Ad Hoc Networks 8 (2010) 378–390

Contents lists available at ScienceDirect

Ad Hoc Networks

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

Placement of multiple mobile data collectors in wireless sensor networks

Waleed Alsalih [a,*], Hossam Hassanein [b], Selim Akl [b]

[a] Computer Science Department, King Saud University, Riyadh, Saudi Arabia
[b] School of Computing, Queen's University, Kingston, Ont., Canada

ARTICLE INFO

Article history:
Received 14 July 2008
Received in revised form 10 August 2009
Accepted 27 August 2009
Available online 17 October 2009

Keywords:
Sensor networks
Mobile data collectors
Placement

ABSTRACT

A major challenge affecting the lifetime of Wireless Sensor Networks (WSNs) comes from the unbalanced energy consumption over different parts of the network. This unbalanced energy consumption is a direct result of having a stationary sink: nodes near the sink are intensively used to relay data for other nodes to the sink. A natural solution to such a problem is to have multiple mobile sink nodes (which we call data collectors), and to change their locations periodically so that the load is distributed evenly among all sensor nodes. In this paper we propose a mobile data collector placement scheme for extending the lifetime of the network. In our scheme the lifetime of the network is divided into rounds and data collectors are moved to new locations at the beginning of each round. While previous work has focused on placing data collectors at predefined spots (e.g., the work in Gandham et al. (2003) [1]) or at the boundary of the network (e.g., the work in Azad and Chockalingam (2006) [2]), we define and solve two problems which are more general: the on-track placement where data collectors can be placed only along predefined tracks (roads) spanning the sensing field, and the general placement where data collectors may be placed at any point in the sensing field.

We formulate the problems as Mixed Integer Linear Programs (MILPs) and use a MILP solver (with a constant time limit) to find near-optimal placements of the data collectors and to find routing paths to deliver data to data collectors. Our experiments show that our schemes make significant extension to the lifetime of the network.

1. Introduction

The recently developed tiny sensor nodes have enabled a new generation of large-scale networks of untethered, unattended sensor devices suitable for a wide range of commercial, scientific, health, surveillance, and military applications. This rapidly evolving technology promises to revolutionize the way we interact with the physical environment and to facilitate collecting data which has never been available before [3]. However, as a result of the limited energy supply for sensor nodes, extending the lifetime of Wireless Sensor Networks (WSNs) has been a primary target for a significant amount of research during the last decade.

A sensor node has a wireless communication interface through which it can communicate with other devices in its vicinity. Due to the scarcity of the energy reservoir and due to the fact that communication is the dominant power consumer in a sensor node, the transmission range of these nodes is limited for energy-efficiency purposes. Sensor nodes which are spatially distant from the sink node use multi-hop relaying to deliver data to the sink. Multi-hop communication results in an unbalanced energy expenditure over different parts of the network; nodes around the sink deplete their energy reserve much faster than distant nodes [4,1,5]. Not only does this stop those nodes around the sink from functioning, but it also renders the sink unreachable by other nodes. While

* Corresponding author.
E-mail addresses: wsalih@ksu.edu.sa (W. Alsalih), hossam@cs.queensu.ca (H. Hassanein), akl@cs.queensu.ca (S. Akl).

1570-8705/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2009.08.007

existing energy-aware protocols, at physical, Medium Access Control (MAC), and network layers, achieve significant energy savings for individual sensor nodes, they fail to solve this topology-related problem. One way to balance the load over the network is to deploy more nodes in areas closer to the sink. The authors in [6] proposed a nonuniform node distribution strategy that divides the sensing field into a number of coronas and gives the ratio in node densities between two consecutive coronas. One problem of this approach is the exponential growth of the total number of sensor nodes in the network. Moreover, since the area around the sink will have too many nodes, there will be a need for a complicated MAC protocol to control the access of these nodes to the wireless channel and/or to manage their duty cycles.

In this paper, we argue for using multiple mobile data collectors and propose a scheme for placing these data collectors in a way that balances the energy expenditure and increases the lifetime of the network. Our scheme divides the lifetime of the network into fixed length rounds (e.g., hours, days, or weeks) and moves the data collectors, which can be mounted on Autonomous Unmanned Vehicles (AUVs), to new locations at the beginning of each round. Some recently proposed schemes have addressed the issue of mobile sinks. However, they are either limited to a given set of predefined spots for sink locations and/or provide results that can be arbitrarily far from the optimal ones. To this end, the novel contribution of this paper is twofold:

1. We define and solve two placement and routing problems. The first one assumes the existence of predefined tracks (e.g., a road network) spanning the sensing field, and data collectors can be moved over and placed at any point along these tracks. This would be practical in a situation where data collectors are carried on AUVs or robots that move along paved roads only. In the second one, a data collector can be placed anywhere in the sensing field. Some previous work assumes the existence of a set of predefined spots (i.e., points) where data collectors may be placed [1], and some is limited to placing data collectors at the boundaries of the field [2].
2. We discretize the search space without affecting the quality of the derived solutions: we devise an algorithm that finds a finite set of relatively small number of points, and we prove the existence of an optimal placement in which each data collector is placed at a point in that set. Since the problem is modeled as a Mixed Integer Linear Program (MILP), making the cardinality of this set as small as possible would significantly improve the efficiency and the solution quality. Moreover, our approach involves solving one linear program per round; this makes our approach more efficient than earlier schemes, viz [7], which is based on solving a number of linear programming instances which is exponential in the number of data collectors.

By formulating the problems as MILPs, optimal solutions can be found. However, that may require exponential run-time in the worst case [8]. Therefore, we impose a time limit on a branch-and-bound solver in order to find near-optimal solutions in reasonable time.

The rest of this paper is organized as follows. Section 2 describes the model of the system and gives a formal problem definition. In Section 3, we present our placement schemes. Section 4 shows the experimental results. Finally, in Section 5, we conclude by summarizing the contributions and pointing out some related future research directions.

2. System model and problem definition

We consider a WSN consisting of N sensor nodes and R data collectors. Each sensor node collects data from the surrounding environment and sends the collected data to one of the data collectors. The transmission range, which is modeled as a disk, of all sensor nodes is fixed to r (m). The topology of the network is modeled as a graph $G = (V, E)$, where $V = \{n_0, n_1, \ldots, n_{N-1}\}$ is the set of N sensor nodes, and $(i,j) \in E$, if sensor nodes n_i and n_j are within the transmission range of each other. Each sensor node n_i has a data generation rate G_i; G_i is the number of data units generated by node n_i per time unit (without loss of generality, we assume that the round is our time unit).

