Engineering Applications of Artificial Intelligence 50 (2016) 20–31

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Heuristics for designing multi-sink clustered WSN topologies

Andréa Cynthia Santos [a,*], Christophe Duhamel [b], Lorena Silva Belisário [b]

[a] ICD-LOSI, UMR CNRS 6281, Université de Technologie de Troyes 2, rue Marie Curie, CS 42060, 10004 Troyes CEDEX, France
[b] LIMOS-UBP, UMR CNRS 6158, Université Blaise Pascal, Campus des Cézeaux, BP 10125, 63173 Aubière CEDEX, France

ARTICLE INFO

Article history:
Received 12 September 2014
Received in revised form
15 December 2015
Accepted 16 December 2015

Keywords:
Wireless Sensor Network
Multiple sinks
Cluster-based topologies
Heuristics
Independent dominating sets

ABSTRACT

In this study, the problem of building cluster-based topologies for Wireless Sensor Networks with several sinks is considered. The optimization relies on different levels of decision: choosing which sensors are masters and balancing the load among sinks. The topology associated with each sink is modeled as an Independent Dominating Set with Connecting requirements (IDSC). Thus, the solution is a partition of a given graph into as many IDSC as there are sinks. In addition, several optimization criteria are proposed to implicitly or explicitly balance the topology. The network lifetime is improved since it benefits from a clustered structure and the number of hops control. The former reduces the average amount of messages to be sent and the latter improves the average energy consumption for messages to be sent. Different combinations of criteria are proposed in lexicographical order. They are compared in terms of maximum number of clusters per topology, of deviation between the smallest and the biggest number of clusters considering all topologies, and of total number of clusters in the final topology. Two local searches, a two-step local search and a Variable Neighborhood Descent, are developed. Each one is embedded into a multi-start framework. Results are provided for instances with up to 10 000 sensors and up to five sinks.

1. Introduction

A Wireless Sensors Network (WSN) contains a set of sensors which communicate to transmit information about specific detections. A wide range of monitoring applications have already been identified such as risk detection on industrial sites, protected and reserve areas (Ferentinos et al., 2005), intelligent transportation (Tacconi et al., 2010; Tubaishat et al., 2009), and underwater monitoring (Ibrahim et al., 2008). Designing a WSN involves two main levels of decisions: operational and strategic. In the context of WSN, the operational level is usually related to protocols, network issues, communication policies, and traffic loads and their distribution; while the strategic level addresses decisions able to better cope with some issues like minimizing the energy consumption, reducing the traffic, balancing the network load, enhancing the reliability, maximizing the network lifetime, for instance. In this study, we focus on a strategic and theoretical optimization problem occurring in the design of WSN.

Let a topology be a logical structure responsible for the network communication. Here, cluster-based topologies are considered in the design of WSN. A cluster is a set of sensors such that each one plays a specific role of slave, master or bridge. A slave collects and transmits data to the master, the master coordinates the cluster and a bridge is connected to at least two masters and allows inter-cluster communication. Sinks manage the WSN. They usually differ from other sensors and are typically more powerful. Thus, a cluster-based WSN relies on a specific topology where slaves send messages for their corresponding master, and each message is sent on a path towards a sink. Such a path alternates masters and bridges until arriving at a sink. Note that bridges are optional in cluster-based topology. However, inter-cluster communication with bridges globally consumes less energy than a direct master to master communication. In general, clusters are applied when data are highly-correlated (Vlajic and Xia, 2006), since masters can gather similar messages and perform efficient data compression. Thus, in cluster-based topologies, the total number of messages sent on the network is reduced. Another advantage of such a structure is that routing is simplified once a path toward the sink alternates bridges and masters. In Ji et al. (2014), the authors provide an interesting state of the art on data aggregation and data collection. For the latter, some algorithms are presented for tree networks and cell-based data collection. The authors also review these techniques for applications sensitive to accuracy, time complexity and reliability. An interesting study investigating a mathematical model and decomposition for data-gathering cluster-based WSN is found in Lin and Üster (2014). The authors propose an exact model to select sinks and clusters over time periods and solve it by means of a Benders decomposition, using strengthened cuts. The approach is tested on instances with up 250 nodes and 16 candidates sinks.

* Corresponding author.
E-mail address: andrea.duhamel@utt.fr (A.C. Santos).

http://dx.doi.org/10.1016/j.engappai.2015.12.008
0952-1976/

Work of Santos et al. (2012) present a mathematical model and algorithms to compute cluster-based topologies with a unique sink. The problem is modeled as an Independent Dominating Set problem with Connecting requirements (IDSC). This topology follows the IEEE 802.15.4 association procedure (Amendment of IEEE Std 802.15.4, 2007) and is an emerging trend (Cipollone et al., 2007). The mathematical formulation is based on trees and a constructive heuristic, a local search and a Greedy Randomized Adaptive Search Procedures (GRASP) metaheuristic are proposed. Building cluster-based topologies with a single sink means that the methods only have to take a decision on the role of each sensor and ensure the final topology is connected. Then, the optimization relies on minimizing the number of clusters and the average hops (number of edges in the path towards the unique sink). The primary criterion is the minimization of the total number of clusters, the secondary being the average number of hops. It is worth mentioning the latter is not considered in the local search and in GRASP.

This study proposes an extension of work (Santos et al., 2012). Here, we consider multi-sink cluster-based topologies. Thus the problem consists of assigning each sensor to exactly one sink, then partitioning them into clusters. Hence two main levels of decision are involved: balance the number of sensors assigned to each sink and choose the role for each sensor. Both the model and the methods to solve the problem are impacted and this results in the following contributions: (i) the model proposed in Santos et al. (2012) is generalized by considering multiple sinks. Instead of having a unique level of decision (*i.e.* set the role of each sensors subject to connecting requirements), two levels of decisions now have to be handled: the first one consists in assigning each sensor to a unique sink and the second in setting its role. One may note the complexity of solving the problem is increased since it raises the challenge of balancing the load between each sink, which in our clustered model corresponds to balancing the number of sensors assigned to each sink; (ii) several criteria are used in a lexicographical order to balance the assignment of sensors to each sink. Considering these levels of decisions, the following optimization criteria are investigated in this study: minimize the maximum (*min max*) number of clusters per topology, minimize the total number of clusters (*min sum*) in the solution and minimize the deviation (the difference between the largest and the smallest) (*min regret*) number of clusters over all the topologies. These three optimization criteria (*min max*, *min sum*, *min regret*) can also be applied to the average hop distance. (iii) Some interesting insights are raised by the computational experiments such as which criteria are conflicting, giving directions for designing multi-objective strategies. The methods also change significantly compared to the work of Santos et al. (2012) due to the balancing. Thus, (iv) the construction of a feasible solution handles the fact that sensors can be the candidate to several sinks, and that each sink is extended in parallel with each iteration. In addition, (v) dedicated neighborhood structures are developed to improve the balancing, resulting in two local searches: a two-step local search and a Variable Neighborhood Descent (VND). In the former, each topology is first improved independently using the ideas proposed in Santos et al. (2012). Then, new moves are proposed in the second step in order to balance the overall topology. A scalable multi-start framework using either a two-step local search or a VND is proposed and applied to WSNs with up to 10 000 sensors and up to five sinks.

The multi-sink version of the IDSC is referred here as m-IDSC. It is also a NP-hard problem since it generalizes the IDSC (Clark et al., 1991). Multi-sink topologies are adapted for particular applications such as monitoring buildings, whenever the WSN contains a large number of uniformly distributed sensors, as well as when WSNs have a high traffic load. It allows the total network charge to be divided and spread over the network. As a consequence, it can significantly improve the network lifetime. An interesting alternative to multiple sinks consists of adding more sensors in areas with higher traffic, *e.g.* close to the sink (Lian et al., 2006; Wu et al., 2008). However, this strategy is not always convenient since it requires an explicit – and manual – positioning of the new sensors after the initial deployment. Moreover, it may induce unbalanced sensing coverage on the overall network area. Besides, it would require dedicated simulation tools to evaluate the *a priori* energy consumption of a potential sensor and its optimization.

The reminder of this work is organized as follows: a bibliographical review is done in Section 2. Then, the problem definition is given in Section 3, followed by the proposed multi-start metaheuristic in Section 4. Computational experiments are provided in Section 5. Finally, concluding remarks and perspectives are given in Section 6.

2. Related works

Several works in the literature bury the optimization issues into simulations which are done to solve operational issues, with no formal definition of the corresponding optimization problem. As a consequence, the proposed solutions may not properly handle the core of the optimization problem since optimization is a desired feature and not the main focus. Investigating the optimization problems involved in WSN allows to understand its complexity and improve the control, the management and the design of WSN.

Here, the bibliographical review mainly focuses on the works dedicated to optimization problems for WSN using multi-sink. Rather than being exhaustive, we describe works strongly related to our main concerns, *i.e.* to better understand the core of optimization problems involved in a WSN. An interesting entry point is found in Santos et al. (2012) for a state-of-the-art on WSN optimization problems. Moreover, Abbasi and Younis (2007) survey WSN cluster algorithms and present a taxonomy of clustering attributes. The multi-sink impact on energy consumption is assessed by Cipollone et al. (2007) through simulations on specific network topologies (*i.e.* trees routed at multiple sinks). In particular, the authors analyze the performance and the network lifetime whenever this kind of topology is applied. The network lifetime is defined as the first time no path exists to send an event to a sink, events being randomly generated in the simulation scenario. The authors use the following two scenarios: sinks are set uniformly into a grid and, second, sinks are located on the border of a grid. The simulations aim at analyzing the network lifetime, the reliability and the energy consumption. Results indicate trees with fewer hops are more suitable. Obviously, reliability may not be ensured since no secondary path exists whenever an arc fails in a tree. Such an issue could be partially addressed in cluster-based topologies with several bridges between each pair of masters.

