Koç and Korpeoglu *EURASIP Journal on Wireless Communications and Networking* (2015) 2015:245
DOI 10.1186/s13638-015-0472-5

EURASIP Journal on
Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

Coordinated movement of multiple mobile sinks in a wireless sensor network for improved lifetime

Metin Koç* and Ibrahim Korpeoglu

Abstract

Sink mobility is one of the most effective solutions for improving lifetime and has been widely investigated for the last decade. Algorithms for single-sink mobility are not directly applied to the multiple-sink case due to the latter's specific challenges. Most of the approaches proposed in the literature use mathematical programming techniques to solve the multiple-sink mobility problem. However, doing so leads to higher complexities when traffic flow information for any possible sink-site combinations is included in the model. In this paper, we propose two algorithms that do not consider all possible sink-site combinations to determine migration points. We first present a centralized movement algorithm that uses an energy-cost matrix for a user-defined threshold number of combinations to coordinate multiple-sink movement. We also give a distributed algorithm that does not use any prior network information and has a low message exchange overhead. Our simulations show that the centralized algorithm gives better network lifetime performance compared to previously proposed MinDiff-RE, random movement, and static-sink algorithms. Our distributed algorithm has a lower network lifetime than centralized algorithms; sinks travel significantly less than in all the other schemes.

Keywords: Wireless sensor network, Energy-efficiency, Multiple-sink mobility, Network lifetime improvement

1 Introduction

Moving sink nodes (base stations or mobile agents) is among the most effective solutions to improve network lifetime in wireless sensor networks (WSNs) where sinks can move [1]. This approach aims to solve the so called *hot-spot* problem in a WSN, in which first-hop neighbors of sinks suffer from quick energy depletion due to a high rate of message forwarding [2]. In order to distribute traffic forwarding load as evenly as possible among all sensor nodes, sinks can be moved through the region so that all sensor nodes have nearly equal chance of being the first-hop neighbors of sinks nodes. In this way, data is collected in a mobile manner.

In the multiple-sink mobility problem, s sinks should choose different sites among p alternatives in a mutually exclusive manner so that any two sinks do not decide to move to the same location at the same re-configuration instance. In the single-sink case, the base station chooses one point out of p alternatives, but in the multiple-sink case, the number of possible combinations to select in each decision is $\binom{p}{s}$. These make the multiple-sink more complex compared to the single-sink problem.

Although they should not be used in a real sensor node environment due to computational constraints, mathematical programming-based solutions are mostly preferred, as in the single-sink case, for maximizing network lifetime using multiple-sink movements. Additionally, an exponential number of constraints can exist in these formulations, since all possible sink-site combinations are included. Solving these formulations can take several days even with current computation technology. These calculations must be done before the network begins operation, and the solutions (sink sojourn time for each sink site) should be given to the sinks. Obviously, these solutions are not dynamic and fail to adapt to changes (such as unavailable sites or node failures due to physical conditions) that occur in the area over the network lifetime.

*Correspondence: mkoc@cs.bilkent.edu.tr
Department of Computer Engineering, Bilkent University, Ankara, Turkey

In this paper, we present two low-complexity algorithms for multiple-sink mobility problem, multiple-sink movement algorithm (MSMA) and prevent and move away (PMA) algorithm. Our algorithms have low complexity and low overhead and therefore can be used directly with sensor nodes and mobile sink nodes.

Our MSMA algorithm runs in each sink node and relies on the energy consumption information of each sensor node for a subset of all possible sink-site combinations. Most algorithms in literature use traffic-load information for each possible sink-site combination as part of their solutions. Since the number of sink-site combination alternatives increases exponentially, it becomes impractical to calculate traffic-load for every sink-site combination for a large number of sites and sinks. In our MSMA algorithm, we address this problem, and we do not require enumeration of all possible sink-site combinations. Our algorithm uses a threshold and limits the number of combinations to consider for movement of sink nodes.

Our PMA algorithm is a totally distributed algorithm and does not require any prior information about the network and possible traffic-load of nodes for specific sink-site combinations. Each sink checks the residual energy values of its descendant nodes and *forbids* placement in sites nearest to these low-energy nodes. This information, as well as the minimum energy nodes available within the sites' first-hop neighbors, are shared among sinks. Then, half the sites are selected according to these values. The other half of the sites are chosen to be far away from the previously selected sites so that sink nodes are not clustered to a particular sub-region in the area and the load is balanced as much as possible without collecting and using global network information.

The rest of the paper is organized as follows. In the next section, we discuss the related work about the multiple-sink problem. We describe the system model we used in Section 3. We present our proposed algorithms in Section 4, and we give the performance evaluation of them in Section 5. Finally, we give our conclusions in Section 6.

2 Related work

Different energy-efficient approaches in various network layers are proposed in the literature for improving network lifetime, such as power control mechanisms (physical layer) [3, 4], energy-efficient MAC layer protocols (data-link layer) [5, 6], data-gathering/routing protocols (network layer) [7–16]. Sink mobility is another approach for improving network lifetime. Sink mobility can be either *uncontrolled* or *controlled*, depending on the mobility scheme used [17]. In uncontrolled mobility, data MULEs (Mobile Ubiquitous Lan Extensions) move randomly in the area without considering any network parameter (such as nodes' remaining energy). In controlled mobility, however, network conditions (nodes' remaining energy, node density in the region, etc.) are taken into consideration. The controlled single-sink mobility problem is widely investigated in the literature [18–24]. Although it has not been investigated as much as its single-sink counterpart, there are some studies about the multiple-sink mobility problem.

Gandham et al. [25] presents one of the earliest works about the multiple-sink mobility problem. The authors divide the sensor network lifetime into equal periods of time, called rounds, and relocate the sinks at the start of each round. They present an integer linear program (ILP) that minimizes the maximum energy spent in each round in order to determine the locations of the sinks. A variation of this model minimizes the total energy consumption in a round. They compare these two ILP formulations as well as the random and static-sink approaches. Minimizing the maximum energy spent increases network lifetime significantly compared to other approaches. Although authors give ILP formulations to find the optimal lifetime, the number of base stations is fixed to three. The solution is not scalable to a higher number of base stations, since the number of constraints increase exponentially.

Azad and Chockalingam [26] also deals with the multiple-sink mobility problem. The authors choose feasible sink-site locations along the periphery of the field and propose three heuristics, Top-K_{max}, Max-Min-RE, and MinDiff-RE, to determine the sojourn points of the sinks in each round. In Top-K_{max}, K sites are chosen, in which the nearest neighbor nodes have maximum residual energies. The Max-Min-RE heuristic aims to maximize the minimum residual energy for all combinations of sink sites and sinks for a given routing algorithm. The MinDiff-RE method uses the same approach as Max-Min-RE, except the goal is to minimize the difference in residual energy. Experiments show that MinDiff-RE gives better network lifetime compared to other approaches. Although these heuristics are presented as energy-efficient and low-complexity algorithms, Max-Min-RE and MinDiff-RE need to process all combinations before selecting the next migration points. These methods can be practical for small values of number of sinks and sites. However, in typical scenarios, for instance for 30 sink sites and 8 sink nodes, there may be millions of sink-site combinations to consider, which make these solutions impractical.

Basagni et al. [27] proposes a linear programming (LP) model as well as centralized and heuristic algorithms for the multiple-sink mobility problem. Any combination of s sinks occupying s locations among k sites is called a *configuration*. The authors define an LP model with the goal to find a sequence of configurations that maximizes network lifetime. Although the LP has an exponential number of constraints, the authors use a separation algorithm to resolve the LP a polynomial number of times.

They also give a centralized heuristic that uses the output of the LP and runs in polynomial time. Finally, authors define a deployable distributed heuristic for coordinating the motion of multiple sinks throughout the network. The simulation results show that the proposed schemes improve lifetime significantly compared to the cases of a static-sink case and random mobility. Authors both give an efficient method for solving LP and a centralized heuristic for solving the multiple-sink problem. These methods enable them to make experiments of up to 8 sinks (for 16 sink sites). However, the proposed distributed heuristic requires a large amount of data exchange between the sink nodes. This can cause delay while determining the next migration point which makes it disadvantageous for delay intolerant applications

Liang and Luo [28] formulates the network lifetime maximization for the h-hop constrained multiple-sink mobility problem such that the total travel distance of each sink is bounded by L, and the maximum number of hops from each sensor to a sink is bounded by h, where $h \geq 1$. The authors show that the problem is NP-hard and propose a three-stage heuristic for calculating the sojourn time at each location for K mobile sinks. In the first stage, the sojourn time profile at each h-feasible configuration is determined using a linear program. Then, optimal trajectories for each mobile sink are built via greedily adding h-feasible configuration if the *benefit* to the network lifetime is maximized. Finally, the exact sojourn time in each configuration is calculated using another linear program. The experimental results show that the proposed algorithm performs at up to 93 % of the optimal solution. Although, the authors give a strong heuristic for solving the problem, its complexity, $O(m\, n\, k\, log\, n + n^2k + k^3)$ (m number of edges, n number of nodes, k number of sinks), makes it infeasible to be used dynamically in real sensor nodes.