Our schemes are independent of the underlying MAC protocol. We assume a capacity limit for each sensor node which limits the number of data units that can be transmitted by a sensor node during one round. The capacity of a sensor node n_i is denoted by C_i. This parameter can be adjusted to comply with any constraints imposed by any MAC protocol.

We define the lifetime of the network as the time until the first sensor node dies. Yet other definitions (e.g., the time until a particular proportion of the sensors die) can be equally used in our approach.

We also consider a set of tracks in the sensing field; these tracks are modeled as a set of L line segments $\{l_0, l_1, \ldots, l_{L-1}\}$.

2.1. Assumptions

1. For each sensor node n_i, the location loc_i, the residual energy E_i, and the data generation rate G_i are known. The location of a sensor node can be obtained using the Global Positioning System (GPS) or other GPS-less schemes [9,10]. The values of loc_i, E_i and G_i are estimated at the node itself and piggybacked with data packets.
2. Data collectors are not energy constrained (as they can be recharged easily).
3. Every generated data unit is sent to any data collector (not to all of them).

2.2. Problem definition

The two placement problems can be described as follows:

The lifetime of the network is divided into equal length rounds. At the beginning of each round, find the optimal locations of R data collectors together with the routing paths to deliver the generated data to data collectors, which maximizes the minimum residual energy at the end of the round. In the on-track placement problem, data collectors can be placed only at points along the tracks. In the general placement problem, data collectors may be placed anywhere in the sensing field.

3. Data collector placement scheme

A major challenge in this placement problem is the infinite search space for data collector locations. The first step in our approach is to make the search space finite without affecting the quality of the solution. To explain our method of finding such a finite search space, we make the following definitions.

Definition 1. A finite set of points K is *complete* if and only if it satisfies the following property: there exists an optimal[1] placement of data collectors in which each data collector is placed at a point in K.

Finding a complete set would make the placement problem a discrete optimization problem rather than a continuous one. In the following two subsections we show how to find complete sets for the on-track and the general placement problems.

3.1. A complete set for the on-track placement problem

For the on-track placement problem, a data collector may be placed at any point on any track as long as it is within the transmission range of at least one sensor node. This results in an infinite number of possible locations for each data collector and, hence, an infinite search space. Recall that a track is modeled as a line segment; we will use the terms line segment and track interchangeably. To explain our method of discretizing the search space for the on-track problem, we make the following definitions.

Definition 2. For any line segment l, an *overlapping segment* is the intersection of l and the transmission disks of a nonempty subset of sensor nodes. For an overlapping segment α, let $S(\alpha)$ denote the subset of sensor nodes whose transmission disks overlap with l at α.

In Fig. 1, the overlapping segments of the track st with respect to sensor nodes n_1, n_2, n_3, n_4, and n_5 are sa, bc, cd, de, ff (i.e., the point f), and gt. $S(sa) = \{n_1\}$, $S(bc) = \{n_2\}, S(cd) = \{n_2, n_3\}, S(de) = \{n_3\}, S(ff) = \{n_4\}$, and $S(gt) = \{n_5\}$.

Definition 3. For any line segment l, an overlapping segment α is *maximal* if there is no overlapping segment β on l, where $S(\alpha) \subset S(\beta)$.

In Fig. 1, the overlapping segments sa, cd, ff, and gt are maximal. Now we state the following lemma.

Lemma 1. *For every overlapping segment β, there exists a maximal overlapping segment α, such that $S(\beta) \subseteq S(\alpha)$.*

Proof. If β is maximal, we choose α to be β itself. If β is not maximal then, by definition, there exists an overlapping segment α_1 such that $S(\beta) \subset S(\alpha_1)$. If α_1 is maximal, we choose α to be α_1, and if α_1 is not maximal then, by definition, there exists an overlapping segment α_2 such that $S(\alpha_1) \subset S(\alpha_2)$. The process continues until a maximal overlapping segment α_z is found; we choose α to be α_z. It is obvious that $|S(\alpha_z)| \leqslant N$, and $|S(\beta)| < |S(\alpha_1)| < |S(\alpha_2)| < \ldots < |S(\alpha_z)| \leqslant N$. Therefore the process of finding the maximal overlapping segment α_z takes a finite number of steps which is at most $N - 1$. □

Then, we deduce the following theorem.

Theorem 1. *A set K that contains one point from every maximal overlapping segment is complete.*

Proof. To prove this theorem, it suffices to show that for any arbitrary placement $\aleph$ we can construct an equivalent placement $\aleph^*$ in which every data collector is placed at a point in K. Let us say that in $\aleph$, a data collector B is placed such that it is within the transmission range of a subset of sensor nodes H. It is obvious that there exists an overlapping segment β, such that $H \subseteq S(\beta)$. From the previous lemma, there exists a maximal overlapping segment α, such that $S(\beta) \subseteq S(\alpha)$. In $\aleph^*$, we place B at the point in K that belongs to α, so that B is placed at a point in K and is still within the transmission range of all sensor nodes in H. Repeating for all data collectors, we construct a placement $\aleph^*$ which is equivalent to the placement $\aleph$. □

Before we proceed to the algorithm that finds all maximal overlapping segments, we define the notations of entry points, exit points, and tangent points. Each line segment (i.e., track) has a set of intersection points; an intersection point is a point where the line segment intersects with the boundary of the transmission disk of a sensor node. In Fig. 1, the line segment st has 7 intersection points: a, b, c, d, e, f, and g. By walking over a line segment from left to right (and from down to up if the line segment is parallel to the y-axis), intersection points incident to that line segment can be classified into entry points, exit points, and tangent points. An entry point is one at which we enter the transmission disk of a sensor node. An exit point is one at which we leave the transmission disk of a sensor node. If the line segment is a tangent to the transmission disk, their intersection is a tangent point. In Fig. 1, b, c, and g are entry points; a, d, and e are exit points; and f is a tangent point. Note that if multiple transmission disks intersect with a line segment at the same point, we consider multiple intersection points at that point. These points have the same coordinates but some of them may be entry points, some may be exit points, and some may be tangent points.

[1] Optimality can be defined according to any energy-based objective function (e.g., total consumed energy, maximum amount of energy consumed by a single node, network lifetime,…, etc).