Some works focus on the optimal positioning of sinks by means of optimization. Such problems belong to the well-known class of location-allocation and location-relocation problems, which are NP-hard. The main idea in such strategies is to define optimal location for the sinks, which implies a strong hypothesis on the global knowledge of the WSN topology. A case study for setting WSNs in buildings is presented by Saad and Tourancheau (2009). The authors propose a Mixed Integer Linear Programming formulation to provide optimal positioning for mobile sinks. In the instances, sensors are deployed in a grid. Experiments are provided on abstract grids with up to 100 sensors, up to four sinks, and several sink location policies such as on the network border or randomly over the grid. The network lifetime is defined as the first time a sensor runs out of energy. Kim et al. (2005) also investigate the problem of finding optimal sink locations in a WSN, where the optimization consists of using a network flow model to locate p sinks in a WSN in order to maximize the network lifetime, while routing data to the sinks. Two mathematical formulations are presented: the first one considers a fixed number of sinks and the second one sets an upper bound on the number of sinks. Results are provided for WSNs with 20 nodes and up to five sinks. The study (Friedmann and Boukhatem, 2007) focuses on finding an initial optimal sink location and a repositioning strategy. A hill climbing local search with random restarts is used to provide real-time solutions. Experiments are presented for WSNs with 16 sensors and up to five sinks. The authors in Sitanayah et al. (2015) also consider the deployment of sinks and relay nodes in a WSN. The solution has to be robust against the failure of one node and lengths of the communication path for each node towards a sink is upper-bounded. GRASP-based heuristics are proposed to minimize the deployment cost, either considering relay nodes or not. Both variations are evaluated on instances with 100 nodes and 81 candidate relay nodes. The deployment of multiples sinks in order to minimize the average distance of each sensor to its closest sink is investigated in Vincze et al. (2007). The authors consider a multi-hop communication, *i.e* multiple paths can exist from sensors to the sinks and that each sensor sends messages to its closest sink. A mathematical formulation is proposed along with two algorithms: the first one considers a global knowledge of the sensors distribution and is based on the k-means (Forgy, 1965) algorithm, while the second one uses local information to ensure at least 1-hop (path) from sensors to a sink. In the experiments, sending and receiving a packet consumes 1 unit of energy and the network lifetime is defined as the first time an active sensor cannot communicate with the closest sink. Results are presented for WSNs with 1000 sensors and 80 m radio range. Moreover, each sensor has 10 000 units of initial energy power.

Another growing research area in optimization problems for WSNs consists of coupling different levels of decisions and different criteria such as routing and clustering, routing and density control, routing and load balancing, routing for mobile sinks with coverage, and scheduling active sensors to cover target points. For example, Elhabyan and Yagoub (2015) propose a Particle Swarm Optimization (PSO) protocol for performing clustering and routing. Communication between clusters does not use bridges. In addition, a model for energy consumption is also introduced. Results are presented for networks with up to 500 sensors. Aioffi et al. (2011) consider the optimization of mobile sink routes while controlling sensors density. Such a problem is modeled as a covered *min max* selective vehicle routing problem and several heuristic strategies are proposed such as GRASP and Iterated Local Search (ILS). Results are reported for WSN with 600 sensors and up to four mobile sinks. The mobile vehicle to collect data is not necessarily terrestrial. For instance, Wichmann and Korkmaz (2015) consider unmanned aerial vehicles which sets limits on the turning radius. Thus, the authors first focus on computing a set of trips by means of genetic algorithms. Then, each trip is transformed into a smooth trip by removing the sharp turns. The delivery rate and average delays are analyzed for WSN with up to 200 gateways. Work (Kuila and Jana, 2014) focuses on two problems: the first consists of routing strategies with a trade-off between the transmission range and the number of data forwards, and the second relies on balancing energy by applying a cluster-based topology. The cluster-based topology does not consider the set of cluster heads as an independent set. This means two clusters heads can be directly connected in the topology. As a consequence, it can generate conflicts managing messages and also be a source of redundancy. Moreover, a cluster head is a gateway, that is a sensor with a larger battery. The authors propose a multi-objective PSO working in a route first, cluster second strategy. Thus, route information is used to decide about clusters, while trying to balance the load for cluster heads. Experiments are presented on WSNs ranging from 200 to 700 sensors and 60 to 90 cluster heads. In our work, we apply the opposite strategy: first we decide about the network design structure, then the routing is done. Note that an evolutionary algorithm for computing load balanced topologies in WSN is proposed by Kuila et al. (2013), using a similar definition of clusters as in Kuila and Jana (2014). Simulations on instances with 400 sensors and 30 gateways showed the method performs well in terms of load balancing and network lifetime. Load balancing together with mobile data gathering is studied by Zhao et al. (2015), where the clustering is done in a distributed approach and inter-clustering is performed without any bridge. This means an initial configuration is done in order to determine clusters. WSN ranging from 50 to 500 nodes is used in the computational experiments. Work of Castaño et al. (2013) addresses multiple sinks and aims at scheduling active sensors to cover a set of target locations. The sinks are connected by means of a super node and a target location is said to be covered if there is a path to transmit data towards the super node. The lifetime corresponds to the sum of the times assigned to each feasible cover. A decomposition approach is proposed and solved using column generation coupled with heuristics to address the auxiliary problem. Results are presented for WSNs with 50, 100, 150 and 250 sensors, 15 and 30 target locations, and up to 3 sinks.

A work closely related to the problem we consider has been done by He et al. (2012). The authors have addressed the Load-Balanced Connected Dominating Set (LBCDS) problem to compute energy-efficient topologies for WSNs. They propose a Genetic Algorithm whose efficiency is evaluated on medium-size WSNs with up to 1000 nodes. The solutions obtained significantly improve the balance and the network lifetime over existing approaches for the Connected Dominating Set problem. Yet the set of masters are not an independent set.

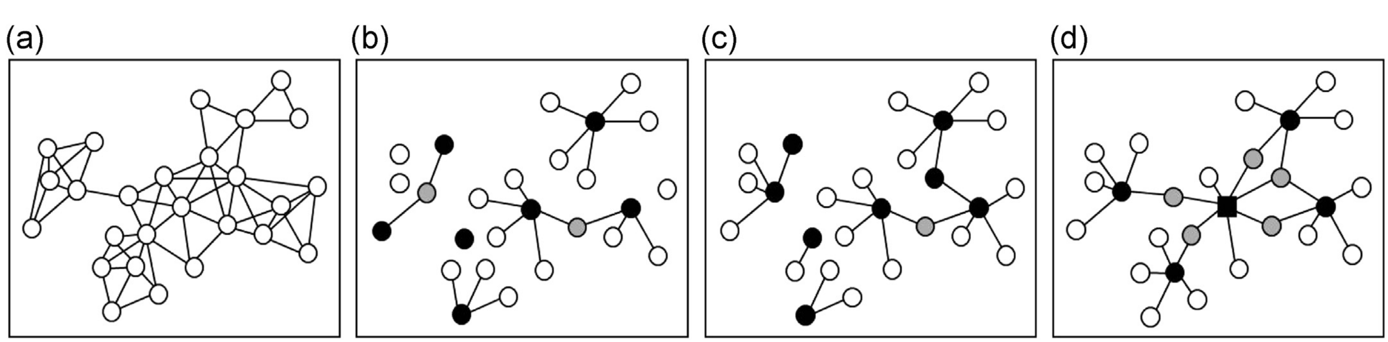

Fig. 1. Example of Independent, Dominating and a feasible cluster-based topology for one sink.

3. Cluster-based topologies with several sinks

Let $G=(V,E)$ be the communication graph associated with the WSN. The set V of vertices contains the subset T of sinks. An edge $[i,j]$ belongs to E if and only if vertices i and j can perform bidirectional communication. Fig. 1(a) presents an example of a communication graph $G=(V,E)$. An independent set $I \subseteq V$ is a subset of vertices that are pairwise non-adjacent like the set of black circles in Fig. 1(b). A dominating set $D \subseteq V$ is a subset of V such that every vertex $v \in V \backslash D$ is adjacent to at least one vertex in D. As an example, the set of black circles in Fig. 1(c) corresponds to a dominating set. Bridges are displayed as gray circles.

Building a cluster-based topology according to Santos et al. (2009, 2012), Vlajic and Xia (2006) means that each vertex is given a specific task: master, slave or bridge, the sink being a special (and predefined) case of master. Thus, the set $M \subseteq V$ of masters must be an independent dominating set of V since it has to hold both properties. Besides, each master has to be connected to the sink through a path, hence the connecting requirement. Fig. 1(d) illustrates a feasible cluster-based topology where the black square, the black circles, the gray circles and the white circles respectively stand for sinks, masters, bridges and slaves. Moreover, the sinks are considered as masters and $T \subseteq M$. The set $S=V \backslash M$ of the remaining vertices are slaves and bridges, the later corresponding to slaves connected to at least two masters. Therefore, they can be implicitly deduced from S and the two subsets M and S define a partition of G. Given such a partition, the resulting topology is a subgraph in which only edges connecting nodes in S with nodes in M are considered. The IDSC problem deals with a single sink, *i.e.* $|T|=1$. It consists of building a set M that minimizes the number of clusters and whose resulting topology is connected.

The m-IDSC extends the IDSC by considering a set T of $m>1$ sinks (hence $m=|T|$). Thus, each vertex must be first assigned to a sink. This means V has to be partitioned into subsets V_t, $\forall t \in T$, which corresponds to the first level of decision. Once the nodes have been assigned to a sink, an IDSC problem has to be solved for each partition V_t. Moreover, for each V_t, $t \in T$, the second level of decision corresponds to the definition of the subsets M_t and S_t, respectively the masters and the slaves/bridges for the sink t. For the sake of clarity, the final topology with m sinks is referred here as an m-topology, made of m subgraphs $G_t=(V_t,E_t)$ where V_t is the corresponding set of sensors and E_t is the set of edges. Here, topologies are considered balanced if they have the same number of clusters. A difference of up to one cluster is acceptable and tolerated since it may not be possible for some sensor distributions to design a full balanced m-topology.