In [29], the authors propose a multi-sink relocation algorithm with several components. They adopt a centralized approach for the routing methodology. Link weights are computed using a cost function, and finally, a mathematical model is given for determining optimal sink positions, i.e., network configuration. The authors use a local search algorithm for determining local optima in the solution space. The experimental results are presented around emphasizing mobility utilization compared to stationary sinks. Authors give an efficient mechanism to place the sinks; however, they do not consider mobility and sojourn times of the sinks for each possible sink location. This makes it usable for the cases where the sinks are placed once and do not change their positions.

3 System model

We consider a sensor network with n stationary sensor nodes deployed to an area of interest. The set of sensor nodes is denoted with $|N| = n$. There are s mobile sink nodes in the environment, which may change their locations periodically according to network conditions. Sinks choose these locations from a predetermined sojourn point set P, where $|P| = p$. Each sensor $i \in N$ has a packet generation rate of Q packets per second.

Tree-based routing is used such that each sensor node sends its own packet and relays its children's packets to the same parent for a given sink-site combination. Sinks learn the related parameters in a *training* phase so that the same routing tree is constructed in each visit to the same site combination. This method enables us to construct an energy-load matrix structure T of sink-site combinations (rows) and nodes (columns), such that each entry t_{ij} in the

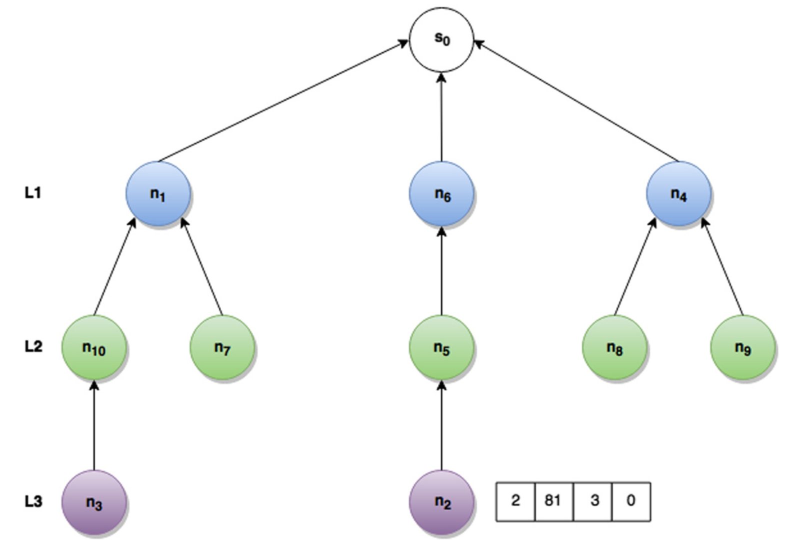


Fig. 1 Sample topology and broadcast packet

matrix lists how much energy a node *j* would consume to transmit its packets to its parent in the routing tree for a given sink-site combination *i*.

Sink sites (sojourn points) are either predetermined or chosen using a sink-site determination method. Some example methods are given in [30]. Those sink sites are associated with each sink using a method that we explain in the next section. Sinks visit those sites (combinations) together and start the topology tree formation process to construct their matrices if the sensor nodes' positions are not known a priori.

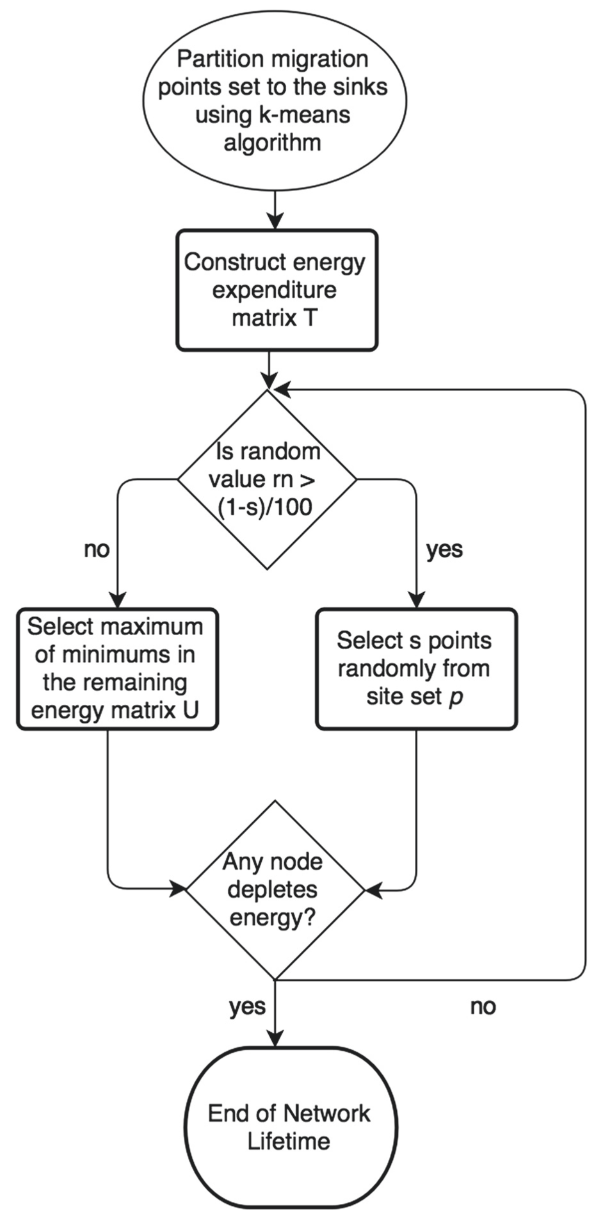
In this phase, each sink broadcasts a packet to construct a tree rooted at that sink. This packet contains the sender ID (the node ID that broadcasts the packet), topology ID (indicates the current sink combination - unique for each combination), hop count (0 for sink, and incremented by one for each received node), and sink ID (the node that initiated the topology). An example of this packet structure is given in Fig. 1. This tree-topology is constructed by sink with ID 0 (s_0), when sinks sojourn at certain positions specified by combination 81. In this example, node 2, n_2, constructs a packet with its ID 2, combination ID 81, its logical level 3, and its sink ID 0.

The sinks' first-hop neighbor receiving those packets set the sink ID as their parent ID. These packets are re-broadcast after incrementing the hop count (count to the sink). When a sensor node receives such a packet, it first checks its current hop count to the sink; if it is not set or is bigger than that of the received message, it updates its hop count and sets the sender ID as its new parent ID. This process continues until all nodes in the area have received these topology set-up packets. When the tree has been set up, each sensor node saves the topology ID and parent ID pair to remember which node to send to when sinks select the same combination at any time over the network lifetime.

When a node receives a packet from a different sink, it saves its ID and hop-count to the sink even if this hop-count is bigger than its current value. It becomes a *junction* node and it informs all the sinks it receives packets from. These junction nodes can be used for two different purposes: sink communication and topology maintenance.

If junction nodes are known by the sinks, then those sinks can reach each other using the paths through junction nodes. Although sinks have higher transmission power than sensor nodes do, sometimes they can be far enough from each other that direct communication cannot be achieved. In these cases, using those junction nodes help sinks exchange information for the next site selection decision or for remaining energy values.

In delay-intolerant applications, it is always critical for sensor nodes to send their data to an available sink (any-


Partition migration points set to the sinks using k-means algorithm
Construct energy expenditure matrix T
Is random value rn > (1-s)/100
no
yes
Select maximum of minimums in the remaining energy matrix U
Select s points randomly from site set *p*
Any node depletes energy?
yes
no
End of Network Lifetime
Fig. 2 Flowchart of MSMA

cast) instead of buffering it. Multiple sinks do not move just exactly at the same time since at least one of the sinks must stay to gather the sensor nodes' data. When a sink decides to move, it informs its neighbor nodes and sends a re-connect request packet to this junction node indicating that the junction node should connect to another network. The sink's first-hop neighbors change their parents toward the junction node. For the subtree that junction node belongs to, its ancestor, which is one of the sink's first-hop neighbors, reverses the packets' direction. For other subtrees, rooted at the sink's other first-hop neighbors, these nodes forward their packets to the junction node's ancestor, until a new topology construction packet with a smaller hop-count value is received by the nodes, which means that a sink arrives at its new location. There are some works about mobile dynamic tree construction and maintenance, such as [31–33]; however, this topic is beyond the scope of this paper.

4 Proposed algorithms

As mentioned in the related work section, most studies about multiple-sink mobility in the literature use linear/integer programming formulations to achieve optimal values of network lifetime.