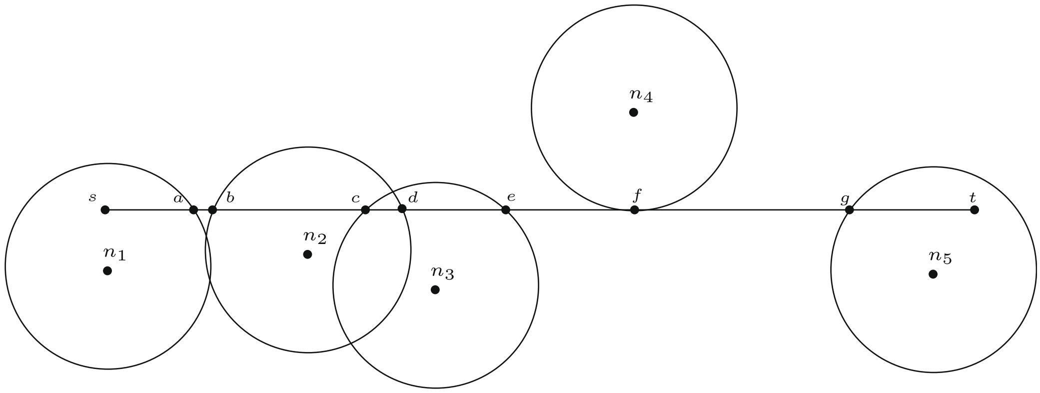


Fig. 1. Intersection points on a line segment.

Let the rank of an intersection point be 1 if it is an entry point, 2 if it is a tangent point, and 3 if it is an exit point. Intersection points incident to a line segment can be sorted in a non-decreasing lexicographic order according to their (x-coordinate, y-coordinate, rank), i.e., $(x_1, y_1, r_1) \leqslant (x_2, y_2, r_2)$ if and only if:

$x_1 < x_2$,
$x_1 = x_2$ and $y_1 < y_2$, or
$x_1 = x_2, y_1 = y_2$, and $r_1 \leqslant r_2$.

Now we state the following obvious observation without a proof.

Algorithm 1 (*Finding all maximal overlapping segments).*

```
Procedure FindMaximalOverlappingSegments()
Output: A set K that contains one point from every
   maximal overlapping segment.
K := φ;
foreach track l do
  Find all intersection points incident to l;
  Sort all intersection points incident to l in a non-
   decreasing lexicographic order according to their (x-
   coordinate, y-coordinate, rank);
  Active := True;
  foreach intersection point p incident to l do
    if (Active AND p is an exit point) OR p is a tangent
    point then
      K := K ∪ {p};
      Active := False;
    end
    if p is an entry point then
      Active := True;
    end
  end
  if Active then
    p := the last intersection point incident to l;
    K := K ∪ {p};
  end
end
```

Observation 1: When intersection points incident to a line segment st are sorted in a non-decreasing lexicographic order according to their (x-coordinate, y-coordinate, rank), a maximal overlapping segment on st can take one of the following forms only:

1. A tangent point (e.g., the point f in Fig. 1).
2. If an entry point is followed by an exit point, the line segment between the two is a maximal overlapping segment (e.g., the line segment cd in Fig. 1).
3. If the first intersection point is an exit point, the line segment between s and the first intersection point is a maximal overlapping segment (e.g., the line segment sa in Fig. 1), or
4. If the last intersection point is an entry point, the line segment between the last intersection point and t is a maximal overlapping segment (e.g., the segment gt in Fig. 1).

Algorithm 1 constructs a complete set K that contains exactly one point from every maximal overlapping segment. This algorithm runs in $O(L\,N\log N)$ time, where L is the number of tracks and N is the number of sensor nodes.

3.2. A complete set for the general placement problem

In the general placement problem a data collector can be placed at any point in the sensing field as long as it is within the transmission range of at least one sensor node. To show our method of constructing a complete set for the general placement problem, we use some notations similar to those used in the previous subsection.

Definition 4. An *overlapping region* is a region where the transmission disks of a nonempty subset of sensor nodes overlap. For an overlapping region α, let $S(\alpha)$ denote the subset of sensor nodes whose transmission disks overlap at α.

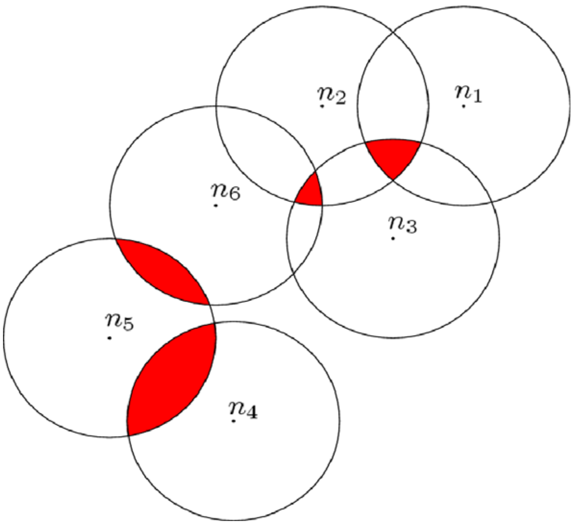


Fig. 2. Maximal overlapping regions.

Definition 5. An overlapping region α is *maximal* if there is no overlapping region β, where $S(\alpha) \subset S(\beta)$.

Fig. 2 shows six sensor nodes and their maximal overlapping regions (MORs).

We next show that a complete set can be derived from the set of MORs. It is obvious that Lemma 1 and Theorem 1 can be applied to the general placement problem as follows.

Lemma 2. *For every overlapping region β, there exists a MOR α, such that $S(\beta) \subseteq S(\alpha)$.*

Theorem 2. *A set K that contains one point from every MOR is complete.*

The proofs of the above lemma and theorem can be directly derived from those of Lemma 1 and Theorem 1. One just needs to observe the analogy between maximal overlapping regions here and maximal overlapping segments in Lemma 1 and Theorem 1.

In order to find all MORs, we need to find the arrangement of transmission disks. Let $D(i)$ denote the transmission disk of sensor node n_i, and if p is an intersection point of the boundaries of $D(i)$ and $D(j)$, let $succ_i(p)$ ($succ_j(p)$) denote the intersection point incident to $D(i)$ ($D(j)$) that comes right after p according to a clockwise order, and let $other_i(p) = j$ and $other_j(p) = i$.

In general, it is possible that the boundaries of more than two transmission disks intersect at the same point as shown in Fig. 3. Let the boundaries of $D(i), D(j)$, and $D(k)$ intersect at a point p. In this case, $other_i(p)$ is defined as follows. Let T_i, T_j, and T_k be the tangent lines of $D(i), D(j)$, and $D(k)$, respectively, at p. Let $\angle(x,p,y)$ denote the magnitude of the anticlockwise angle from T_x to T_y as shown in Fig. 4.[2] $other_i(p)$ is the index of the transmission disk, amongst those that intersect with $D(i)$ at p, whose tangent line at p makes the smallest anticlockwise angle from T_i. In Fig. 3, p is the intersection point of $D(1), D(2)$, and $D(3)$. $other_1(p) = 2, other_2(p) = 3$, and $other_3(p) = 1$.