The first level of decision plays an important role in the whole optimization process. Fig. 2(a) illustrates a poor assignment, even if the topologies are balanced, due to a large number of hops. This induces more energy consumption when sending messages to the sink, since more hops are used to route messages to a sink. Fig. 2(b) shows a better structure with a fewer number of hops, while the topologies remain balanced.

As mentioned before, the clustered model differs from the model used in He et al. (2012) since no connection is allowed between any two masters, and bridges are defined to ensure connection between clusters and the sinks. Thus, the set of masters is also an independent set which makes the m-IDSC problem more difficult than the LBCDS.

Two evaluations are used for the IDSC: the total number of masters and the hop average. The former corresponds to the number of clusters in the topology. The later defines the average distance of each slave to its corresponding sink. Thus, it gives an estimation of the energy consumption due to message transmission. Both values have to be minimized, the former being used as primary criterion and the latter as secondary criterion. Such an approach might lead to highly unbalanced energy consumption among the sinks since the m-IDSC problem involves several sinks. Thus the criteria have to be adapted to the context of the m-IDSC. Two values are now considered in the optimization: (A) the number of clusters per topology and (B) the average hop number per topology. They are subjected to the three following optimization objectives: (1) minimizing the maximum value (A or B) per topology, (2) minimizing the sum of values over the topology (A or B) and (3) minimizing the deviation (the difference between the largest and the smallest) values over the m-topology (A or B), which leads to six potential combinations for single objective optimization criteria.

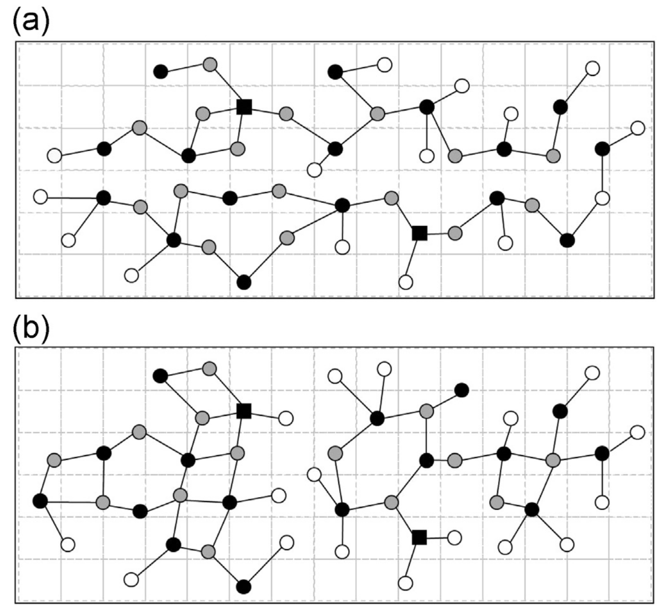

Fig. 2. Example of multi-sink topologies with $m=2$ sinks.

Several combinations of the possible evaluations have been investigated. We present here the two most promising. Thus, the solutions and the moves investigated will be evaluated by using one of the following two combinations:

$(A1 \to A2 \to B1)$: The first (A1) aims at minimizing the maximum number of clusters per topology; the second, (A2) looks to minimizing the total number of clusters over all topologies; and, the third (B1) is used to minimize the maximum average hop per topology. Thus, a solution is evaluated using the criteria in the lexicographic order $A1 \to A2 \to B1$.

$(A1 \to A3 \to B1)$: The first (A1) aims at minimizing the maximum number of clusters per topology; the second (A3) looks to minimizing the deviation of the number of clusters over all topologies; and, the third (B1) is used to minimize the maximum average hop per topology. In this strategy, a solution is evaluated by the criteria in the lexicographic order $A1 \to A3 \to B1$.

The second combination $(A1 \to A3 \to B1)$ is referred here as "Balanced" as it explicitly uses the balancing strategy $A3$, while the combination $(A1 \to A2 \to B1)$ is referred as "Unbalanced".

4. Multi-start heuristics for the *m*-IDSC problem

Multi-start heuristics are effective strategies for solving NP-hard problems (Martí, 2003). They consist of building a feasible solution at each iteration, which is then improved by a local search. This sequence is repeated to get several solutions in the search space and the incumbent solution is kept, characterizing a multi-start. Such a framework is interesting in the WSN context since it requires few parameters and their calibration remains quite light.

Algorithm 1 provides a macroview of a multi-start heuristic scheme, where S and S^* are respectively the incumbent and the best solution. Variables are initialized in line 1. The loop of lines 2–8 is repeated until the stopping criterion is met. A solution is built and improved, respectively in lines 3 and 4. The best solution is updated from lines 5 to 7. The optimization criteria used in the Balanced and the Unbalanced strategies are respectively applied in the lexicographic order $(A1 \to A3 \to B1)$ and $(A1 \to A2 \to B1)$, to evaluate the solutions in the local search and when updating the incumbent solution.

Algorithm 1. General multi-start framework.

1 Initialize S and S^*;
2 **repeat**
3   Build an initial solution S;
4   Apply a local search to improve S;
5   **if** *S improves S^** **then**
6   Update S^*;
7   **end**
8 **until** (*stopping criterion is not met*);

A greedy constructive heuristic and two local searches have been developed for the m-IDSC problem. The general idea of the constructive procedure is to expand a topology from each sink, by adding one master at a time, and its available neighbors (a cluster) at each iteration. The procedure stops when every sensor belongs to a topology. A list of candidate nodes to become master is built and updated throughout the process. The candidate is randomly chosen in this list. Such a generation is tailored for the multi-start procedure since the random choice allows us to build the initial solution in different regions of the search space. A node can belong to the neighborhood of several partial topologies and it may be selected several times. To avoid any conflict (*i.e.* two or more sinks ask for the same candidate) and inconsistency (*i.e* a node is selected as a master in two or more topologies), information is kept to identify so far if a candidate is still available or not.

The greedy heuristic tries to keep the m-topology balanced by inserting one cluster for each topology, at each iteration. Balancing is not guaranteed, since decisions are taken locally. This especially holds when a node is candidate for several topologies. Moreover, wrong decisions may block the expansion of a topology. For instance, the list of candidates in Fig. 3 is $\{a, b, c\}$, dotted circles being the available sensors to enter the m-topology. Choosing node a blocks the expansion of the other topology.

Algorithm 2 illustrates the constructive procedure. Let $N(i)$ be the set of all direct neighbors of $i \in V$, *i.e.* $N(i) = \{j \in V \mid [i,j] \in E\}$. Thus $j \in N(i)$ is said to be a neighbor of i. All other notations have already been defined in Section 3. The m-topology is initialized (lines 1–7). Initially, every node $i \in V$ is made available (line 1). Then, a sink $t \in T$ is included in the topology, and M_t and V_t are respectively updated (lines 3 and 4). The added nodes are set unavailable (line 5). The procedure sets slaves for each topology (line 7). One slave is included and connected to a sink t at a time, and then set unavailable. The first level of slaves plays a key role in providing a more balanced m-topology. Inserting one slave at a time allows a better balancing of the number of sensors in the initial m-topology. The loop (lines 8–18) is repeated until all nodes have been assigned. The candidate list refers to the nodes that can immediately become master. The candidate list associated with each sink is updated (line 9). Only available nodes are considered.

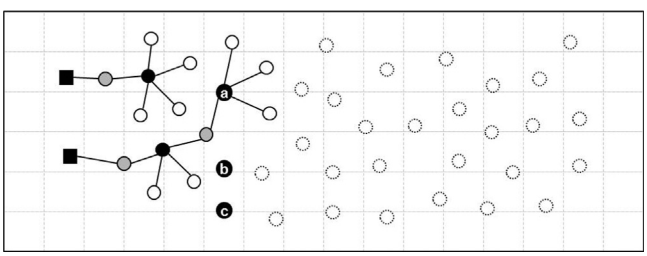

Fig. 3. Example of a wrong decision.

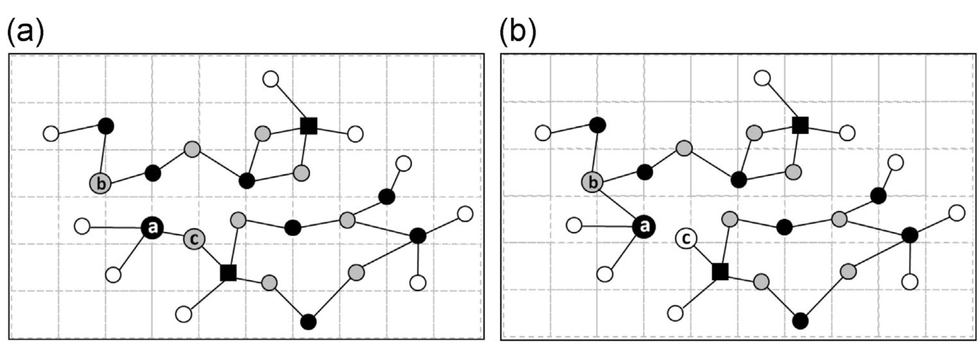

Fig. 4. Example of a move in $\mathcal{N}_{2a}$.

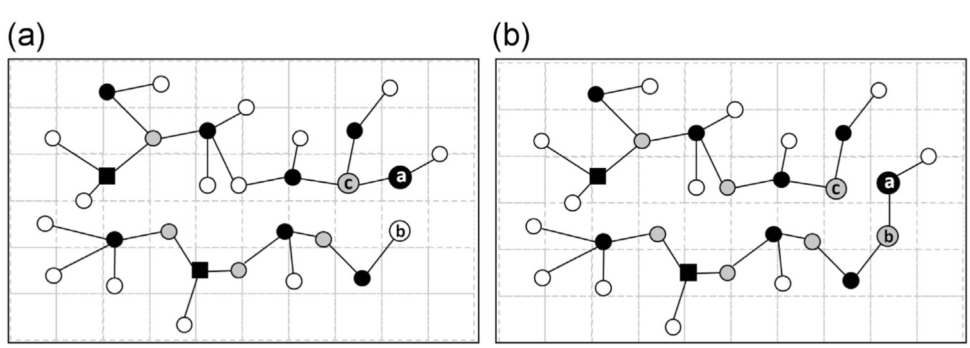

Fig. 5. Example of a move in $\mathcal{N}_{2b}$.