These methods, however, cannot be used by sink nodes on-the-fly during network operation, since they require a huge amount of computation which sink nodes cannot achieve with the current technology. In this chapter, we want to present algorithms that do not require too much complexity, but still achieve considerable lifetime improvement compared to random movement and static-sink cases.

4.1 Multiple sinks movement algorithm (MSMA)

Our first algorithm to coordinate the movement of multiple mobile sinks for improving network lifetime is called MSMA. The algorithm extends our earlier algorithm MS-ELMA [34]. We improve the method of limiting the possible position combinations before the network begins operation. We also change the algorithm to add limited combinations on-the-fly to further improve its performance.

Algorithms proposed to solve the multiple-sink mobility problem usually consider the amount of traffic (directly or indirectly) between nodes when sinks are placed at certain locations/sites [26, 27]. Assuming that each different sink placement is a sink-site combination, a set of sinks has many possibilities to get placed into these locations. When sinks are placed according to a combination k and routing trees are formed from sinks to nodes, the traffic from a node i to its parent node j can be denoted with x_{ijk}. Knowing x_{ijk} values for all possible site combinations as well as the current remaining energy values of nodes help us to select the best alternative site combination to move because we can see how the nodes' energy values will be affected if sinks would move to a specific site combination. However, calculating x_{ijk} values for all possible site combinations before the network starts its operation can take a very long time, which is not feasible for reasonable p and s values, such as 50 sink sites and 7 sinks.

Since choosing s points among p alternatives increases exponentially, the main idea should be to decrease those alternatives as much as possible to reduce complexity without sacrificing lifetime improvement. For instance, choosing seven locations among 50 alternatives generates approximately 10^8 alternatives. In our algorithm, we require all sinks to make a decision about their next migration points using their information, which are further evaluated, and a final decision is made collectively.

Each sink uses an energy-expenditure matrix in which each element t_{ij} corresponds to the energy required at a node j to transmit packets to its parent when the sink nodes are in position-combination i, i.e., the combination tuple - $(p_1, p_2, \ldots p_s)$. The number of columns of this matrix apparently equals the number of nodes n in the network. The number of rows, i.e., the number of position combinations, is, however, a parameter in our algorithms, which can be determined using sink characteristics. This is called *binomial threshold* and is not directly involved in our algorithms, instead affects the th_s parameter (threshold for each sink), which determines how many migration points each sink will select to calculate different combinations. In other words, th_s is the maximum integer satisfying $\binom{th_s}{s} \leq t_b$, where s is the number of sinks and t_b is the binomial threshold. If we select t_b as 2000, then th_s becomes 13, when the number of sinks is eight, since $\binom{13}{8} = 1287$.

After determining these parameters, we select the possible migration points for each sink. There will be a total of s groups of points. For a sink, we assign possible migration points from different regions of the area. First, we determine the number of points (k) to assign for a sink. It is given by $k = p/s$. Then, we start using the *k-means* algorithm to select migration points for sinks. For the first sink, we run the k-means algorithm considering the whole migration point set. Given the whole point set (p points initially), the k-means algorithm selects k centroids (positions), and the k migration points with minimum distance to these k centroids are assigned to the first sink. We remove these selected positions from our point set, and we continue to select points for the next sink by re-running the k-means algorithm to select another k centroid from the remaining point set. This process continues until all points are partitioned among the sinks. If the allocated number of points is smaller than the predetermined threshold value, then new points are assigned randomly to close this gap.

After each sink has its group of sites, i.e., possible position-combinations to move, it will construct an energy-load matrix. For each candidate sink-site combination i, that means for each row, the algorithm computes how much energy the nodes would consume to transmit their packets to the next node (parent) in the routing tree for that combination i. If coordinates of the nodes are known a priori, these matrices can be constructed by a central authority, i.e., via one of the sinks. The other option is to visit those location combinations and form topology trees for each combination to determine the child-parent relationship and energy expenditure values to construct matrices in a training phase, as in [27], before

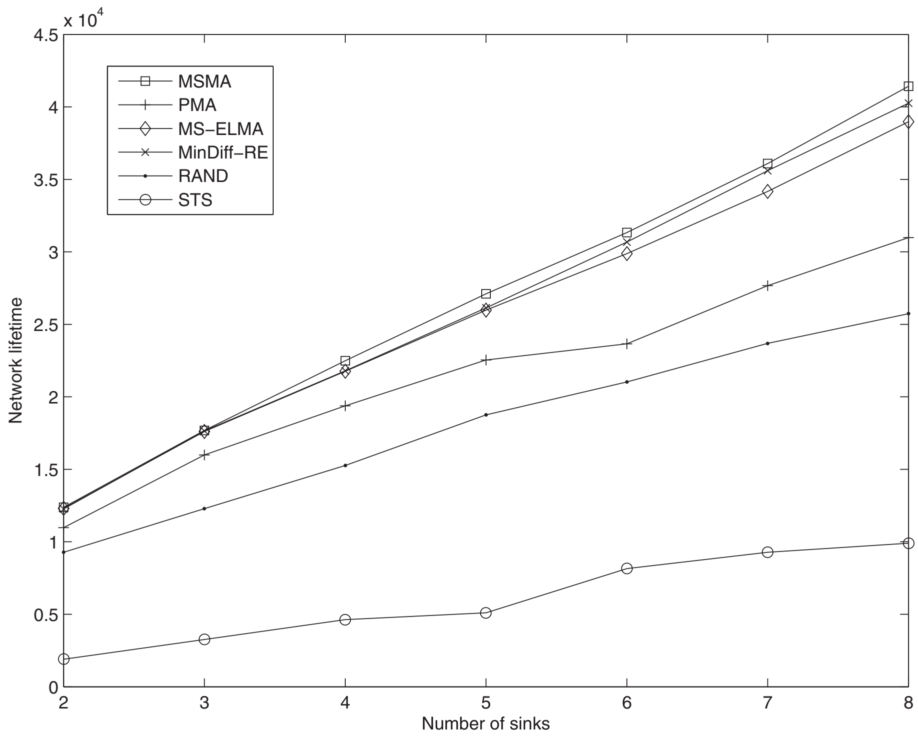


Fig. 3 Network lifetime values for various numbers of sinks

the network starts operation. Each sink keeps its own matrix that has $\binom{th_s}{s}$ rows and that includes combinations of the points previously assigned to it. Sinks use these matrices to take role in point selection process. Combination with best value is selected for next migration point collectively. Therefore, sinks are not limited with the points in their matrices and instead can go to any point in the area according to the solution given by the algorithm.

Sinks use the nodes' remaining energy values and the energy-load matrix to determine the next migration points. The sensor nodes' remaining energy values, e, can be represented as an nx1 matrix (a vector or a row). Sinks share their descendants' values with each other to make the row complete for all entities. Each matrix row indicating a sink-site combination is subtracted from this remaining energy row to project the nodes' remaining energy values if sinks would move to those locations in the next round. The sinks focus on the minimum values of each subtracted row and select the maximum of them. These values, i.e., the maximum remaining energy values and sink-site combinations, are sent to the sink with the minimum-ID through either single-hop or multi-hop communication. If multi-hop communication is needed, then junction nodes are utilized. A sink compares the received values with its own and sends the one with the maximum value to the other sinks. Last, the decisive sink compares all received values and picks the maximum. Then, the selected migration point information is sent back to the sinks in the reverse direction.

In the MSMA, sinks can also select migration points randomly but with a small probability ($s/100$). Since the algorithm starts with fewer combinations, it is a good idea to increase these combinations without exceeding the t_b value. But more importantly, since the topology is constructed at the selection, the algorithm should take advantage of the sensor nodes' current remaining energy information. Instead of using the topology constructed using full energy levels (prior to the network beginning operation), sinks can construct more energy-efficient topology trees using the nodes' current residual energies. Since sinks use this option rarely, selecting those sites randomly is a good choice because it decreases decision time and message overhead. The procedure is summarized in Fig. 2.

Although the next migration points have been determined, which sink is going to exactly which point has not been set. This decision can also made by the same sink, since it knows the current location of each sink. Sinks should inform other sinks about their locations to prevent two or more sinks going to the same location. This is the well-known *assignment problem*, and the Hungarian method can find the optimal solution for assigning the sinks to new locations with $O(n^3)$ complexity [35].

After sinks migrate to the new points, they will operate there until either a certain amount of change in the energy of its first-hop neighbors is detected or a fixed number of rounds - t_{min} - has been achieved. Then, when the sojourn time expires, sinks share their children's remaining energy values with each other and again run the max-min search over the matrix to determine new migration points. This iteration continues until a node depletes its energy. The algorithm is detailed in Algorithm 1. It takes $O(n\,t_b)$ to construct the energy load matrix. In each round, we need $O(t_b)$ computation to determine the sinks' next migration points.