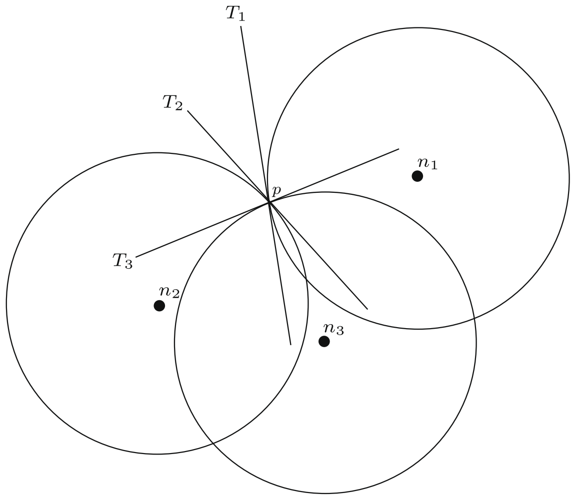


Fig. 3. An intersection point of multiple disks.

The arrangement of disks is a data structure by which for any intersection point p incident to a disk $D(i)$, we can get $succ_i(p)$ and $other_i(p)$ in $O(1)$ time. Such a data structure can be constructed by Algorithm 2, which runs in $O(N^2 \log N)$ time. For each sensor node n_i, Algorithm 2 makes a list L_i that has an entry for each intersection point incident to the boundary of $D(i)$. Each entry in L_i has the following fields:

- *point*: the intersection point (defined by its x- and y-coordinates).
- *other*: the index of a sensor node whose transmission disk intersects with $D(i)$ at *point*.
- *rank*: the magnitude of the anticlockwise angle from T_i to T_{other} at *point* (i.e., $\angle(i, \textit{point}, \textit{other})$).
- *flag*: a Boolean value used to exclude some intersection points.
- *pointer*: a pointer to the entry in L_{other} that is associated with *point*.

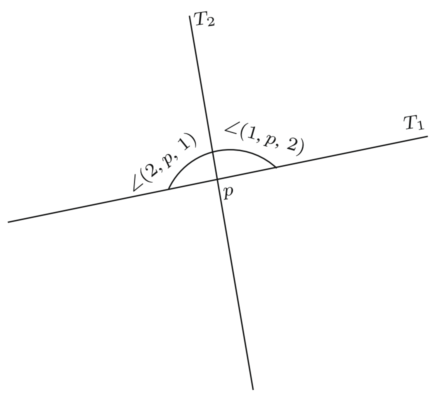


Fig. 4. Angles between tangent lines.

[2] If T_x is parallel to T_y, $\angle(x,p,y) = \angle(y,p,x) = 0$.

Algorithm 2. Arrangement of transmission disks.

Procedure FindArrangement()
Output: For each sensor node n_i, make a sorted list L_i, that contains all intersection points incident to the boundary of $D(i)$.
foreach *sensor node* n_i **do**
  $L_i := \phi$;
end
foreach *sensor node* n_i **do**
  foreach *sensor node* n_j, *where* $j > i$ **do**
    foreach *point p at which the boundaries of* $D(i)$ *and* $D(j)$ *intersect* **do**
      $ListEntry_i.point := p$;
      $ListEntry_i.other := j$;
      $ListEntry_i.rank := \angle(i, p, j)$;
      $ListEntry_j.point := p$;
      $ListEntry_j.other := i$;
      $ListEntry_j.rank := \angle(j, p, i)$;
      $ListEntry_i.pointer := ListEntry_j$;
      $ListEntry_j.pointer := ListEntry_i$;
      insert $ListEntry_i$ in L_i;
      insert $ListEntry_j$ in L_j;
    end
  end
end
foreach *sensor node* n_i **do**
  sort all entries in L_i *in a clockwise order of their corresponding intersection points around the boundary of* $D(i)$. *If two or more entries in* L_i *have the same point fields (i.e., the same coordinates), sort these entries according to a non-decreasing order of their rank fields*;
end
foreach *sensor node* n_i **do**
  foreach *point p that has at least one entry in* L_i **do**
    Flag := 0;
    foreach *entry ListEntry associated with p in* L_i **do**
      if (*p is not a tangent point with respect to* $D(i)$ *OR* $i > ListEntry.other$) *AND p is a tangent point with respect to* $D(ListEntry.other)$ **then**
        Flag := 1;
      end
    end
    foreach *entry ListEntry associated with p in* L_i **do**
      $ListEntry.Flag :=$ Flag;
      $TempEntry :=$ *the first entry for p in* $L_{ListEntry.other}$;
      $ListEntry.pointer := TempEntry$;
    end
  end
end
foreach *sensor node* n_i **do**
  foreach *point p that has at least one entry in* L_i **do**
    remove all entries for p in L_i *except the first one*;
  end
end

Note that in Algorithm 2 we consider some details of the data structure being used, yet in Algorithms 3 and 4 we simplify the presentation by ignoring these details. More specifically, in Algorithms 3 and 4 we use the point p itself to refer to an entry for p in a particular list L_i, we use $other_i(p)$ to refer to the *other* field of that entry, we use $succ_i(p)$ to refer to the entry that follows p's entry in L_i, and we use Flag(p, i) to refer to the *flag* field of p's entry in L_i.

By walking over the boundary of a transmission disk $D(i)$ in a clockwise direction, intersection points incident to $D(i)$ can be classified into three groups: *entry points*, *exit points*, and *tangent points*. An entry point p is one at which we enter the transmission disk $D(other_i(p))$. An exit point p is one at which we leave the transmission disk $D(other_i(p))$. An intersection point p is a tangent point if $D(i)$ and $D(other_i(p))$ intersect at p only (i.e., both disks have the same tangent line at p). Note that an intersection point of two disks is either a tangent point on both disks or an entry point with respect to one disk and an exit point with respect to the other disk. Fig. 5 gives an example of four sensor nodes (n_1, n_2, n_3, and n_4) where the intersection points incident to $D(1)$ are classified with respect to $D(1)$. In Fig. 3, p is an exit point with respect to $D(1)$ and it is an entry point with respect to $D(2)$ and $D(3)$.