Algorithm 2. The randomized constructive heuristic.

Input: $G=(V,E)$, T
Output: $m-$topology $=\{G_t=(V_t,E_t),\ M_t\ \forall t\in T\}$

$\forall i \in V$ set i as available;
for $t \leftarrow 1$ *to* m **do**
  $M_t \leftarrow \{t\}$;
  $V_t \leftarrow M_t$;
  set t as unavailable;
end
update slaves $(G_t=(V_t,E_t), S_t\ \forall t\in T)$;
while ($\exists$ *available nodes*) **do**
  update candidate list for all $m-$topology;
  for $t \leftarrow 1$ *to* m **do**
    select i from the candidate list of sink t;
    $M_t \leftarrow M_t \cup \{i\}$;
    $S_t \leftarrow S_t \cup \{j \mid$ for all $j\in N(i)$ and j is available$\}$;
    $V_t \leftarrow M_t \cup S_t$;
    $E_t \leftarrow E_t \cup \{[i,j] \mid$ for all $j \in N(i)$ and j is available$\}$;
    set i and all available neighbors of $N(i)$ as unavailable;
  end
end
return *m-topology*, $M_t\ \forall t \in T$;

Table 1
Results for the Unbalanced ($A1 \rightarrow A2 \rightarrow B1$) strategy on the first test set.

m	n	RCH					MS+2P					MS+VND				
		$A1$	$A2$	$A3$	$B1$	$t(s)$	$A1$	$A2$	$A3$	$B1$	$t(s)$	$A1$	$A2$	$A3$	$B1$	$t(s)$
2	100	7	13	1	4.33	0.06	6	12	0	3.60	0.07	6	12	0	3.60	0.18
	150	7	14	0	5.67	0.07	7	13	1	4.00	0.06	7	13	1	4.00	0.41
	200	7	14	0	6.00	0.09	7	13	1	3.60	0.10	7	13	1	3.60	0.75
	250	8	15	1	3.43	0.09	6	12	0	4.40	0.15	6	12	0	4.40	1.13
	300	8	15	1	4.67	0.10	7	13	1	2.80	0.19	7	13	1	2.80	1.58
	350	8	15	1	4.57	0.12	7	13	1	4.00	0.21	7	13	1	4.00	2.27
	400	8	15	1	2.86	0.15	7	13	1	2.80	0.24	7	13	1	2.80	2.91
	450	8	15	1	3.33	0.18	7	14	0	3.33	0.30	7	13	1	3.60	3.88
	500	8	16	0	3.71	0.21	7	14	0	3.33	0.37	7	14	0	3.33	5.23
	550	8	15	1	4.33	0.25	7	13	1	3.00	0.39	7	13	1	3.00	6.37
	600	8	15	1	3.43	0.29	7	13	1	4.40	0.46	7	13	1	3.60	6.87
	650	8	16	0	2.86	0.36	6	12	0	3.60	0.50	6	12	0	3.60	7.79
	700	8	16	0	3.71	0.37	7	13	1	3.33	0.58	7	13	1	3.33	9.37
	750	8	15	1	2.86	0.42	7	12	2	2.67	0.63	7	12	2	2.67	11.76
	800	8	16	0	3.14	0.48	7	13	1	2.80	0.71	7	13	1	2.80	13.51
	850	8	16	0	2.86	0.53	7	14	0	3.00	0.80	7	14	0	3.00	14.66
	900	8	15	1	3.43	0.60	7	13	1	3.00	0.85	7	13	1	3.00	16.57
	950	8	15	1	4.29	0.65	7	13	1	4.67	1.06	7	13	1	4.67	19.78
	1000	8	15	1	2.86	0.70	7	13	1	2.80	1.07	7	13	1	2.80	21.34
3	100	5	13	1	3.50	0.03	5	13	1	3.50	0.04	5	13	1	3.50	0.19
	150	5	13	1	2.67	0.04	5	12	2	3.00	0.06	5	12	2	3.00	0.37
	200	5	15	0	2.50	0.05	5	13	1	3.33	0.09	5	13	1	3.33	0.69
	250	5	14	1	2.67	0.07	4	12	0	3.33	0.12	4	12	0	3.33	1.06
	300	5	15	0	4.00	0.09	5	13	1	5.00	0.14	5	13	1	5.00	1.61
	350	6	16	1	3.60	0.12	5	14	1	4.00	0.17	5	14	1	4.00	2.13
	400	5	15	0	5.00	0.21	5	13	2	3.50	0.21	5	13	2	3.50	3.29
	450	5	15	0	4.00	0.17	5	13	1	3.33	0.25	5	13	1	3.33	3.50
	500	5	15	0	4.00	0.20	5	14	1	3.00	0.28	5	14	1	2.67	4.53
	550	6	16	1	3.20	0.25	5	14	1	3.00	0.35	5	14	1	3.00	5.89
	600	6	16	1	3.20	0.28	5	13	1	4.00	0.43	5	13	1	4.00	7.31
	650	5	15	0	2.50	0.32	5	13	1	4.00	0.47	5	13	1	4.00	9.09
	700	6	16	1	3.00	0.38	5	13	1	3.33	0.52	5	13	1	3.33	9.54
	750	5	15	0	5.00	0.41	5	13	1	0.67	0.64	5	13	1	0.67	14.21
	800	5	15	0	3.50	0.46	5	13	1	2.67	0.61	5	13	1	2.67	13.75
	850	6	16	1	3.50	0.52	5	15	0	2.50	0.69	5	15	0	2.50	16.83
	900	6	17	1	3.20	0.59	5	14	1	2.67	0.77	5	14	1	2.67	20.07
	950	6	16	1	6.00	0.62	5	14	1	5.00	0.86	5	14	1	5.00	19.24
	1000	6	16	1	3.50	0.73	5	14	1	2.67	0.90	5	14	1	2.67	22.58
4	100	4	13	2	3.00	0.03	4	13	2	3.00	0.04	4	13	2	3.00	0.29
	150	4	14	2	2.67	0.04	4	13	2	3.00	0.06	4	13	2	3.00	0.35
	200	4	15	1	2.67	0.06	4	14	1	3.33	0.09	4	14	1	3.33	0.67
	250	4	14	1	3.00	0.07	4	13	1	2.00	0.13	4	13	1	2.00	1.15
	300	4	16	0	3.33	0.09	4	13	1	4.00	0.19	4	13	1	4.00	1.61
	350	4	16	0	2.67	0.11	4	15	1	2.67	0.17	4	15	1	2.67	2.05
	400	4	15	1	3.00	0.14	4	13	1	3.00	0.21	4	13	1	3.00	2.66
	450	5	16	2	3.00	0.17	4	14	1	3.00	0.30	4	14	1	3.00	4.40
	500	5	15	2	3.00	0.20	4	14	1	2.67	0.36	4	14	1	2.67	4.79
	550	5	16	2	3.00	0.23	4	15	1	3.33	0.33	4	15	1	3.33	5.07
	600	4	16	0	4.00	0.28	4	15	1	3.33	0.39	4	15	1	3.33	6.20
	650	4	15	1	3.33	0.31	4	14	1	2.67	0.46	4	14	1	2.67	8.15
	700	5	16	2	3.00	0.35	4	15	1	4.00	0.52	4	15	1	4.00	9.54
	750	5	16	3	3.00	0.41	4	14	1	2.00	0.67	4	14	1	2.00	12.33
	800	4	16	0	2.67	0.41	4	14	1	2.67	0.65	4	14	1	2.67	14.09
	850	5	17	1	2.67	0.48	4	16	1	2.00	0.63	4	16	1	2.00	14.27
	900	5	16	2	3.50	0.54	4	15	1	2.67	0.95	4	15	1	2.67	19.42
	950	4	16	0	4.00	0.56	4	14	1	3.00	0.82	4	14	1	3.00	17.59
	1000	5	16	2	2.67	0.63	4	14	1	2.67	0.96	4	14	1	2.67	23.67

Table 2
Results for the Balanced ($A1 \to A3 \to B1$) strategy on the first test set.