4.2 Prevent and move away (PMA) algorithm

In this section, we describe our distributed prevent and move away (PMA) algorithm, which is fully distributed and does not require excessive information exchange. As explained above, the current schemes proposed in the literature consider the amount of traffic passing through each node to its parent while sinks are at specific sites. Although this approach improves network lifetime significantly, considering all possible sink-site combinations is computationally expensive. Therefore, it is important

Algorithm 1 Multiple-Sinks Movement Algorithm

```
1:  procedure MSMA($c, t_x,$)                          ▷ c: node coordinates, $t_x$: transmission range
2:      $p \leftarrow getMigrationPoints(c, t_x)$       ▷ determine possible migration points
3:      $cs \leftarrow \lfloor size(p)/s \rfloor$         ▷ initial cluster size for each sink
4:      $p\prime \leftarrow p$
5:      for $i \leftarrow 1, s$ do
6:          $cntrs \leftarrow kmeans(p\prime, cs)$       ▷ get cluster centers using kmeans algo
7:          $p_i \leftarrow pdist(p\prime, cntrs)$       ▷ migration points nearest to centroids
8:          $p\prime \leftarrow p \setminus p_i$
9:          if $cs < th_s$ then                         ▷ if # of centroids < threshold for given # of sinks
10:             $df \leftarrow th_s - cs$
11:             $p_i = p_i \cup rand(p_i\prime, df)$    ▷ finalize point set
12:         end if
13:         $cmb_i \leftarrow combos(p_i, s)$           ▷ get combinations
14:     end for
15:     for $i \leftarrow 1, s$ do
16:         for $j \leftarrow 1, size(cmb_i)$ do
17:             $tt \leftarrow constructTopology(c, t_x, cmb_i(j))$
18:             for $k \leftarrow 1, n$ do              ▷ for each node
19:                 $t_{ijk} \leftarrow tx_k + rc_k$    ▷ transmission and receive cost for node k
20:             end for
21:         end for
22:     end for
23:     $rgVal \leftarrow 1 - (s/100)$
24:     while $e > 0$ do                                ▷ any node's energy is not depleted
25:         for $i \leftarrow 1, s$ do
26:             if $rn > rgVal$ then
27:                 $c \leftarrow rand(p, s)$           ▷ select s points randomly from site set p
28:             else
29:                 $u_i \leftarrow e_i - t_{ijk}$      ▷ update current energy matrix
30:                 for $j \leftarrow 1, size(cmb_i)$ do
31:                     $mp(j) \leftarrow min(t_j)$     ▷ get minimum element for each row
32:                 end for
33:                 $r_i \leftarrow max(mp)$            ▷ index of maximum of minimums
34:             end if
35:         end for
36:         $r_m \leftarrow max(r_1, r_2, \ldots, r_s)$  ▷ get global maximum
37:         for $i \leftarrow 1, s$ do
38:             $c_i \leftarrow cmb_m(r_m, i)$          ▷ set sinks' next coordinates
39:         end for
40:     end while
41: end procedure
```

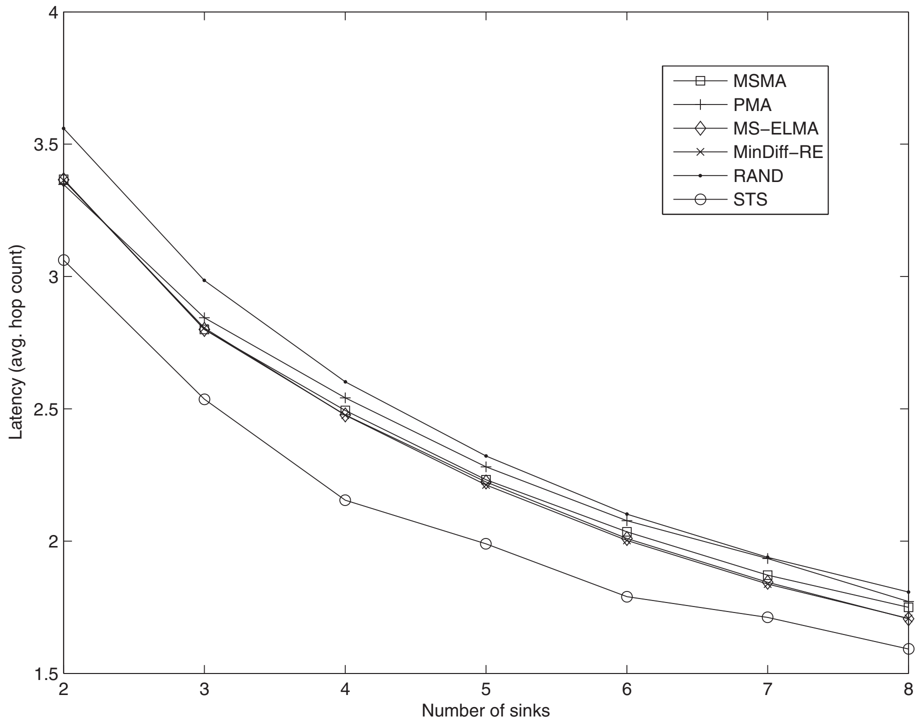


Fig. 4 Latency values for various numbers of sinks

to also come up with algorithms which do not require evaluation of so many combinations and distributing the related information while selecting next migration points.

Although there are a few distributed algorithms that attempt to solve the multiple-sink mobility problem, most require a large amount of data exchange, e.g., nodal residual energy, nodal data rate, energy needed to communicate packets for the whole network, as in [27]. Transferring such a large amount of data in each round causes excessive energy consumption in resource-constrained sensor nodes. Therefore, it is very important to find a low complexity and an efficient heuristic that requires minimum information exchange to decide sink movements and that can be run on sink nodes, which is what we achieve with our PMA algorithm.

Our PMA algorithm first filters out the sites closer to the nodes that have relatively lower residual energy compared to other nodes in the network. We do this because we only want to consider the sites close to high energy nodes as possible migration points so that when sinks migrate to those points, the high energy nodes will take the majority of the traffic-forwarding load since they will be at the highest level in the routing trees. PMA also t next migration points. The selected points should not be very close to each other, since moving all sinks to the same sub-region is not useful and makes the routing paths to the majority of sensor nodes to be excessively long. Therefore, we provide a balance between energy and distance while considering the next migration points. We select half of the sites to be close to high-energy nodes and the other half to be far away from these selected sites. In this way, the selected sites are tried to be evenly distributed to the whole sensor network region and are not clustered to a sub-region.

In PMA, the next migration points for sinks are determined as follows at each round of sink site determination. At the end of the current round, before determining next migration points for the next round, the sinks are at some locations, and their combined routing trees span the whole network. Each sink has its own routing tree spanning a portion of the nodes in the network. These nodes are considered to be the descendants of that sink. The routing trees of sinks do not overlap.

Each sink knows its descendants and their remaining energy levels (residual energy information is piggybacked into the data packets). Each sink selects the m_p percent of the minimum-energy nodes among its descendants, where m_p is a design parameter. The sink then determines the nearest sites to those minimum energy sensor nodes to filter out for the next round. These filtered out sites are *forbidden*. All sinks then exchange this forbidden site information among each other. This requires a total of $s(s-1)$ message exchanges among sink nodes. Each sink