The arrangement of disks divides the plane into one open face and several closed faces; the closed faces are the overlapping regions. The closed faces are of three types:

1. A disk that does not overlap with any other disks (see part (a) of Fig. 6).
2. A concave overlapping region, i.e., it has at least one concave arc in its boundaries (see part (b) of Fig. 6).
3. A convex overlapping region, i.e., all arcs in its boundaries are convex (see part (c) of Fig. 6).

Now, it is straightforward to realize the following observation.

Observation 2: An overlapping region β is maximal if and only if it satisfies one of the following properties:

1. β is a tangent point.
2. β is a disk that does not overlap with any other disks.

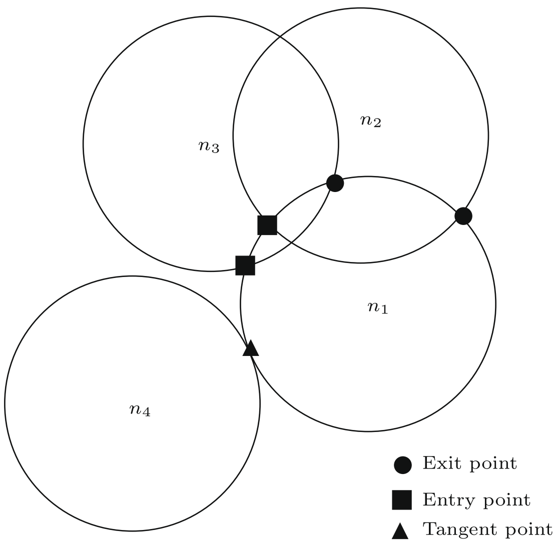
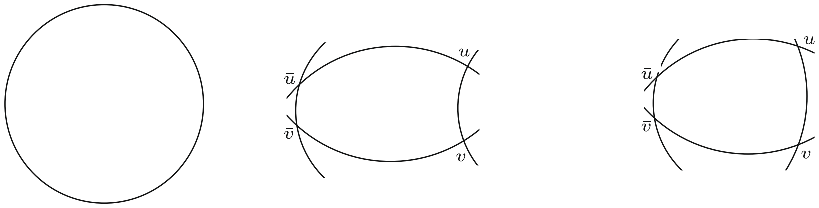


Fig. 5. Entry points, exit points, and tangent points.



(a) Single disk. (b) Concave overlapping region. (c) Convex overlapping region.

Fig. 6. Different types of closed faces.

3. β is a convex overlapping region whose boundaries do not have any tangent points.

Proof. To prove this observation, we need to show that any of these three properties is a sufficient condition for an overlapping region to be maximal (i.e., the if direction) and that neglecting all of them is a sufficient condition for an overlapping region to be non-maximal (i.e., the only if direction).

The if direction:

First: since the tangent point is the overlapping region of two disks that do not overlap anywhere else, a tangent point is a MOR.

Second: a disk that does not overlap with any other disks is a MOR. If it was not a MOR, that disk would overlap with at least one other disk which makes a contradiction.

Third: in order to show that β, which is a convex overlapping region whose boundaries do not have any tangent points, is maximal, it suffices to show that there is no point outside β that lies inside all disks in $S(\beta)$. To do so, let us assume, for the sake of contradiction, that there exists a point u outside β which is inside all disks in $S(\beta)$. Let us take another point v inside β. Since β is convex, the line uv must intersect with the boundaries of β at exactly one point. Let the line uv intersects with the boundaries of β at a point on an arc that is part of the perimeter of a disk $D(i)$. Now, either v is outside $D(i)$ which contradicts with the fact that β is convex or v is inside $D(i)$ and, thus, u is outside $D(i)$, which contradicts with the fact that u is inside all disks in $S(\beta)$ because $i \in S(\beta)$. Therefore, there is no point outside β which is inside all disks in $S(\beta)$ and, hence, β is maximal.

The only if direction:

When none of these three properties is satisfied, we end up with either a concave overlapping region or a convex overlapping region with a tangent point on its boundaries. We prove that both cases result in a non-maximal overlapping region. First, let β be a concave overlapping region whose concave arc belongs to a disk $D(i)$, and let α be the overlapping region on the other side of that concave arc. Obviously, $S(\alpha) = S(\beta) + \{i\}$ and, hence, $S(\beta) \subset S(\alpha)$ which makes β non-maximal. Second, let β be a convex overlapping region with a tangent point u on its boundaries. Then, u must be the intersection point of a disk $D(i)$ and another disk $D(j)$, such that $i \in S(\beta)$ and $j \notin S(\beta)$. Therefore, $S(u) = S(\beta) + \{j\}$ and, hence, $S(\beta) \subset S(u)$ which makes β non-maximal. □

We check whether an overlapping region β is convex or concave as follows. Let uv be an arc on the boundaries of β as shown in Fig. 6b or c, let $\bar{u}$ be the point that comes before u on the boundaries of β according to a clockwise order, and let $\bar{v}$ be the point that comes after v on the boundaries of β according to a clockwise order. Let $\bar{u}u$ be a convex arc belonging to a disk $D(i)$. Obviously, the arc uv is convex if u is an exit point with respect to $D(i)$, and it is concave if u is an entry point with respect to $D(i)$. By applying this test to all arcs that make the boundaries of β, we can tell whether β is convex or concave.[3]

Algorithm 3. Testing whether an overlapping region is maximal or not.

Function Maximal(n_i: a sensor node, p: a point)
Input: A sensor node n_i and an intersection point p incident to $D(i)$.
Output: True if a MOR is found, and False otherwise.
if *Flag*$(p, i) = 1$ *OR p is an entry point with respect to* $D(i)$ **then**
  return False;
end
if *p is a tangent point with respect to* $D(i)$ **then**
  Flag$(p, other_i(p))$:= 1;
  return True;
end
Flag(p, i) := 1;
q := p;
j := $other_i(p)$;
p := $succ_j(p)$;
i := j;
While $p \neq q$ **do**
  if *Flag*$(p, i) = 1$ *OR p is an entry point with respect to* $D(i)$ *OR p is a tangent point with respect to* $D(i)$ **then**
  return False;
  end
  Flag(p, i) := 1;
  j := $other_i(p)$;
  p := $succ_j(p)$;
  i := j;
end
return True;

[3] This test is initialized by a convex arc uv, where u and v are two intersection points incident to the same disk $D(i)$ and $succ_i(u) = v$.

Algorithm 4. Finding all MORs.