m	n	RCH					MS+2P					MS+VND				
		A1	A2	A3	B1	t(s)	A1	A2	A3	B1	t(s)	A1	A2	A3	B1	t(s)
2	100	7	14	0	3.33	0.03	6	12	0	3.60	0.04	6	12	0	3.60	0.20
	150	7	14	0	5.67	0.04	7	14	0	4.33	0.08	7	14	0	4.33	0.41
	200	7	14	0	6.00	0.06	7	14	0	3.67	0.09	7	14	0	3.67	0.76
	250	8	16	0	3.43	0.08	6	12	0	4.40	0.12	6	12	0	4.40	1.15
	300	8	16	0	4.00	0.10	7	14	0	2.33	0.17	7	14	0	2.33	1.63
	350	8	16	0	4.57	0.12	7	14	0	4.00	0.21	7	14	0	4.00	2.37
	400	8	16	0	3.14	0.14	7	14	0	2.00	0.24	7	14	0	2.00	2.94
	450	8	16	0	3.43	0.18	7	14	0	3.33	0.28	7	14	0	3.33	3.90
	500	8	16	0	3.71	0.21	7	14	0	3.33	0.36	7	14	0	3.33	5.38
	550	8	16	0	3.43	0.24	7	14	0	2.67	0.38	7	14	0	2.67	6.33
	600	8	16	0	3.43	0.29	7	14	0	3.67	0.47	7	14	0	3.67	6.96
	650	8	16	0	2.86	0.34	6	12	0	3.60	0.51	6	12	0	3.60	8.08
	700	8	16	0	3.71	0.38	7	14	0	3.33	0.58	7	14	0	3.33	10.34
	750	8	16	0	2.86	0.43	7	14	0	2.67	0.65	7	14	0	2.67	12.67
	800	8	16	0	3.14	0.57	7	14	0	3.00	0.70	7	14	0	3.00	13.67
	850	8	16	0	2.86	0.54	7	14	0	3.00	0.80	7	14	0	3.00	14.76
	900	8	15	1	3.43	0.63	7	14	0	2.67	0.83	7	14	0	2.67	16.83
	950	8	16	0	4.29	0.66	7	14	0	3.67	1.05	7	14	0	3.67	19.96
	1000	8	16	0	3.14	0.71	7	14	0	2.67	1.03	7	14	0	2.67	21.66
3	100	5	15	0	3.00	0.04	5	15	0	3.00	0.04	5	15	0	3.00	0.20
	150	5	15	0	3.00	0.07	5	15	0	3.00	0.06	5	15	0	3.00	0.39
	200	5	15	0	2.50	0.10	5	15	0	2.50	0.09	5	15	0	2.50	0.70
	250	5	15	0	3.00	0.12	4	12	0	3.33	0.11	4	13	0	2.67	1.08
	300	5	15	0	4.00	0.10	5	15	0	2.50	0.17	5	15	0	2.50	1.63
	350	6	18	0	3.60	0.12	5	15	0	3.50	0.18	5	15	0	3.50	2.16
	400	5	15	0	5.00	0.14	5	15	0	3.00	0.22	5	15	0	3.00	3.25
	450	5	15	0	4.00	0.17	5	15	0	2.50	0.28	5	15	0	2.50	3.54
	500	5	15	0	4.00	0.21	5	15	0	2.50	0.30	5	15	0	2.50	4.56
	550	6	18	0	2.80	0.26	5	15	0	3.00	0.35	5	15	0	3.00	5.87
	600	6	18	0	3.20	0.28	5	15	0	3.00	0.41	5	15	0	3.00	7.33
	650	5	15	0	2.50	0.34	5	15	0	2.50	0.47	5	15	0	2.50	8.72
	700	6	18	0	2.80	0.38	5	15	0	3.00	0.52	5	15	0	3.00	8.93
	750	5	15	0	5.00	0.41	5	15	0	1.00	0.63	5	15	0	1.00	12.80
	800	5	15	0	3.50	0.47	5	15	0	2.50	0.61	5	15	0	2.50	13.08
	850	6	18	0	3.20	0.52	5	15	0	2.50	0.66	5	15	0	2.50	16.38
	900	6	18	0	2.80	0.59	5	15	0	3.00	0.77	5	15	0	3.00	21.03
	950	6	18	0	4.40	0.63	5	15	0	3.00	0.85	5	15	0	3.00	19.15
	1000	6	18	0	2.80	0.69	5	15	0	2.50	0.88	5	15	0	2.50	20.75
4	100	4	16	0	2.67	0.04	4	16	0	2.67	0.04	4	16	0	2.67	0.19
	150	4	16	0	3.33	0.06	4	16	0	2.67	0.06	4	16	0	2.67	0.38
	200	4	16	0	2.67	0.05	4	16	0	2.00	0.08	4	16	0	2.00	0.66
	250	4	16	0	2.67	0.07	4	16	0	2.67	0.11	4	16	0	2.00	1.17
	300	4	16	0	3.33	0.09	4	16	0	2.67	0.14	4	16	0	2.67	1.54
	350	4	16	0	2.67	0.11	4	16	0	2.67	0.16	4	16	0	2.67	2.05
	400	4	16	0	2.67	0.14	4	16	0	2.67	0.20	4	16	0	2.67	2.60
	450	5	20	0	4.00	0.17	4	16	0	2.67	0.24	4	16	0	2.67	4.38
	500	5	20	0	3.00	0.20	4	16	0	2.67	0.27	4	17	0	2.67	4.89
	550	5	20	0	3.00	0.23	4	16	0	2.67	0.29	4	16	0	2.67	5.08
	600	4	16	0	4.00	0.29	4	16	0	2.67	0.37	4	16	0	2.67	6.35
	650	4	16	0	2.67	0.31	4	16	0	2.00	0.38	4	16	0	2.00	8.12
	700	5	20	0	3.00	0.36	4	16	0	2.67	0.49	4	16	0	2.67	9.62
	750	5	20	0	5.00	0.39	4	16	0	2.00	0.54	4	16	0	2.00	12.19
	800	4	16	0	2.67	0.41	4	16	0	2.67	0.55	4	16	0	2.67	13.84
	850	5	20	0	2.50	0.49	4	16	0	2.00	0.59	4	16	0	2.00	14.17
	900	5	20	0	4.00	0.55	4	16	0	2.67	0.75	4	17	0	2.00	19.33
	950	4	16	0	4.00	0.56	4	16	0	2.67	0.74	4	16	0	2.67	17.31
	1000	5	20	0	3.50	0.63	4	16	0	2.00	0.75	4	16	0	2.00	23.60

A node is randomly selected from the candidate list (line 11) to become a master (line 12). Then, its available neighbors are included in S (line 13). $G_t = (V_t, E_t)$ is updated (lines 14–15). The added nodes become unavailable (line 16). The feasible m-topology is returned by the procedure (line 19).

The solution built by the constructive procedure is then submitted to one of the two proposed local search procedures. The first one performs two phases and uses a first improvement strategy. In the first phase, a local optimization is done in each topology to try to reduce its number of clusters. The second phase is focused on balancing the number of clusters among all topologies. The second local search is a Variable Neighborhood Descent (VND) (Mladenovic and Hansen, 1997).

The first phase of the two phases local search corresponds to the procedure proposed in Santos et al. (2012) for the IDSC problem. It relies on the neighborhood structure $\mathcal{N}_1$, in which a move consists of transforming a bridge into a master and updating the topology accordingly. Bridges are connected to at least two masters. Thus, after applying such a move, the final resulting topology necessarily contains less masters if it remains connected. Criterion $A3$ is useless here since only one topology is considered at a time.

In the second phase of the local search, the general idea is to use a neighborhood structure $\mathcal{N}_2$ in order to transfer clusters among the topologies to get a better balance. Thus, two topologies can be considered for a move if they differ by more than one in the number of clusters. Otherwise transferring one cluster cannot improve the balance of the m-topology. Moreover, criterion $A2$ is useless here since the total number of clusters does not change. Two subsets of moves have been considered in $\mathcal{N}_2$: the first one ($\mathcal{N}_{2a}$) connects a master from a topology to a bridge in another topology, and the second one ($\mathcal{N}_{2b}$) connects a master from a topology to a slave in another topology. We have $\mathcal{N}_{2a} \cup \mathcal{N}_{2b} \subset \mathcal{N}_2$ and those moves are investigated simultaneously when exploring $\mathcal{N}_2$.

Let G_1 and G_2 be a pair of unbalanced topologies such that G_2 has more clusters than G_1, *i.e.* $|M_2| \geq |M_1|+2$. In a move from $\mathcal{N}_{2a}$, connecting a master $i \in G_2$ to a bridge $j \in G_1$ means some of its slaves can change their task. They may become bridges in G_1 and some nodes considered as bridges in G_2 may become slaves in G_1. Fig. 4 illustrates such a move. Fig. 4(a) shows the initial m-topology and nodes "a", "b" and "c" involved in the move. In Fig. 4(b), the move has been performed. The cluster with the master "a" is connected to a bridge in the other topology. Then, node "c" becomes a slave and both topologies have the same number of clusters.

In a move from $\mathcal{N}_{2b}$, connecting a master $i \in G_2$ to a slave $j \in G_1$ implies j becomes a bridge. All other slaves are updated accordingly as in the first move. Fig. 5 shows an example for the second move. Fig. 5(a) and (b) display respectively the initial m-topology and the resulting topology after the move. Nodes "a", "b" and "c" are involved in the move. In this case, the connection between "a" and "b" implies "b" becomes a bridge.

The VND local search uses the same neighborhood structures as the two phases local search, but in a different way. Only one neighborhood structure $\mathcal{N}_k$ is active in each iteration. Initially, $k=1$. Given a current solution S and the active neighborhood structure $\mathcal{N}_k$, if an improving solution $S' \in \mathcal{N}_k(S)$ is found, it becomes the new current solution and k is set to 1. Otherwise, the current solution is left unchanged and k is increased. Thus, at the end, the current solution S is a local optimum with respect to $\mathcal{N}_1$ and $\mathcal{N}_2$.

Algorithms 3 and 4 respectively illustrate the two-phases local search and the VND. Given an initial solution S_0, they use the first improvement strategy (selection operator *firstImproving()*) when exploring the neighborhoods $\mathcal{N}_1$ and $\mathcal{N}_2$. The comparison (*betterThan*) between solutions relies on the given lexicographic order.

Table 3
Methods comparison on the first test set.

Methods	m	$A1$ only			$A1 \rightarrow A2 \rightarrow B1$		
		Better	Similar	Worse	Better	Similar	Worse
Unbalanced strategy							
RCH *vs.* MS+2P	2	0	2	17	0	0	19
	3	0	10	9	0	1	18
	4	0	11	8	0	1	18
RCH *vs.* MS+VND	2	0	2	17	0	0	19
	3	0	10	9	0	1	18
	4	0	11	8	0	1	18
MS+2P *vs.* MS+VND	2	0	19	0	0	17	2
	3	0	19	0	0	18	1
	4	0	19	0	0	19	0
Balanced strategy							
RCH *vs.* MS+2P	2	0	2	17	0	0	19
	3	0	10	9	0	4	15
	4	0	11	8	0	5	14
RCH *vs.* MS+VND	2	0	2	17	0	0	19
	3	0	10	9	0	4	15
	4	0	11	8	0	4	15
MS+2P *vs.* MS+VND	2	0	19	0	0	19	0
	3	0	19	0	0	18	1
	4	0	19	0	0	17	2

Table 4
Overview of the deviation $A3$ on the first test set.

Method	m	$A3$ equals 0		$A3$ equals 1	
		Unbalanced	Balanced	Unbalanced	Balanced
MS+2P	2	6	19	12	0
	3	2	19	15	0
	4	0	19	17	0
MS+VND	2	5	19	13	0
	3	2	19	15	0
	4	0	19	17	0

Table 5
Overview of the total number of clusters $A2$ in the m-topology on the first test set.