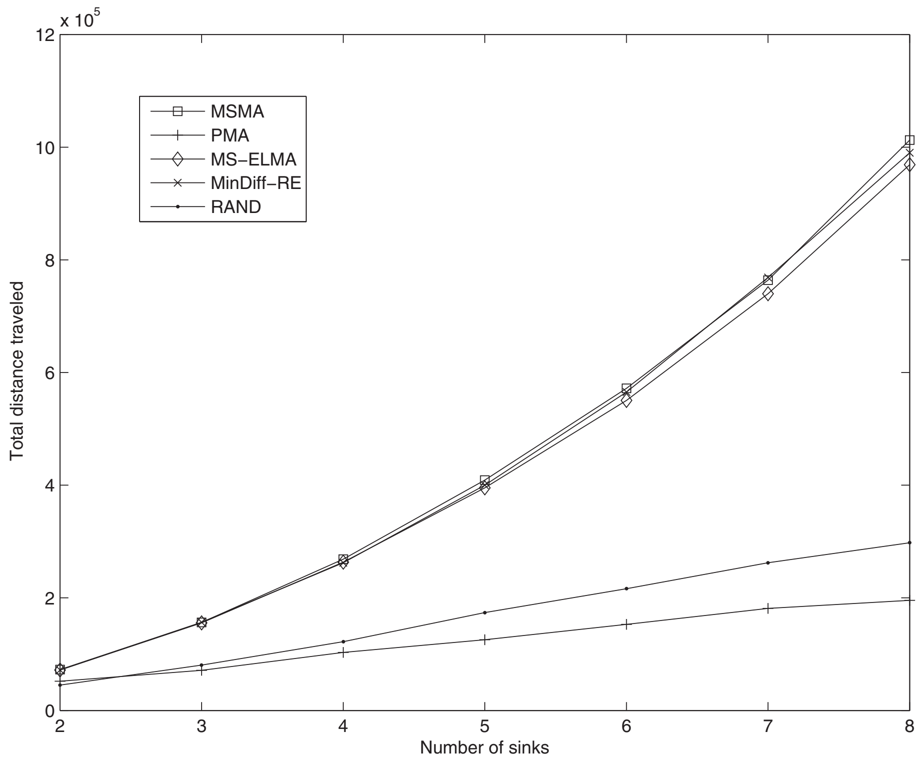


Fig. 5 Distances traveled for various numbers of sinks

then aggregates the forbidden site information coming from other sinks (takes the union) and in this way obtains a forbidden site list for the whole network. The remaining sites are *allowed* sites and can be used as possible migration points for the next round. Since the forbidden sites can be significant portion of the original point set, we use a parameter called *forbidden point threshold* (t_{fp}) to control its size. The total number of forbidden sites can not be bigger than this threshold. The introduction of this threshold eliminates the possibility of having the number of allowable sink sites to be less than the number of sinks s. Additionally, filtering out too many sites without global network information (like the energy matrix in the MSMA or MinDiff-RE) can eliminate advantageous sites to move to.

When a sink decides its forbidden sites, that means it decided on its allowable sites. For each allowable site in the region of its routing tree, the sink considers the sensor nodes around the site and finds out the node with maximum remaining energy. This minimum energy level is associated with that site. Then, the sink selects the first $s/2$ sites that have the largest associated minimum remaining energy levels. In other words, the first $s/2$ sites are selected when sites are sorted in decreasing order with respect to their associated minimum energy levels. We call these as max-min sites. They are good positions for sinks to move next. Then each sink sends $s/2$ max-min sites together with the corresponding energy values to a designated sink (minimum ID sink, for example). This requires a total of $s - 1$ message transfers to the designated sink. In this way the designated sink collects $s(s - 1)/2$ max-min positions. It then selects $s/2$ of them, i.e., the first $s/2$ sites with largest associated minimum-energy values.

After this selection, the remaining $s/2$ sites are determined one by one according to the total distance to previously selected sites. First, distance to each of the $s/2$ previously selected sites is calculated to determine total distance value for each remaining *allowable* site. The site with maximum value is selected as the $(s/2) + 1$st point and added to the previously selected site list. The iteration is done for each site until all remaining $s/2$ sites, that are far away from each other, are determined. This process is sequential since determining the remaining $s/2$ sites once (from the sorted list according to the maximum distance) can cause selecting sites near to each other.

After s sites are selected, s sinks are assigned to these sites using the well-known Hungarian assignment algorithm, like in the MSMA. This minimizes the total distance traveled while sinks move to the assigned sites afterwards. The other alternative is to notify the nearest sink to move as each sink site is determined, if network is delay-intolerant or movement cost is not very significant.

In this way, the algorithm finds a balance between the energy and distance components of the problem. It takes the energy values of the nodes into consideration in the first part by removing the sites close to low energy nodes. It also increases the distance between sinks to balance the forwarding load of the sensor nodes. The process is detailed in Algorithm 2. In each round, we need $O(s)$ computations to determine the next migration points.

In the PMA algorithm, each sink shares the site IDs of the forbidden sites (t_{fp} sites for each sink, in the worst case), as well as the $s/2$ minimum energy values (with site IDs) in each movement decision. This corresponds to $O(t_{fp}\, log p)$ and $O(s\,(log\, p + log\, E))$ (where E is the initial energy value) storage complexity, respectively. For 50 sink sites, 4 sinks and 10 joules (t_{fp} becomes 10), it requires 60 bits for each sink to share t_{fp} (10, for this case) forbidden site IDs and 40 bits for $s/2$ (2, for this case) minimum energy values and site-id pairs. $s(s-1)$ message exchanges are done sharing forbidden site IDs and $s-1$ exchanges are required for energy and ID pairs to flow across all sinks. If network lifetime is 500 rounds, total message exchange of all sinks during this lifetime becomes around 51 KB.

In the approach described in [27], the information exchange part is much more expensive, where each sink shares all its descendants' residual nodal energy levels with the others. The maximum energy amount that a node can have is divided into 40 levels (L), which requires 6 bits to represent. Sinks also need to append sensor IDs to a message so each counterpart can construct a full residual-energy-level table. This operation requires additional $log\, N$ bits. Since all nodes' energy levels should be shared, totally $N(log\, N + log\, L)$ bits are needed to convey the required information for a decision. For 500 nodes (and 40 levels), it requires 7500 bits for one sink to send the related information. For 500 rounds and 4 sinks, the total message exchange cost increases to 5.36 MB for the algorithm of [27]. This is 108 times more than what our PMA algorithm needs.

5 Performance evaluation

In this section, we discuss the results of our MATLAB-based performance evaluation of the proposed schemes. We compare our MSMA and PMA algorithms with four different schemes:

- MinDiff-RE: Azad and Chockalingam's algorithm minimizes the residual energy difference [26].
- MS-ELMA: Our naive algorithm which limits possible site-sink combinations and selects the one that maximizes the minimum remaining energy [34].

Algorithm 2 Prevent and Move Away Algorithm

```
1:  procedure PMA($c, t_x, cm, s, n$)                      ▷ c: coordinates, $t_x$: transmission range
2:      $cmp \leftarrow cm$                                  ▷ gets a copy of cm: candidate migration points
3:      while $e > 0$ do                                    ▷ any node's energy is not depleted
4:          for each round do
5:              $men \leftarrow (n * m_p * s)$               ▷ # of min. energy nodes to consider
6:              $t_{fp} \leftarrow (|cm| * m_p * s)$          ▷ forbidden site threshold
7:              for $i \leftarrow 1, s$ do                    ▷ for each sink
8:                  $min\_nodes(i) \leftarrow min(energy, men)$   ▷ min. energy nodes
9:              end for
10:             for $i \leftarrow 1, s$ do
11:                 $cmf(i) \leftarrow dist(cm, min\_nodes(i))$   ▷ get closest sink sites
12:             end for
13:             $cmf \leftarrow cmf_1 \cup cmf_2 \cup \ldots cmf_s$   ▷ forbidden sink sites
14:             $cmf' \leftarrow cmf(1 : t_{fp})$             ▷ get first $\#t_{fp}$ sink sites
15:             $cmp \leftarrow cmp - cmf'$                   ▷ allowable sink sites
16:             for $i \leftarrow 1, \lfloor s/2 \rfloor$ do
17:                 $sc_i \leftarrow maxmin(cmp)$             ▷ choose $s/2$ sites with max-min
18:             end for
19:             for $i \leftarrow \lfloor s/2 \rfloor + 1, s$ do
20:                 $sc_i \leftarrow max\left(dist\left(cm', \sum_{j=1}^{i-1} sc_j\right)\right)$   ▷ choose furthest $s/2$ sites
21:             end for
22:         end for
23:     end while
24: end procedure
```

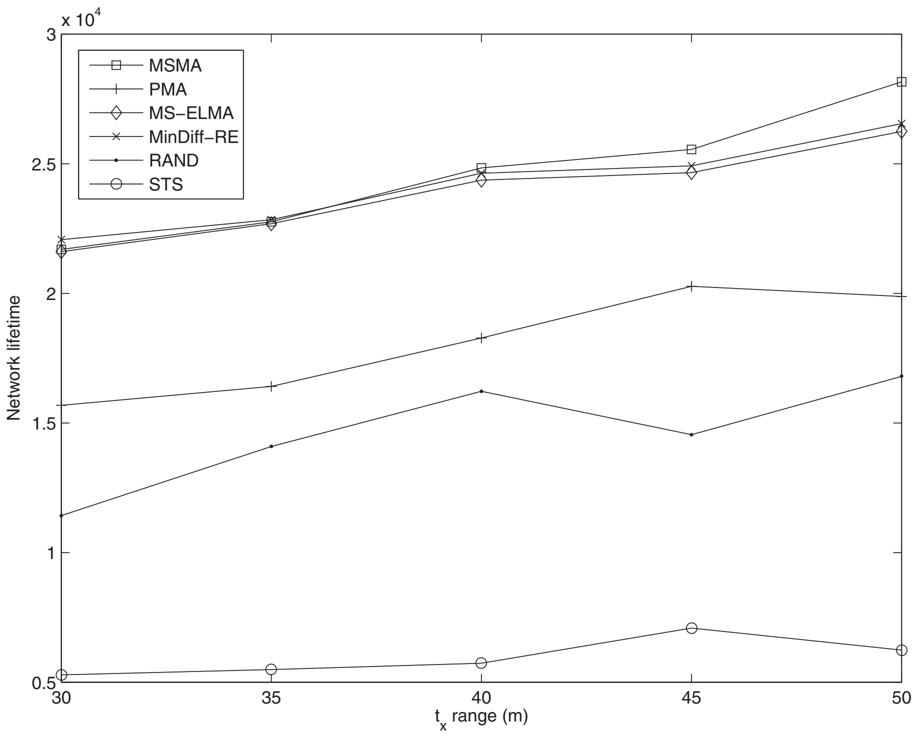


Fig. 6 Network lifetime values for various transmission ranges

- RAND: Sinks select sites randomly without considering any network parameter.
- STS: Sinks select sites at the beginning and stay there.