Procedure FindMORs()
Output: A set K that contains one point from every MOR.
$K := \phi$;
FindArrangement();
foreach sensor node n_i **do**
  if *D(i) has no intersection points* **then**
    $K := K \cup \{loc_i\}$;
  else
    foreach intersection point p incident to D(i) **do**
      if $Maximal(n_i, p)$ **then**
        $K := K \cup \{p\}$;
      end
    end
  end
end

Algorithm 3 uses Observation 2 to test whether an overlapping region is maximal or not. Algorithm 4 uses Algorithm 2 and Algorithm 3 to find all MORs. Note that the Boolean value $Flag(p, i)$ (i.e., the field *flag* in L_i) is used to guarantee that we do not check the same overlapping region again and to make sure that every MOR adds exactly one point to the complete set. The overall complexity of Algorithm 4 is $O(N^2 \log N)$.

3.3. MILP formulation

When a complete set $K = \{k_0, k_1, \ldots, k_{M-1}\}$ is found, the problem of finding the optimal locations of R data collectors and the flow paths from sensor nodes to data collectors can be formulated as a MILP. Note that once we obtain a complete set, the on-track and the general placement problems become the same; that is because in the two problems, data collectors are to be placed only at points in the complete set. The two problems differ, however, in the algorithm that finds complete sets as we have seen in the previous two subsections. Therefore, the MILP formulation we present here is applicable to the two problems.

Now, we define the following constants and variables.

Constants:

G_i is the data generation rate of sensor node n_i (i.e., the number of data units generated by node n_i per round).
$N(i)$ is a set of indices such that $j \in N(i)$ if n_j is within the transmission range of n_i (i.e., n_j is a neighbor of n_i).
$M(i)$ is a set of indices such that $j \in M(i)$ if the point k_j, which belongs to a point in the complete set, is within the transmission range of n_i.
$B(j)$ is a set of indices such that $i \in B(j)$ if the point k_j, which belongs to a point in the complete set, is within the transmission range of n_i.
N is the number of sensor nodes.
R is the number of data collectors.
M is the size of the complete set (i.e., $|K|$).
E_i is the residual energy of sensor node n_i at the beginning of the round.
E_{Tr} is the energy consumed to send one data unit.
E_{Rc} is the energy consumed to receive one data unit.
C_i is the traffic capacity of sensor node n_i (i.e., the maximum number of data units that can be relayed by n_i per round).
α is the weight assigned to the minimum residual energy.
β is the weight assigned to the total consumed energy.

Variables:

$d_i = 1$ if a data collector is located at k_i, which belongs to a point in the complete set, and $d_i = 0$ otherwise.
If $j \in N(i), f_{ij}$ is the flow from sensor node n_i to sensor node n_j (i.e., the number of data units to be sent from n_i to n_j per round).
If $j \in M(i), h_{ij}$ is the flow from sensor node n_i to the data collector at k_j (if no data collector is placed at k_j, h_{ij} will be set to 0).
E_{min} is the minimum residual energy over all sensor nodes at the end of the round.
E_{total} is the total consumed energy during the round.

Our policy of maximizing the lifetime is to maximize the minimum residual energy (i.e., E_{min}) at the end of each round. However, it is easy to see that more than one solution may have the same optimal value for E_{min} but possibly different values for E_{total}. Amongst those solutions for which E_{min} is maximized, we want to pick the one with the minimum E_{total}. In order to do so, we have the following objective function:

$$\alpha\, E_{min} - \beta\, E_{total},$$

which is composed of a linear combination of E_{min} and E_{total} with a much higher weight given to E_{min} (i.e., $\alpha \gg \beta$). We use

$$\alpha = 1 \quad \text{and} \quad \beta = \frac{1}{\sum_{0 \leqslant i < N} E_i}$$

(note that E_i is constant, $0 \leqslant i < N$). So that $0 < \beta E_{total} < 1$ and, hence, any increase to E_{min} dominates any decrease to E_{total}.

While other objective functions can be easily integrated in our MILP formulation, we claim that residual energy must be taken into account in order to balance the energy levels over the network and, hence, prolong its lifetime. For example, nodes with low energy supply should be relieved from relaying data for other nodes.

A similar MILP formulation was introduced in [1]. Yet the authors in [1] assume the existence of a set of feasible locations (i.e., the set K which we construct in the previous subsections). Furthermore, our objective function is different from the one in [1]. Our scheme prolongs the lifetime by maximizing the minimum residual energy over all nodes at the end of each round. Including the work in [1], several existing proposals for different routing and placement problems aim at prolonging the lifetime by minimizing the total consumed energy or minimizing the maximum amount of energy consumed by an individual node. We argue that these schemes are not really suitable for the placement of mobile data collectors. This is because the optimal solution towards such objectives will not change over time and, hence, will not change the locations of data collectors. Residual energy must be taken into account for any mobile data collector scheme as it is the only attribute that changes over time, and the only factor that makes such mobility desirable. In our scheme a MILP that reflects our policy is to be solved at the beginning of each round in order to move data collectors to new locations.

The MILP is shown in Fig. 7. Eq. (1) satisfies the traffic capacity constraints and Eq. (2) guarantees the flow balance. Eq. (3) makes E_{min} the minimum residual energy over all sensor nodes (note that we maximize E_{min}). Eq. (4) sets E_{total} to the total energy consumption. Eq. (5) guarantees that the energy expenditure of any sensor node is not more than its current residual energy. Eq. (6) guarantees that if no data collector is located at k_j (i.e., $d_j = 0$), no flow is sent to k_j. Eq. (7) satisfies the constraint that only R data collectors are available. This MILP is supposed to be solved at the beginning of each round in order to move data collectors to new locations.

The time complexity of our approach is $O(N^2 \log N)$ time to find a complete set plus the time to solve a MILP. While MILP problems are NP-hard, there exist several advanced algorithms (e.g., branch-and-bound, branch-and-cut, and branch-and-price), which can be used to find near-optimal solutions [8]. For small MILPs, these algorithms are able to find the optimal solution within a reasonable time. For relatively large MILPs, one can use any of these algorithms, let it run and search for better solutions for a given amount of time, and then take the best solution it has found. In our experiments, we were able to obtain very good solutions (as compared with the best existing schemes) within reasonable time (10–20 min) for networks of up to 200 sensor nodes and five data collectors. With the fact that these MILPs are to be solved once every round (and the round lasts for at least a couple of hours), 20 min is not too long; during the last 20 min of a round, the MILP of the next round is solved.