Method	m	Unbalanced *vs.* Balanced		
		Better	Similar	Worse
MS+2P	2	13	6	0
	3	17	2	0
	4	18	1	0
MS+VND	2	16	3	0
	3	18	1	0
	4	18	1	0

Algorithm 3. The two-phases local search.

Input: $G=(V,E)$, initial solution S_0
Output: solution S

```
S ← S_0;
// first phase: improve each topology
stop ← false;
while (stop = false) do
    if (∃S' ∈ N_1(S) | S' betterThan S) then
        S ← firstImproving(N_1(S));
    else
        stop ← true;
    end
end
// second phase: improve the balance
stop ← false;
while (stop = false) do
    if (∃S' ∈ N_2(S) | S' betterThan S) then
        S ← firstImproving(N_2(S));
    else
        stop ← true;
    end
end
return S
```

Table 6
Results for the Unbalanced ($A1 \rightarrow A2 \rightarrow B1$) strategy on the second test set.

m	n	RCH					MS+2P					MS+VND				
		$A1$	$A2$	$A3$	$B1$	$t(s)$	$A1$	$A2$	$A3$	$B1$	$t(s)$	$A1$	$A2$	$A3$	$B1$	$t(s)$
2	2000	153	304	2	18.03	0.36	128	255	1	19.37	27.14	128	255	1	19.37	27.42
	3000	164	325	3	19.90	0.59	129	257	1	22.34	51.53	128	255	1	22.25	52.15
	4000	168	331	5	15.95	0.87	130	260	0	18.64	93.88	130	260	0	18.47	95.49
	5000	169	332	6	20.23	1.17	134	268	0	19.04	154.24	133	266	0	19.03	157.39
	6000	169	336	2	14.40	1.58	135	270	0	13.58	232.35	135	269	1	14.58	240.51
	7000	172	342	2	18.93	1.92	137	273	1	16.01	322.17	135	270	0	18.10	331.84
	8000	176	349	3	23.01	2.35	140	279	1	20.00	420.76	139	277	1	24.91	439.11
	9000	175	346	4	13.80	2.85	138	275	1	14.26	562.67	138	275	1	14.26	588.62
	10 000	175	350	0	12.00	3.44	139	278	0	12.20	715.04	138	276	0	13.15	747.66
3	2000	105	312	2	19.16	0.36	88	262	1	28.74	22.55	88	262	1	28.74	35.74
	3000	110	330	0	21.50	0.58	90	260	7	18.90	45.34	86	257	4	18.94	72.97
	4000	111	330	2	17.64	0.84	90	269	1	27.37	76.29	89	267	2	20.77	128.34
	5000	115	339	4	14.65	1.18	91	273	0	16.49	137.08	91	272	2	13.89	229.93
	6000	116	345	2	15.15	1.51	93	279	0	16.33	204.93	92	277	1	15.05	344.99
	7000	117	347	4	18.52	1.87	93	279	0	21.28	276.40	92	280	0	21.43	461.90
	8000	118	344	6	15.13	2.33	93	279	0	15.91	384.16	93	277	1	16.00	658.48
	9000	119	356	1	15.54	2.77	94	281	1	15.16	501.43	93	279	2	13.80	849.72
	10 000	119	356	1	11.88	3.35	94	281	1	10.39	766.03	94	280	1	10.39	1105.46
4	2000	80	313	4	18.21	0.32	71	280	2	25.51	19.49	70	277	3	25.51	22.15
	3000	84	329	4	12.98	0.57	67	266	1	13.69	39.78	67	266	1	13.69	42.27
	4000	85	329	7	14.20	0.84	69	268	4	17.88	74.29	68	265	5	17.97	83.23
	5000	87	337	4	12.07	1.13	73	271	11	11.14	134.86	71	268	11	11.20	141.48
	6000	92	348	18	17.45	1.50	71	282	2	16.31	203.11	71	282	2	16.31	219.25
	7000	89	345	9	21.37	1.94	72	282	3	22.34	266.79	71	278	6	21.41	301.13
	8000	89	350	4	15.43	2.38	72	286	1	12.71	412.79	70	282	2	13.73	410.88
	9000	91	355	5	23.57	2.74	73	287	2	23.97	538.58	72	286	1	23.60	500.10
	10 000	92	357	6	9.88	3.32	72	286	1	9.92	675.11	71	286	1	10.12	689.23
5	2000	67	313	13	22.09	0.33	64	281	16	20.23	13.98	63	303	6	26.33	24.81
	3000	69	327	10	16.09	0.55	58	282	2	18.04	33.60	54	266	9	19.15	59.58
	4000	73	348	14	18.66	0.82	62	278	11	22.73	61.91	59	274	9	23.38	105.83
	5000	69	341	2	10.12	1.15	56	277	1	11.09	112.97	55	275	2	11.96	183.82
	6000	76	342	29	20.86	1.50	60	286	11	19.62	176.44	60	286	11	19.62	280.33
	7000	76	357	14	23.07	1.91	60	287	12	23.63	241.80	60	287	12	23.63	414.60
	8000	73	349	6	11.06	2.29	57	282	1	11.13	346.19	56	282	0	11.60	599.62
	9000	77	365	17	16.16	2.72	59	293	1	18.66	419.92	59	293	1	18.66	469.93
	10 000	72	354	3	14.32	3.29	58	284	3	11.19	562.71	56	283	2	11.06	652.36

Algorithm 4. The VND local search.

Input: $G=(V,E)$, initial solution S_0
Output: solution S

$k \leftarrow 1$;
$S \leftarrow S_0$;
while $(k<3)$ **do**
  if $(\exists S' \in \mathcal{N}_k(S) \mid S'$ *betterThan* $S)$ **then**
    $S \leftarrow \text{firstImproving}(\mathcal{N}_k(S))$;
    $k \leftarrow 1$;
  else
    $k \leftarrow k+1$;
  end
end
return S;

5. Computational experiments

Experiments were performed on a 3.00 GHz Intel Core Duo with 8 GB of RAM. The algorithms were developed in ANSI C. Two test sets have been generated to evaluate the metaheuristics. The first one contains 19 instances ranging from 100 to 1000 sensors (sinks included) randomly located in a $100 \times 100\ \mathrm{m}^2$ area with a 20 m radio range. Results are provided on this test set for 2–4 sinks. The second set contains 9 instances ranging from 2000 to 10 000 sensors (sinks included), randomly located in a $500 \times 500\ \mathrm{m}^2$ area with a 20 m radio range. Results are reported on this set for 2–5 sinks.

Results obtained using the Balanced ($A1 \rightarrow A3 \rightarrow B1$) and the Unbalanced ($A1 \rightarrow A2 \rightarrow B1$) optimization strategies are compared in terms of the maximum number of clusters per topology, the maximum deviation between the smallest and the biggest number of clusters considering the m-topology and the total number of clusters in the m-topology. The two proposed multi-start metaheuristics are also evaluated in terms of running time. The results produced by the Randomized Constructive Heuristic (RCH) and the two metaheuristics over a determined number of iterations are presented. The multi-start with the two-phases local search and the multi-start with the VND local search will be respectively referred as MS+2P and MS+VND.

The total number of iterations is used as stopping criterion. This has been set according to the results obtained with the first test set. In fact, 200 iterations seem to be a good compromise between the running time and the quality of the computed m-topology. Thus, 200 independent iterations using RCH and the metaheuristics have been performed on the two test sets.

Tables 1 and 2 present results using the first test set respectively for the Unbalanced and the Balanced strategy. Column m refers to the number of sinks and n to the number of sensors (sinks included). For RCH and the two multi-start metaheuristics, $A1$ stands for the maximum number of cluster per topology, $A2$

Table 7
Results for the Balanced ($A1 \rightarrow A3 \rightarrow B1$) strategy on the second test set.