We compared the performance of these six schemes with respect to the following metrics:

- *Network lifetime*: The time until the first node depletes all its energy, which is the commonly used definition in the literature.
- *Latency*: Average hop-count that a packet travels until it reaches one of the sinks.
- *Total distance*: Total distance that mobile sinks travel during the network lifetime.

5.1 Simulation parameters

The sensor networks in our simulations have N static sensor nodes randomly deployed to a region and s mobile sinks (base stations). The deployment region is a square-shaped region with area size $300 \times 300\ m^2$. After mobile sinks move to their initial locations, they start broadcasting topology construction messages to construct tree-based multi-hop routing topologies (routing trees) from top to bottom. After topology construction, each sink node will have its routing tree constructed rooted at itself. Routing trees do not overlap. We use a constant packet generation rate Q (1 packet/s) for each sensor node. We define the network lifetime as the period of time until the first node dies. This is a commonly used definition in the literature. We defined data latency as the average hop-count a packet originating at a sensor node travels to reach to the corresponding sink.

The energy model and the radio characteristics used in the simulations are from [36]. Transmission energy cost is related to the number of bits transmitted and to the square of distance between transmitter and receiver, whereas receive energy cost is related just to the number of bits received. In our simulations, we assume a data packet is 50 bytes long and a control packet (used for topology construction, for example) is 20 bytes. We assume radio device consumes E_{elec} = 50 nJ/bit to run the transceiver circuitry and ϵ_{amp} = 100 pJ/bit/m^2 for the transmit amplifier to achieve an acceptable $\frac{E_b}{E_n}$ [36]. Each sensor node has 50J initial amount of energy. These simulation parameters and their typical values are summarized in Table 1.

5.2 Simulation results

Network lifetime results for the number of sinks (sink count) between two and eight are given in Fig. 3. Our MSMA algorithm gives a better network lifetime compared to other schemes for all values of sink count. Even MinDiff-RE uses four times more combinations than MSMA when the number of sinks is eight, the MSMA

Table 1 Simulation parameters

Parameter	Value
Area	$300 \times 300\ m^2$
Number of sensor nodes	500
Node deployment	Random and uniform deployment
Transmission range	40m
Data routing	Shortest path
Sink site determination	Neighborhood-based SSD algorithm [30]
Nodes' initial energy	50 J
Radio characteristics	First-order radio model [36]

gives a three percent better lifetime performance compared to it. The PMA algorithm performs 22 % worse compared to centralized algorithms on the average, but it still outperforms RAND approach by 21 % on the average.

Data latency (average hop-count) is another important metric for wireless sensor networks. Algorithms should have lower data latencies, especially for delay-intolerant applications (like fire detection). The latency comparison of the schemes is given in Fig. 4. The static sink approach has the best (lowest) latency values—10 % lower than the centralized algorithms and 13 and 16 % lower than the PMA algorithm and the RAND approach, respectively. Centralized algorithms have almost the same latency as each other. The PMA algorithm also has the same latency as centralized algorithms, but it has lower latency (3 % on the average) compared to the RAND.

Although it is not taken into consideration most of the time, travel distance between two sites is another cost to consider for sink mobility problems because mobile sinks spend energy (for example, fuel) to move from one site to another. The distance traveled effects also the reconfiguration latency. Figure 5 shows the total distance covered by mobile base stations for different sink-count values. Centralized algorithms cause sinks to travel more compared to our PMA algorithm and the RAND approach. For example, when the sink-count is eight, centralized algorithms cause the sinks to cover 5.2 times more distance than the other two schemes. Our PMA algorithm achieves the least travel-distance. Sinks cover 1.5 times more distance when they use the RAND approach compared to