4. Simulation results

We compare our proposed schemes with two other schemes: a static scheme where data collectors are stationary and a mobile scheme that ignores the residual energy of different sensor nodes. In the static scheme, data collectors are placed randomly in the sensing field, and we use a similar MILP to find near-optimal flow paths. While our schemes have the objective of Maximizing the minimum Residual energy (MR), the mobile scheme we compare with Minimizes the Maximum energy (MM) consumed by a single sensor node (this objective has been used in several routing and placement problems, such as the work in [1]).

We use the general energy consumption model presented in [4] which can be described as follows.

$$E_{Tr}(r,b) = b \times (e_{elec} + e_{amp} \times r^{\gamma}) \tag{8}$$

$$E_{Rc}(b) = b \times e_{elec} \tag{9}$$

where $E_{Tr}(r,b)$ is the energy consumed to send b bits over r (m), $E_{Rc}(b)$ is the energy consumed to receive b bits, e_{elec} is

Maximize $\alpha E_{min} - \beta E_{total}$
s.t.,

$$\sum_{j\in N(i)} f_{ij} + \sum_{j\in M(i)} h_{ij} \le C_i, 0 \le i < N \tag{1}$$

$$\sum_{j\in N(i)} f_{ij} + \sum_{j\in M(i)} h_{ij} - \sum_{j\in N(i)} f_{ji} = G_i, 0 \le i < N \tag{2}$$

$$E_i - E_{Tr}\Big(\sum_{j\in N(i)} f_{ij} + \sum_{j\in M(i)} h_{ij}\Big) - E_{Rc}\sum_{j\in N(i)} f_{ji} \ge E_{min} \quad ,0 \le i < N \tag{3}$$

$$\sum_{0\le i<N}\Big(\sum_{j\in N(i)} E_{Rc} f_{ji} + \sum_{j\in N(i)} E_{Tr}\, f_{ij} + \sum_{j\in M(i)} E_{Tr} h_{ij}\Big) = E_{total} \tag{4}$$

$$E_{min} \ge 0 \tag{5}$$

$$\sum_{i\in B(j)} h_{ij} \le d_j \sum_{0\le i<N} G_i, 0 \le j < M \tag{6}$$

$$\sum_{0\le i<M} d_i = R \tag{7}$$

$$d_i \in \{0,1\}, 0 \le i < M$$

Fig. 7. MILP formulation.

the energy consumed by the transmitter (receiver) to send (receive) one bit, e_{amp} is the energy consumed by the transmission amplifier for one bit, and γ is the path-loss exponent. In our simulation, r is set to 50 (m), e_{elec} is set to 50 nJ/bit, e_{amp} is set to 0.1 nJ/bit/$(\text{m})^2$, and γ is set to 2. The packet size is 512 bits. Every sensor node has an initial energy of 6 J. Data generation rates are uniformly distributed between 100 and 200 packets/round.

Our simulations involve networks of 200 sensor nodes randomly deployed in a 300 (m) × 300 (m) field. To add tracks to the sensing field, we generate 10 points uniformly distributed over the sensing field, and we construct the *relative neighborhood graph* (RNG) of these points [11]. The edges of the resulting RNG are used as tracks spanning the field. In each network, we tested different scenarios of one, three, and five data collectors. To solve the MILP, we use *lp_solve* 5.5 [12] with a timeout of 20 min.

Fig. 8 shows the lifetime comparison between our MR scheme, the static scheme, and the mobile MM scheme for the on-track placement problem. Fig. 9 shows a comparison of the average energy consumed per bit between the three schemes for the on-track placement problem.

Fig. 10 shows the lifetime comparison between the three schemes for the general placement problem. Fig. 11 shows a comparison of the average energy consumed per bit between the three schemes for the general placement problem.

It can be observed that the three schemes have similar energy consumption per bit, yet our MR scheme has a much longer network lifetime. The lifetime improvement of our MR scheme over the static scheme is in the order of 250%. This is due to the load balancing effect that is achieved by exploiting data collectors mobility. The MR scheme also makes about 40% lifetime improvement over the MM scheme. This is a direct result of the consideration made for residual energy at single nodes. One of the interesting results that we have is that the MR scheme may consume a little more energy than the MM scheme (as shown in Figs. 9 and 11), yet the MR scheme makes a longer network lifetime (as shown in Figs. 8 and 10). The reason is that the MR scheme may choose a route or a data collector location that spends more energy for the sake of avoiding and reducing the load on sensor nodes with low residual energy. An energy-efficient route or data collector location that puts extra load on a sensor node with a critical energy level is not preferred in the MR scheme.

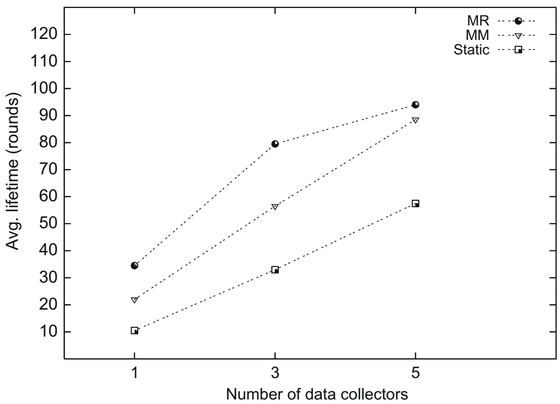


Fig. 8. Lifetime comparison for the on-track placement problem.

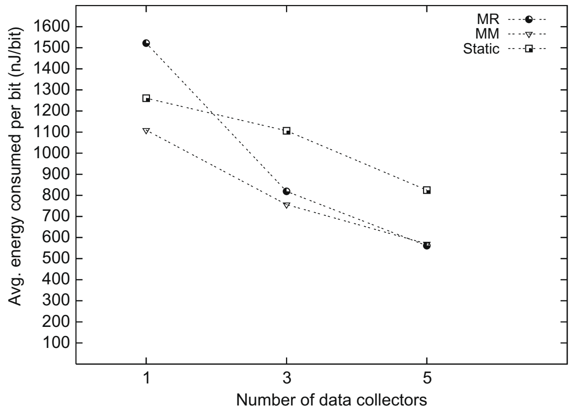


Fig. 9. Energy consumption comparison for the on-track placement problem.

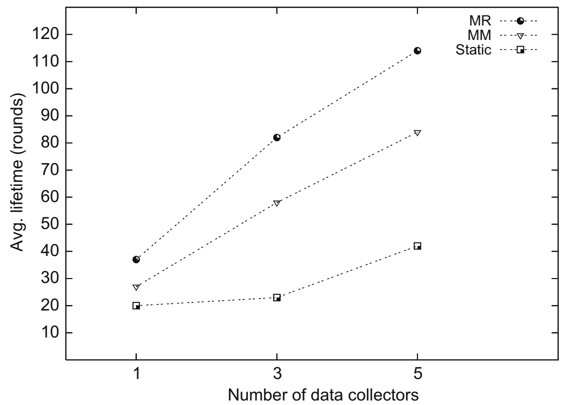


Fig. 10. Lifetime comparison for the general placement problem.