m	n	RCH					MS+2P					MS+VND				
		$A1$	$A2$	$A3$	$B1$	$t(s)$	$A1$	$A2$	$A3$	$B1$	$t(s)$	$A1$	$A2$	$A3$	$B1$	$t(s)$
2	2000	153	304	2	18.03	0.35	128	255	1	19.37	27.43	128	255	1	19.37	27.39
	3000	164	328	0	19.90	0.57	129	258	0	24.28	51.51	128	255	1	22.25	52.24
	4000	168	335	1	18.59	0.86	130	260	0	18.64	93.19	130	260	0	18.47	95.52
	5000	169	332	6	20.23	1.20	134	268	0	19.04	151.47	133	266	0	19.03	157.24
	6000	169	336	2	14.40	1.56	135	270	0	13.58	231.02	135	270	0	13.58	239.73
	7000	172	342	2	18.93	1.92	137	274	0	18.66	317.57	135	270	0	18.10	331.73
	8000	176	349	3	23.01	2.44	140	280	0	24.76	416.94	139	278	0	20.03	435.78
	9000	175	346	4	13.80	2.96	138	276	0	14.03	564.97	138	276	0	14.03	583.56
	10 000	175	350	0	12.00	3.69	139	278	0	12.20	1104.68	138	276	0	13.15	747.02
3	2000	105	312	2	19.16	0.34	88	262	1	28.74	22.11	88	262	1	28.74	36.03
	3000	110	330	0	21.50	0.59	90	268	1	26.61	44.56	86	257	4	18.94	73.58
	4000	111	330	2	17.64	0.83	90	269	1	27.37	75.88	89	267	2	20.77	126.69
	5000	115	339	4	14.65	1.15	91	273	0	16.49	136.78	91	273	0	16.49	224.22
	6000	116	345	2	15.15	1.48	93	279	0	16.33	202.93	92	277	1	15.05	338.51
	7000	117	347	4	18.52	1.88	93	279	0	21.28	276.33	92	280	0	21.43	487.46
	8000	118	354	0	17.59	2.32	93	279	0	15.91	383.61	93	285	0	15.30	653.52
	9000	119	356	1	15.54	2.81	94	281	1	15.16	501.25	93	280	1	13.80	856.20
	10 000	119	356	1	11.88	3.33	94	282	0	10.65	660.54	94	282	0	9.83	1108.75
4	2000	80	319	1	16.41	0.33	71	280	2	25.51	20.81	70	277	3	25.51	30.08
	3000	84	331	3	14.10	0.55	67	266	1	13.69	38.89	67	266	1	13.69	61.03
	4000	85	329	7	14.20	0.84	69	272	2	17.85	76.17	68	275	1	19.12	117.50
	5000	87	337	4	12.07	1.18	73	276	8	12.65	133.58	71	268	11	11.20	208.97
	6000	92	348	18	17.45	1.51	71	282	2	16.31	205.62	71	283	1	14.97	319.22
	7000	89	345	9	21.37	1.89	72	282	3	22.34	273.47	71	278	6	21.41	429.34
	8000	89	350	4	15.43	2.34	72	286	1	12.71	384.52	70	283	1	13.33	628.42
	9000	91	355	5	23.57	2.93	73	287	2	23.97	491.63	72	286	1	23.60	758.12
	10 000	92	358	5	10.90	3.48	72	286	1	9.92	637.42	71	286	1	10.12	1049.47
5	2000	67	313	13	22.09	0.34	64	294	15	27.77	14.34	63	303	6	26.33	15.16
	3000	69	330	6	14.31	0.56	58	282	2	18.04	34.32	54	266	9	19.15	38.30
	4000	73	348	14	18.66	0.82	62	306	2	19.38	62.92	59	275	9	22.53	69.22
	5000	69	341	2	10.12	1.13	56	277	1	11.09	112.36	55	275	2	11.96	135.74
	6000	76	342	29	20.86	1.47	60	286	11	19.62	169.72	60	286	11	19.62	197.92
	7000	76	357	14	23.07	1.85	60	287	12	23.63	239.57	60	287	12	23.63	281.62
	8000	73	356	4	11.71	2.29	57	282	1	11.13	346.76	56	282	0	11.60	394.77
	9000	77	371	6	14.46	2.70	59	293	1	18.66	421.77	59	296	1	14.97	470.11
	10 000	72	354	3	14.32	3.25	58	289	1	13.50	563.51	56	287	1	13.11	655.44

presents the total number of clusters in the m-topology and $A3$ gives the deviation between the biggest and the smallest number of clusters in the m-topology. $B1$ corresponds to the maximum average hop per topology. The running time in seconds to perform the 200 iterations is reported in column $t(s)$. For the first test set, RCH and MS+2P are really efficient and they perform 200 iterations in less than 1 s. On the other hand, MS+VND is slower for the first test set.

For the sake of clarity, a/b refers here to a instances out of b. The Unbalanced strategy does not explicitly use a balance criterion. Thus, deviations are sometimes worse, while the maximum number of clusters per topology or the number total of clusters is improved. This happens for the instance with 3 sinks and 400 nodes where $A3$ increases while $A2$ decreases. Although the Unbalanced strategy implicitly provides balanced m-topologies for 18/19, 17/19, 17/19 respectively using 2, 3 and 4 sinks ($A3$ equals to 0 or 1). The Balanced strategy produces 19/19, 19/19 and 19/19 strictly balanced m-topologies ($A3$ equals 0) for the three methods. In addition, RCH and the multi-start metaheuristics using the Balanced strategy provide very good results for the $A3$ measure, in spite of some increase on $A1$ or $A2$ (which does not exceed 3 clusters for $A2$) over the results from the Unbalanced strategy, see results in Tables 1 and 2. This is an expected behavior since the Balanced strategy explicitly uses an optimization criterion for this purpose. In most of the cases, the total number of clusters $A2$ is higher using the Balanced strategy.

A summary of the comparisons between methods, aggregated on m, is shown in Tables 3–5. In Table 3, the first two columns report the two methods compared and the number m of sinks. Considering the Unbalanced or Balanced strategies, comparisons on the final solutions are done on $A1$ criterion only or on the full lexicographic order. Better (better), similar (similar) and worse (worse) report how many times the first method provides respectively better, similar or worse solutions than the second one. Comparing the maximum number of clusters per topology $A1$ using the Unbalanced strategy, MS+2P and MS+VND produce better topologies than RCH for 17/19, 9/19 and 8/19, respectively with 2, 3 and 4 sinks. RCH is able to compute the best solution for 2/19, 10/19 and 11/19 if only $A1$ is considered. MS+2P and MS+VND produce similar solutions on the $A1$ criterion. When considering all the criteria in lexicographic order, MS+2P and MS+VND compute better solution for 19/19, 18/19 and 18/19, while RCH is able to find the best solution for 0/19, 1/19 and 1/19.

Results for the Balanced strategy (Table 3) are quite similar to the Unbalanced strategy. MS+2P and MS+VND have also the same level of performance, especially when considering only $A1$. MS+VND is able to find a few better solutions than MS+2P when considering the lexicographic order. The difference happens on the third criterion ($B1$), see Table 2. One can note that RCH is able to find better results in the Balanced strategy than in the Unbalanced strategy.

Table 4 provides a summary of the balancing ($A3$ value) for the first test set, considering only MS+2P and MS+VND. Results show

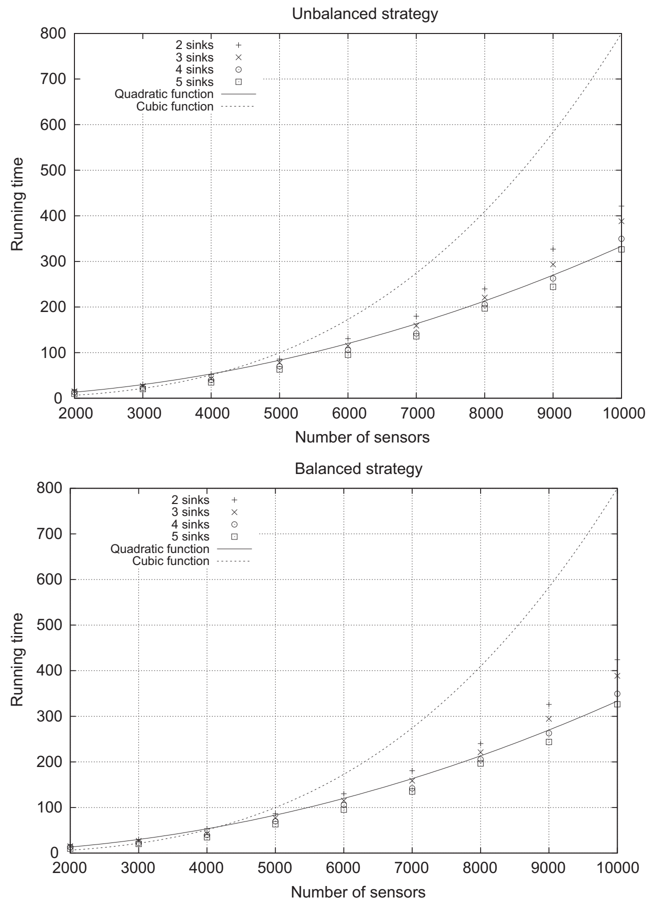

Fig. 6. Evolution of MS+2P running time for the second test set.

Table 8
Methods comparison on the second test set.

Methods	m	$A1$ only			$A1 \rightarrow A2 \rightarrow B1$		
		Better	Similar	Worse	Better	Similar	Worse
Unbalanced strategy							
RCH *vs.* MS+2P	2	0	0	9	0	0	9
	3	0	0	9	0	0	9
	4	0	0	9	0	0	9
	5	0	0	9	0	0	9
RCH *vs.* MS+VND	2	0	0	9	0	0	9
	3	0	0	9	0	0	9
	4	0	0	9	0	0	9
	5	0	0	9	0	0	9
MS+2P *vs.* MS+VND	2	0	4	5	0	1	8
	3	0	4	5	0	1	8
	4	0	2	7	0	2	7
	5	0	3	6	0	3	6
Balanced strategy							
RCH *vs.* MS+2P	2	0	0	9	0	0	9
	3	0	0	9	0	0	9
	4	0	0	9	0	0	9
	5	0	0	9	0	0	9
RCH *vs.* MS+VND	2	0	0	9	0	0	9
	3	0	0	9	0	0	9
	4	0	0	9	0	0	9
	5	0	0	9	0	0	9
MS+2P *vs.* MS+VND	2	0	4	5	0	3	6
	3	0	4	5	0	2	7
	4	0	2	7	0	1	8
	5	0	3	6	0	2	7

Table 9
Overview of the summation over $A3$ on the m-topology on the second test set.

Method	m	Unbalanced	Balanced
MS+2P	2	5	1
	3	11	4
	4	27	22
	5	58	46
MS+VND	2	5	2
	3	14	9
	4	32	26
	5	52	51

how difficult is for the Unbalanced strategy to get m-topologies with no deviation when the number of sinks increases. Table 5 refers to the $A2$ measure. While the Balanced strategy produced balanced m-topologies for all instances, Unbalanced produces m-topologies with a smaller total number of clusters. This probably indicates a compromise between $A2$ and $A3$.

Tables 6 and 7 report the results for the Unbalanced and Balanced strategies respectively on the second test set. The meaning of the columns is similar to Tables 1 and 2. The settings for generating this test set of instances differ, which explains why results cannot be directly compared with results from the first test set. However, the CPU time remains small for the RCH (usually less than 2 s), while it increases for the two metaheuristics. This is a direct consequence of the local searches time complexity. Nevertheless, the running time does not exceed 20 min to perform 200 iterations even for the largest instances with 10 000 sensors.

For both Tables 6 and 7, the two metaheuristics give similar results with respect to criterion $A1$, *i.e.* maximum number of cluster per topology. The difference between the Unbalanced and the Balanced strategies happens for $A2$ and $A3$. However, each time Balanced is worse than Unbalanced in terms of $A3$, it is better in terms of $A2$. This result indicates when the number of sensors is about 4000, the criterion $A3$ is not sufficient to get the topology balanced. Thus, new strategies can be further investigated for such cases.