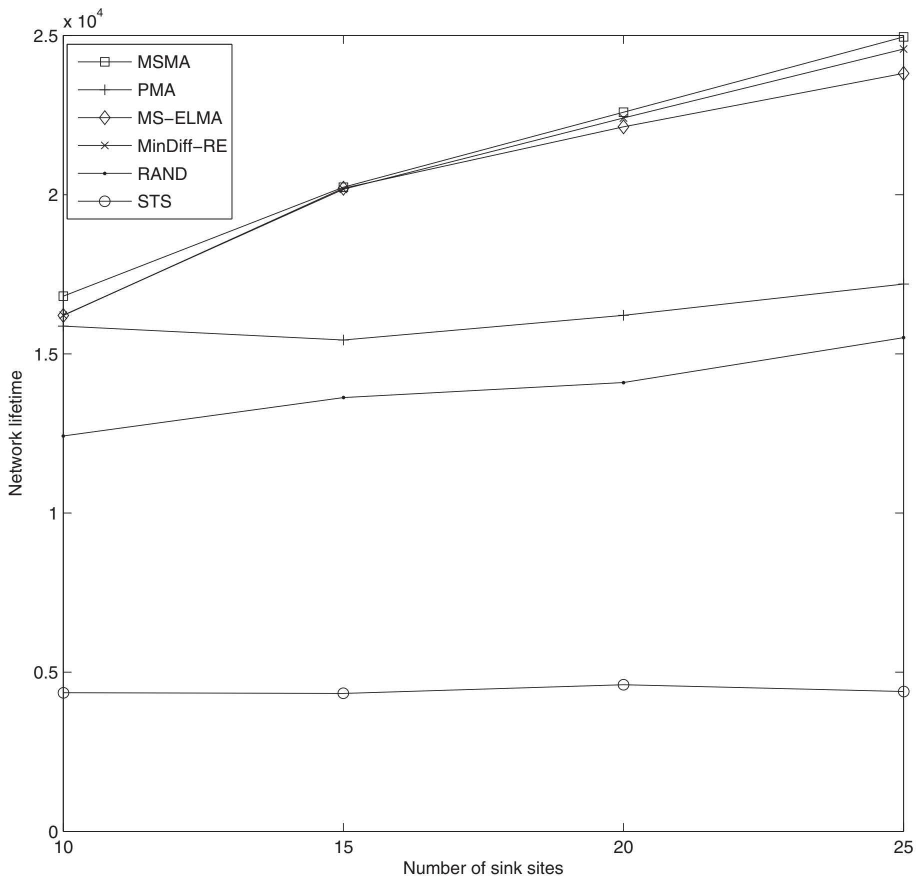


Fig. 7 Network lifetime values for various numbers of sink sites

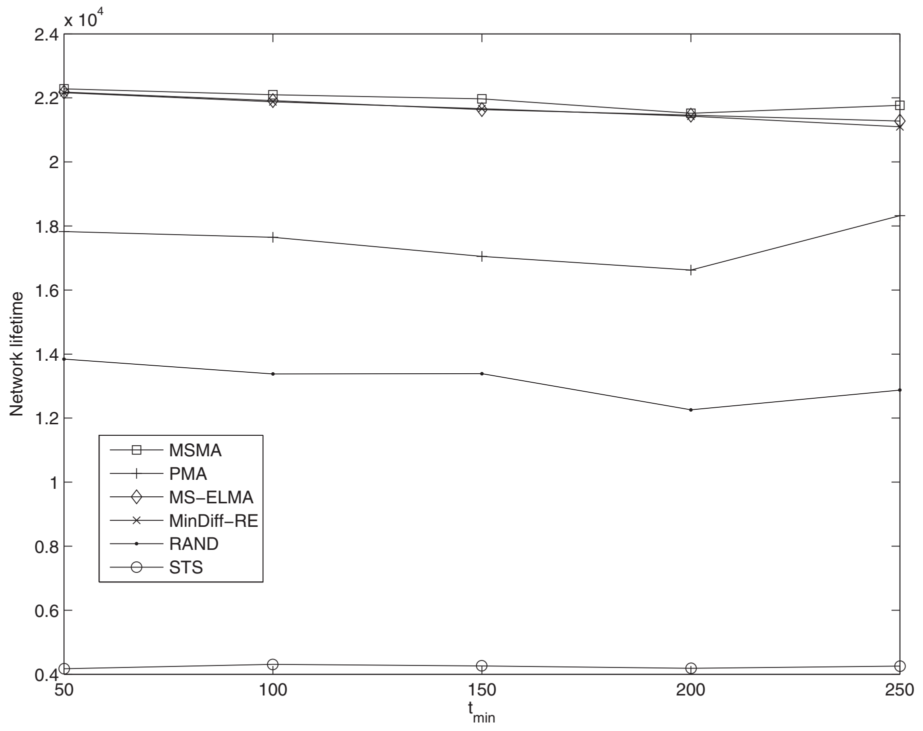


Fig. 8 Network lifetime values for various t_{min} values

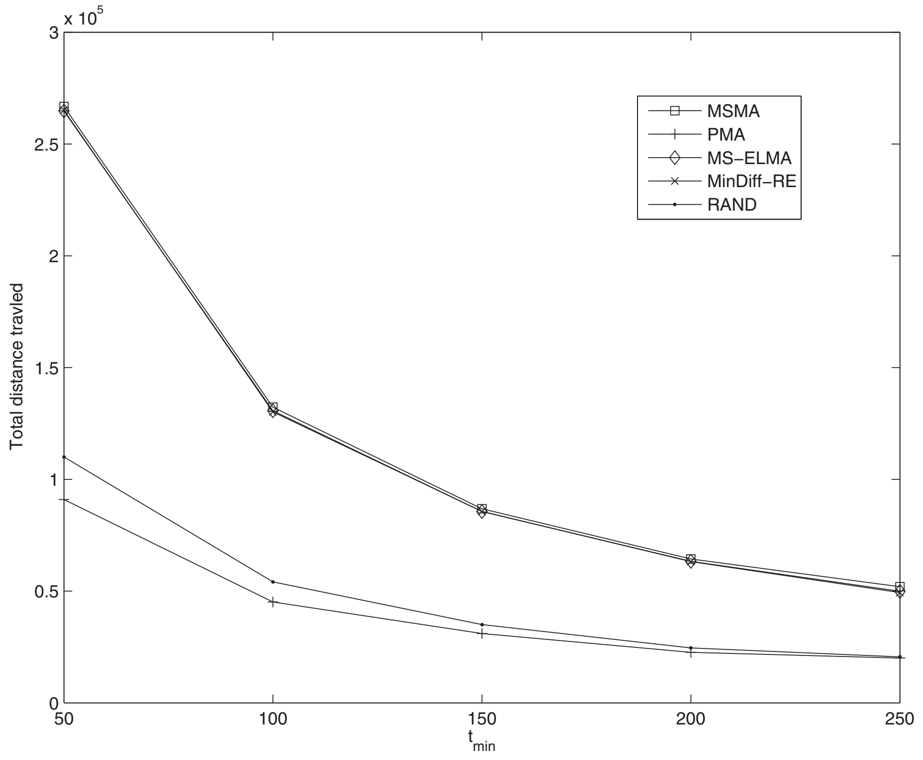


Fig. 9 Distances traveled for various t_{min} values

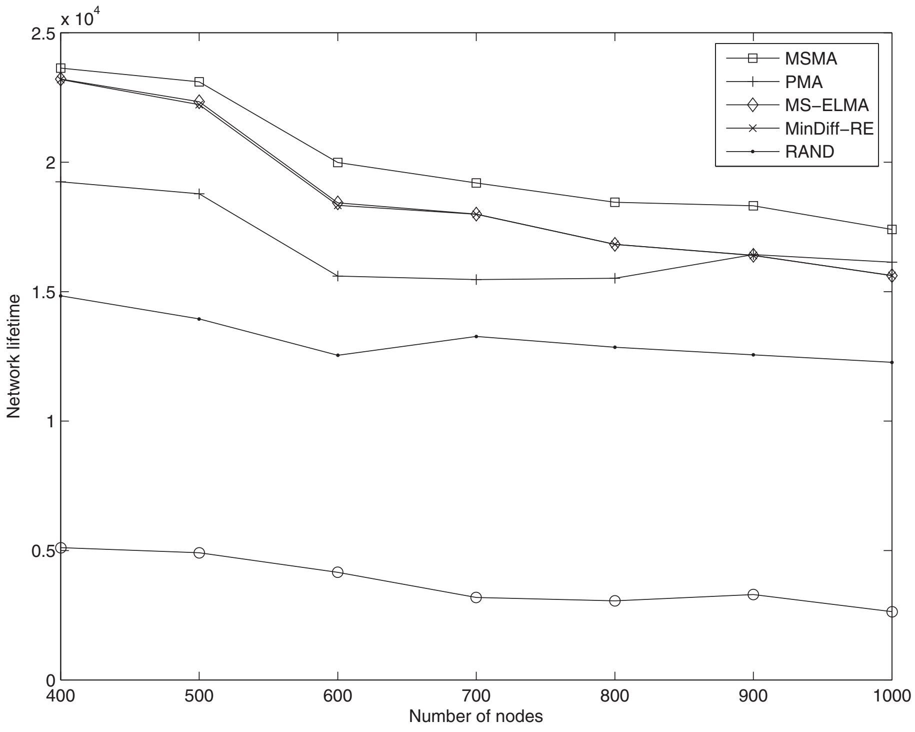


Fig. 10 Network lifetime values for various numbers of nodes

the PMA algorithm. Therefore, our PMA algorithm has a longer network lifetime and better data latency while traveling less, compared to the RAND approach.

Figure 6 gives network lifetime performance of the schemes for different values of transmission-range. The sink-count values, if fixed to five, and the node-count value is fixed to 500. The performance of all schemes gets better when transmission-range increases. The performance of RAND increases by almost 50 % when the transmission range is increased from 30 to 50m. This is more than the increase of any other scheme (others vary between 20 and 30 %). Since RAND does not use any intelligent mechanism while moving sinks, increasing transmission range causes an increase its performance more than the performance increase of any other scheme.

The effect of the number of sink sites (site-count) on the network-lifetime performance of the algorithms is shown in Fig. 7. The sink-count, node-count, and transmission-range values are fixed to 4, 500, and 40 m, respectively. The lifetime achieved by MSMA, MinDiffRE, and MS-ELMA schemes improves 50 % on average when site-count increases from 10 to 25. Increasing possible sink sites creates more options (combinations) for these algorithms to select from, which helps improving the lifetime of the network. This increase amount drops to 10 % for the PMA algorithm, since it does not use any prior information (topology information for each site) while selecting the sites on-the-fly. Therefore, increasing the number of sites does not affect PMA performance as much as its centralized counterparts.

Network lifetime values for different minimum stay time (t_{min}) values are given in Fig. 8. The sink-count and site-count values are fixed to 4 and 18, respectively. The t_{min} value is changed between 50 and 250 s . The performance of all algorithms is not affected by more than 7.5 %. Our MSMA algorithms is affected least (2 %), while RAND approach is affected most (7.5 %). However, increasing t_{min} helps sinks to travel less in the area, as seen in Fig. 9. Total travel distances decrease by an average factor of 5 while t_{min} increases from 50 to 250. That means that it is possible to reduce the mobility cost by a factor of five when the network lifetime performance is decreased by just five percent. However, increasing t_{min} value beyond a threshold will cause algorithms' achieved lifetime values to decrease, since staying longer at a point can cause unbalanced energy values of the nodes. Therefore, ideal t_{min} value can change according to the application or needs. It should be increased if mobility cost is taken into consideration; however, beyond a threshold, this will cause lifetime performance to decrease.

The effect of the number of nodes changed in the area is shown in Fig. 10. Network lifetime performance is inversely affected by increasing sensor nodes in the area. In this case, hot-spot nodes must relay more packets compared to when there are fewer nodes. However, the decreasing ratio in network lifetime changes according to the algorithm used. The static sink case is affected most (93 %) and the PMA algorithm least (19 %). The PMA's algorithm network lifetime becomes better than MinDiff-RE's and MS-ELMA's when the number of nodes increases to 1,000. Since the current energy levels of the nodes are taken into consideration when the network operates, more-balanced topology trees can be constructed. This situation contributes to lifetime more than the pre-calculated topology trees used in MinDiff-RE and MS-ELMA do. Therefore, it would be better to use the PMA algorithm when there are more than a few hundred nodes in the area. Sinks also travel two times less often using that method compared to centralized algorithms.

6 Conclusion

In this paper, we propose two multiple-sink mobility algorithms, MSMA and PMA, to coordinate movements of multiple sinks in a wireless sensor network. The MSMA uses the energy expenditure information of all nodes for different sink-site combinations. Unlike other similar approaches, the MSMA does not use all possible combinations, but instead limits the number of combinations according to a threshold value to reduce complexity. The PMA algorithm is a distributed algorithm that uses no global network information to determine the sinks' next migration points. Each sink forbids some sites from using its children's residual energy information. One group of sinks selects sites from the remaining set using the maximum of the minimum energy of the first-hop nodes. The other group selects points farther from the previously selected ones. We compare the performance of the proposed schemes with the MinDiff-RE, RAND, and static-sink approaches in terms of network lifetime, latency, and total distance covered by the sinks.

