed weighted–multidimensional scaling for node localization in sensor networks," *ACM Trans. on Sensor Networks*, vol. 2, no. 1, pp. 39–64, 2006.

[12] P. Biswas and Y. Ye, "Semidefinite programming for ad hoc wireless sensor network localization," in *IPSN '04: Proc. of the 3rd Int. Symp. on Information Processing in Sensor Networks*, 2004, pp. 46–54.

[13] P. Tseng, "Second–order cone programming relaxation of sensor network localization," *SIAM J. on Optimization*, vol. 18, no. 1, pp. 156–185, 2007.

[14] Z. Wang, S. Zheng, Y. Ye, and S. Boyd, "Further relaxations of the semidefinite programming approach to sensor network localization," *SIAM J. Optim.*, vol. 19, no. 2, pp. 655–673, 2008.

[15] S. Kim, M. Kojima, and H. Waki, "Exploiting sparsity in sdp relaxation for sensor network localization," *SIAM J. on Optimization*, vol. 20, pp. 192–215, April 2009.

[16] P. Biswas, T.-C. Lian, T.-C. Wang, and Y. Ye, "Semidefinite programming based algorithms for sensor network localization," *ACM Trans. Sen. Netw.*, vol. 2, pp. 188–220, May 2006.

[17] R. Connelly, "Generic global rigidity," *Discrete Comput. Geometry*, vol. 33, no. 4, pp. 549–563, 2005.

[18] A. A. Kannan, B. Fidan, and G. Mao, "Analysis of flip ambiguities for robust sensor network localization," *IEEE Trans. Vehicular Technology*, vol. 59, no. 4, pp. 2057–2070, 2010.

[19] S. Severi, G. Abreu, G. Destino, and D. Dardari, "Understanding and solving flip-ambiguity in network localization via semidefinite programming," in *GLOBECOM'09: Proc. of the 28th IEEE Conf. on Global Telecommunications*, 2009, pp. 3910–3915.

[20] F. Marcelloni and M. Vecchio, "Enabling energy-efficient and lossy-aware data compression in wireless sensor networks by multi-objective evolutionary optimization," *Information Sciences*, vol. 180, no. 10, pp. 1924–1941, 2010.

[21] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 173–195, 2000.

[22] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[23] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization," in *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, K. Giannakoglou *et al.*, Eds. Barcelona, Spain: International Center for Numerical Methods in Engineering (CIMNE), 2002, pp. 95–100.

[24] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A Niched Pareto Genetic Algorithm for Multiobjective Optimization," in *Proc. of the 1st IEEE Conference on Evolutionary Computation*, vol. 1, 1994, pp. 82–87.

[25] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the Pareto Archived Evolution Strategy," *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 149–172, 2000.

[26] N. Srinivas and K. Deb, "Multiobjective optimization using Nondominated Sorting in Genetic Algorithms," *IEEE Trans. Evol. Comput.*, vol. 2, no. 3, pp. 221–248, 1994.

[27] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[28] J. J. Durillo, A. J. Nebro, and E. Alba, "The jMetal framework for multi-objective optimization: Design and architecture," in *CEC '10: Proc. of the IEEE Congress on Evolutionary Computation*, 2010, pp. 1–8.

[29] Z. Michalewicz, *Genetic algorithms + data structures = evolution programs*, 2nd ed. New York, NY, USA: Springer-Verlag New York, Inc., 1994.

[30] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.