Fig. 6 reports the evolution of MS+2P running time with respect to the number of sensors and the number of sinks. The time consumption seems to grow nearly quadratically on the number of sensors. MS+VNS CPU consumption also grows quadratically. Moreover, the CPU time slightly reduces when the number of sinks increases. One reason is that each topology has fewer clusters and on average topologies are more balanced. Thus, there are less move opportunities, which reduces the amount of work in the local search.

A summary of the comparison between the methods is given in Table 8. The meaning of each column is similar to Table 3. On the contrary to the first test set, the second test set amplifies the differences between the methods. The improvement done by MS+2P and MS+VND over RCH, especially with respect to $A1$, is significant. The smaller the m is, the larger the difference is. Moreover, MS+VND clearly produces better solutions than MS+2P and the difference in terms of computational time has decreased. As a consequence, MS+2P seems to be the best compromise for small instances (up to 1000 nodes) since MS+2P and MS+VND are equivalent in terms of solution quality but MS+2P is faster. On the other hand, MS+VND becomes more attractive for larger instances since it produces better solutions and the difference in CPU time decreases.

Results also show balancing the m-topologies becomes critical when the number of sinks increases. A summary of the impact of Unbalanced and Balanced strategies on A3 is given in Table 9. It shows the Balanced strategy still produces better balanced solutions.

6. Concluding remarks and perspectives

In this work, the m-IDSC problem has been investigated using two optimization strategies which combine two levels of decision: assigning sensors to a sink and balancing the m-topology.

A constructive randomized heuristic and two multi-start metaheuristics have been developed, one coupled with a two-phase local search and another with a VND. Results show the efficiency of the proposed strategies. The running time remains affordable in the context of WSNs with up to 1000 sensors. The Unbalanced optimization strategy implicitly produces some balanced m-topologies and it leads to solutions with a small total number of clusters. On the other hand, the Balanced optimization strategy produces strictly balanced m-topologies with a slightly higher total number of clusters. Such results indicate a compromise between the optimization goals needs to be established and highlights the complexity of the problem.

To the best of our knowledge, the proposed strategies are the first ones in the literature for the m-IDSC. A genetic algorithm has already been proposed for the LBCDS which does not deal with bridges in He et al. (2012). The multi-start metaheuristics are quite simple and requires few parameters. Moreover, it consumes less than one second to compute a m-topology for instances with up to 1000 nodes. As a consequence, a new topology can be quickly computed whenever a region becomes unreachable in a repair strategy. The low CPU requirement and the quality of the solutions produced make our approach scalable and suitable for applications in a centralized approach to design cluster-based topologies on-the-fly. It can be used to compute the initial topologies as well as a new topology in case of node failure. Besides, using clusters allows the network lifetime to be improved since it benefits from the clustered structure. Among the proposed methods, the MS+2P is better suited for instances with up to 1000 nodes, while MS+VND becomes a valuable alternative beyond 1000 nodes.

As future works, the model can be extended to couple the location problem (positioning sinks on the area) and the m-IDSC problem in the context of WSN, and also to deal with coverage over a set of targets. Moreover, other optimization strategies can be investigated. In terms of algorithms, other metaheuristics can be designed for the m-IDSC such as genetic algorithms. We are actually developing a multiobjective-based metaheuristics for m-IDSC.

References

Abbasi, A.A., Younis, M., 2007. A survey on clustering algorithms for wireless sensor networks. Comput. Commun. 30, 2826–2841.

Aioffi, W.M., Valle, C.A., Mateus, G.R., Cunha, A.S., 2011. Balancing message delivery latency and network lifetime through an integrated model for clustering and routing in wireless sensor networks. Comput. Netw. 55, 2803–2820.

Amendment of IEEE Std 802.15.4, 2007. Approved draft amendment to IEEE standard for information technology-telecommunications and information exchange between systems-part 15.4: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs): Amendment to add alternate PHY. IEEE Approved Std P802.15.4a/D7.

Castaño, F., Rossi, A., Sevaux, M., Velasco, N., 2013. On the use of multiple sinks to extend the lifetime in connected wireless sensor networks. Electron. Notes Discr. Math. 41 (0), 77–84.

Cipollone, E., Cuomo, F., Della Luna, S., Monaco, U., Vacirca, F., 2007. Topology characterization and performance analysis of IEEE 802.15.4 multi-sink wireless sensor networks. In: Proceedings of the 6th Annual Mediterranean Ad Hoc Networking Workshop, Corfu, Greece. pp. 196–203.

Clark, B.N., Colbourn, C.J., Johnson, D.J., 1991. Unit disk graphs. Discr. Math. 86, 165–177.

Elhabyan, R.S.Y., Yagoub, M.C.E., 2015. Two-tier particle swarm optimization protocol for clustering and routing in wireless sensor network. J. Netw. Comput. Appl. 52 (C), 116–128.

Ferentinos, K.P., Tsiligiridis, T.A., Arvanitis, K.G., 2005. Energy optimization of wireless sensor networks for environmental measurements. In: IEEE International Conference on Computational Intelligence for Measurement Systems and Applications CIMSA, pp. 250–255.

Forgy, E.W., 1965. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. Biometrics 21, 768–769.

Friedmann, L., Boukhatem, L., 2007. Efficient multi-sink relocation in wireless sensor network. In: Third International Conference on Networking and Services ICNS, pp. 90–95.

He, J., Ji, S., Yan, M., Pan, Y., Li, Y., 2012. Load-balanced CDS construction in wireless sensor networks via genetic algorithm. Int. J. Sens. Netw. 11 (3), 166–178.

Ibrahim, S., Cui, J.-H., Ammar, R., 2008. Efficient surface gateway deployment for underwater sensor networks. In: IEEE Symposium on Computers and Communications ISCC, pp. 1177–1182.

Ji, S., He, J.S., Cai, Z., 2014. Data gathering in wireless sensor networks. In: Ammari, Habib M. (Ed.), The Art of Wireless Sensor Networks Signals and Communication Technology. Springer, Berlin, Heidelberg, pp. 535–565.

Kim, H., Seok, Y., Choi, N., Choi, Y., Kwon, T., 2005. Optimal multi-sink positioning and energy-efficient routing in wireless sensor networks. In: Cheeha Kim (Ed.), Information Networking. Convergence in Broadband and Mobile Networking. Lecture Notes in Computer Science, vol. 3391. Springer, Berlin/Heidelberg, pp. 264–274.

Kuila, P., Gupta, S.K., Jana, P.K., 2013. A novel evolutionary approach for load balanced clustering problem for wireless sensor networks. Swarm Evolut. Comput. 12, 48–56.

Kuila, P., Jana, P.K., 2014. Energy efficient clustering and routing algorithms for wireless sensor networks: particle swarm optimization approach. Eng. Appl. Artif. Intell. 33 (0), 127–140.

Lian, J., Naik, K., Agnew, G., 2006. Data capacity improvement of wireless sensor networks using non-uniform sensor distribution. Int. J. Distrib. Sensor Netw. 2 (2), 121–145.

Lin, H., Üster, H., 2014. Exact and heuristic algorithms for data-gathering cluster-based wireless sensor network design problem. IEEE/ACM Trans. Netw. 22 (3), 903–916.

Martí, R., 2003. Multi-start methods. In: Fred Glover, Gary Kochenberger (Eds.), Handbook of Metaheuristics. International Series in Operations Research & Management Science. Vol. 57. Springer, New York, pp. 355–368.

Mladenovic, N., Hansen, P., 1997. Variable neighborhood search. Comput. Oper. Res. 24 (11), 1097–1100.

Saad, L.B., Tourancheau, B., 2009. Towards an optimal positioning of multiple mobile sinks in WSNs for buildings. Int. J. Adv. Intell. Syst. IARIA 2 (4), 411–421.

Santos, A.C., Bendali, F., Mailfert, J., Duhamel, C., Hou, K.-M., 2009. Heuristics for designing energy-efficient wireless sensor network topologies. J. Netw. 4 (6), 436–444.

Santos, A.C., Duhamel, C., Belisário, L.S., Guedes, L.M., 2012. Strategies for designing energy-efficient clusters-based WSN topologies. J. Heuristics 18, 657–675.

Sitanayah, L., Brown, K.N., Sreenan, C.J., 2015. Planning the deployment of multiple sinks and relays in wireless sensor networks. J. Heuristics 21 (2), 197–232.

Tacconi, D., Miorandi, D., Carreras, I., Chiti, F., Fantacci, R., 2010. Using wireless sensor networks to support intelligent transportation systems. Ad Hoc Netw. 8 (5), 462–473.

Tubaishat, M., Zhuang, P., Qi, Q., Shang, Y., 2009. Wireless sensor networks in intelligent transportation systems. Wirel. Commun. Mobile Comput. 9 (3), 287–302.

Vincze, Z., Vida, R., Vidacs, A., 2007. Deploying multiple sinks in multi-hop wireless sensor networks. In: IEEE International Conference on Pervasive Services, July, pp. 55–63.

Vlajic, N., Xia, D., 2006. Wireless sensor networks: to cluster or not to cluster? In: Proceedings of the International Symposium on a World of Wireless, Mobile and Multimedia Networks. IEEE Computer Society, Buffalo, New York, pp. 258–268.

Wichmann, A., Korkmaz, T., 2015. Smooth path construction and adjustment for multiple mobile sinks in wireless sensor networks. Comput. Commun. 72, 93–106.

Wu, X., Chen, G., Das, S.K., 2008. Avoiding energy holes in wireless sensor networks with nonuniform node distribution. IEEE Trans. Parallel Distrib. Syst. 19 (5), 710–720.

Zhao, M., Yang, Y., Wang, C., 2015. Mobile data gathering with load balanced clustering and dual data uploading in wireless sensor networks. IEEE Trans. Mobile Comput. 14 (4), 770–785.