Under varying metrics, the experiment results show that the MSMA performs better in terms of network than any other scheme. The MSMA gives better network lifetime than the MinDiff-RE even though it uses up to four times fewer sink-site combinations. Latency (average hop count to any sink) is lowest for the static-sink case and almost the same for the other schemes. The PMA algorithm achieves less lifetime (around 20 % on the average) compared to centralized algorithms (but better than random movement under all conditions); however, its performance increases (becomes better than the MinDiff-RE and MS-ELMA) when there are more than a few hundred nodes. Sinks also travel less compared to other schemes when using the PMA algorithm.

Competing interests
The authors declare that they have no competing interests.

Received: 20 August 2015 Accepted: 27 October 2015
Published online: 17 November 2015

References

1. R Silva, JS Silva, F Boavida, Mobility in wireless sensor networks—survey and proposal. Comput. Commun. **52**, 1–20 (2014)
2. A Nayak, I Stojmenovic, *Wireless sensor and actuator networks: algorithms and protocols for scalable coordination and data communication*. (Wiley-Interscience, New York, NY, USA, 2010)
3. K Han, J Luo, Y Liu, AV Vasilakos, Algorithm design for data communications in duty-cycled wireless sensor networks: a survey. IEEE Commun. Mag. **51**(7), 107–113 (2013)
4. M Cardei, J Wu, M Lu, MO Pervaiz, in *IEEE International Conference on Wireless And Mobile Computing, Networking And Communications (WiMob)*. Maximum network lifetime in wireless sensor networks with adjustable sensing ranges (IEEE Montreal, Canada, 2005), pp. 438–445
5. Y Xiao, M Peng, J Gibson, GG Xie, D-Z Du, AV Vasilakos, Tight performance bounds of multihop fair access for mac protocols in wireless sensor networks and underwater sensor networks. IEEE Trans. Mob. Comput. **11**(10), 1538–1554 (2012)
6. MHS Gilani, I Sarrafi, M Abbaspour, An adaptive CSMA/TDMA hybrid mac for energy and throughput improvement of wireless sensor networks. Ad Hoc Netw. **11**(4), 1297–1304 (2013)
7. X Xu, R Ansari, A Khokhar, AV Vasilakos, Hierarchical data aggregation using compressive sensing (hdacs) in wsns. ACM Trans. Sens. Netw. (TOSN). **11**(3), 45–14525 (2015)
8. L Xiang, J Luo, A Vasilakos, in *Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*. Compressed data aggregation for energy efficient wireless sensor networks (IEEE Salt Lake City, Utah, USA, 2011), pp. 46–54
9. Y Liu, N Xiong, Y Zhao, AV Vasilakos, J Gao, Y Jia, Multi-layer clustering routing algorithm for wireless vehicular sensor networks. IET Commun. **4**(7), 810–816 (2010)
10. X-Y Liu, Y Zhu, L Kong, C Liu, Y Gu, AV Vasilakos, M-Y Wu, CDC: Compressive data collection for wireless sensor networks. IEEE Trans. Parallel and Distrib. Syst. **26**(8), 2188–2197 (2015)
11. G Wei, Y Ling, B Guo, B Xiao, AV Vasilakos, Prediction-based data aggregation in wireless sensor networks: Combining grey model and kalman filter. Comput. Commun. **34**(6), 793–802 (2011)
12. N Chilamkurti, S Zeadally, A Vasilakos, V Sharma, Cross-layer support for energy efficient routing in wireless sensor networks. J. Sensors. **2009**(134165). 9(2009)
13. Y Zeng, K Xiang, D Li, A Vasilakos, Directional routing and scheduling for green vehicular delay tolerant networks. Wirel. Netw. **19**(2), 161–173 (2013)
14. Y Yao, Q Cao, AV Vasilakos, Edal: An energy-efficient, delay-aware, and lifetime-balancing data collection protocol for heterogeneous wireless sensor networks. IEEE/ACM Trans. Netw. **23**(3), 810–823 (2015)
15. Y Yao, Q Cao, AV Vasilakos, in *10th International Conference on Mobile Ad-Hoc and Sensor Systems (MASS)*. Edal: An energy-efficient, delay-aware, and lifetime-balancing data collection protocol for wireless sensor networks (IEEE Hangzhou, China, 2013), pp. 182–190
16. M Li, Z Li, AV Vasilakos, A survey on topology control in wireless sensor networks: taxonomy, comparative study, and open issues. Proc. IEEE. **101**(12), 2538–2557 (2013)
17. S Basagni, A Carosi, C Petrioli, in *Vehicular Technology Conference*. Controlled vs. uncontrolled mobility in wireless sensor networks: some performance insights (IEEE Dublin, Ireland, 2007)
18. J Luo, JP Hubaux, in *IEEE INFOCOM*. Joint mobility and routing for lifetime elongation in wireless sensor networks, vol. 3 (IEEE Miami, Florida, USA, 2005), pp. 1735–1746
19. I Papadimitriou, L Georgiadis, Energy-aware routing to maximize lifetime in wireless sensor networks with mobile sink. J Commun. Softw. Syst. **2**(2), 141–151 (2006)
20. S Basagni, A Carosi, E.Melachrinoudis, C Petrioli, M Wang, Controlled sink mobility for prolonging wireless sensor networks lifetime. ACM J. Wirel. Netw (WINET). **14**(6), 831–858 (2007)

21. W Liang, J Luo, X Xu, in *IEEE Global Telecommunications Conference (GLOBECOM)*. Prolonging network lifetime via a controlled mobile sink in wireless sensor networks (IEEE Miami, Florida, USA, 2010), pp. 1–6
22. Z Xu, W Liang, Y Xu, in *IEEE 8th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. Network lifetime maximization in delay-tolerant sensor networks with a mobile sink (IEEE Hangzhou, China, 2012), pp. 9–16
23. K Akkaya, M Younis, M Bangad, Sink repositioning for enhanced performance in wireless sensor networks. Comput. Netw. **49**(4), 512–534 (2005)
24. Z Vincze, D Vass, R Vida, A Vidács, A Telcs, Adaptive sink mobility in event-driven densely deployed wireless sensor networks. Ad Hoc & Sens. Wirel. Netw. **3**(2–3), 255–284 (2007)
25. SR Gandham, M Dawande, R Prakash, S Venkatesan, in *IEEE Global Telecommunications Conference*. Energy efficient schemes for wireless sensor networks with multiple mobile base stations (IEEE Location: San Francisco, California, USA, 2003), pp. 377–381
26. AP Azad, A Chockalingam, in *IEEE Wireless Communications and Networking Conference (WCNC)*. Mobile base stations placement and energy aware routing in wireless sensor networks (IEEE Las Vegas, Nevada, USA, 2006), pp. 264–269
27. S Basagni, A Carosi, C Petrioli, C Phillips, Coordinated and controlled mobility of multiple sinks for maximizing the lifetime of wireless sensor networks. Wirel. Netw. **17**, 759–778 (2011)
28. W Liang, J Luo, in *Local Computer Networks (LCN), 2011 IEEE 36th Conference On*. Network lifetime maximization in sensor networks with multiple mobile sinks (IEEE Bonn, Germany, 2011), pp. 350–357
29. L Friedmann, L Boukhatem, in *Third International Conference on Networking and Services (ICNS)*. Efficient multi-sink relocation in wireless sensor network (IARIA Athens, Greece, 2007), pp. 90–97
30. M Koc, I Korpeoglu, Controlled sink mobility algorithms for wireless sensor networks. International Journal of Distributed Sensor Networks. **2014**(167508), 12 (2014)
31. R Hoes, T Basten, W Yeow, C Tham, M Geilen, H Corporaal, Qos management for wireless sensor networks with a mobile sink. Lect. Notes Comput. Sci. **5432**, 53–68 (2009)
32. G Wang, T Wang, W Jia, M Guo, J Li, Adaptive location updates for mobile sinks in wireless sensor networks. J. Supercomput. **47**(2), 127–145 (2009)
33. K Lee. A data gathering scheme using mobile sink dynamic tree in wireless sensor networks, vol. 107, (2011), pp. 99–107
34. M Koc, I Korpeoglu, Traffic- and energy-load–based sink mobility algorithms for wireless sensor networks. International Journal of Sensor Networks. In Press
35. R Burkard, M Dell'amico, S Martello, *Assignment Problems, Revised Print*. (Siam, Philadelphia, USA, 2012)
36. W Heinzelman, A Chandrakasan, H Balakrishnan, in *Proceedings of the 33rd International Conference on System Sciences (HICSS '00)*. Energy-efficient communication protocol for wireless microsensor networks (IEEE Maui, Hawaii, USA, 2000), pp. 1–10