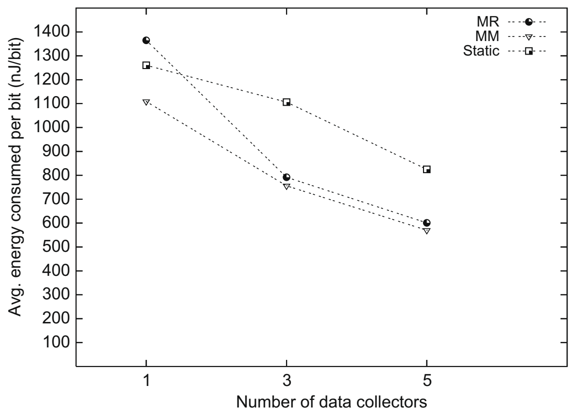


Fig. 11. Energy consumption comparison for the general placement problem.

Using different values for the MILP solver timeout, different sensor densities, and different number of data collectors have shown similar trends.

Figs. 12–19 show an example showing how energy is depleted in a network of 64 nodes and 2 data collectors using the MR and MM schemes with a general placement. Each plot shows the residual energy of every sensor node for each scheme at the end of a round. In each plot, every sensor node is represented by a square and a circle: the height of the square (circle) reflects the residual energy

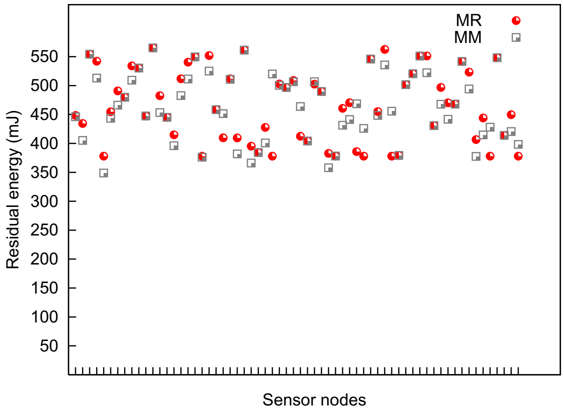


Fig. 12. Round No. 1.

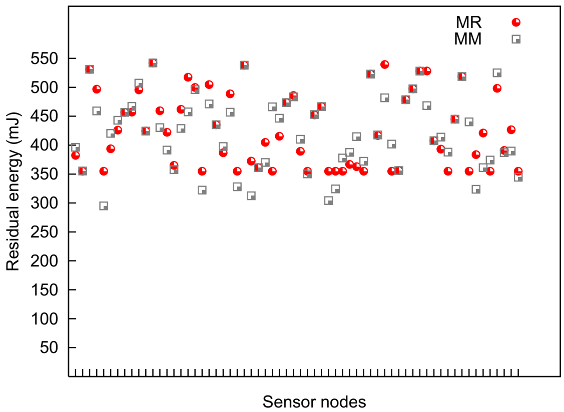


Fig. 13. Round No. 2.

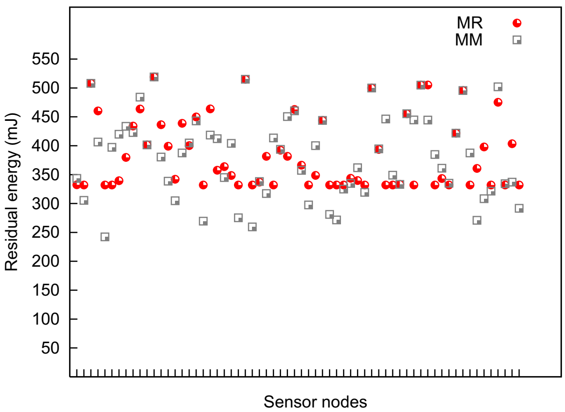


Fig. 14. Round No. 3.

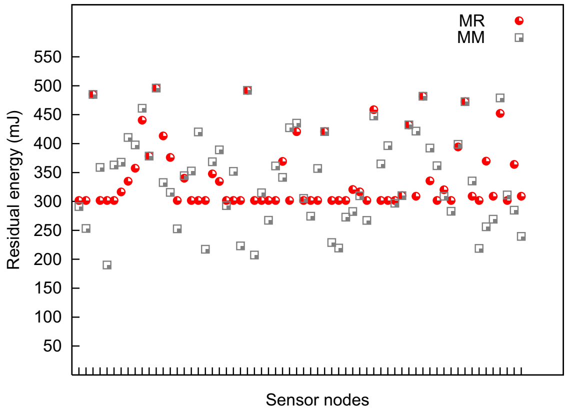


Fig. 15. Round No. 4.

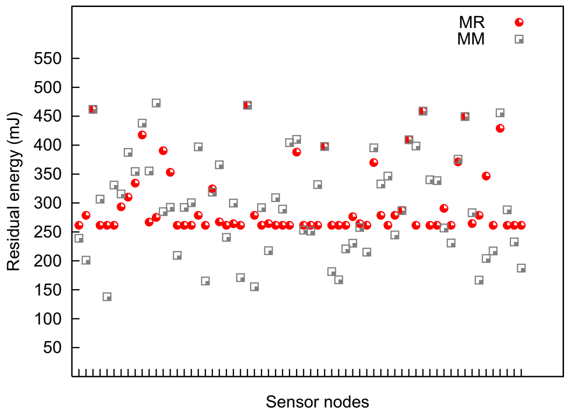


Fig. 16. Round No. 5.

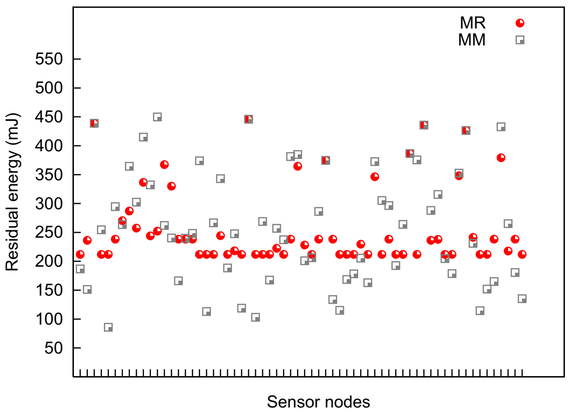


Fig. 17. Round No. 6.

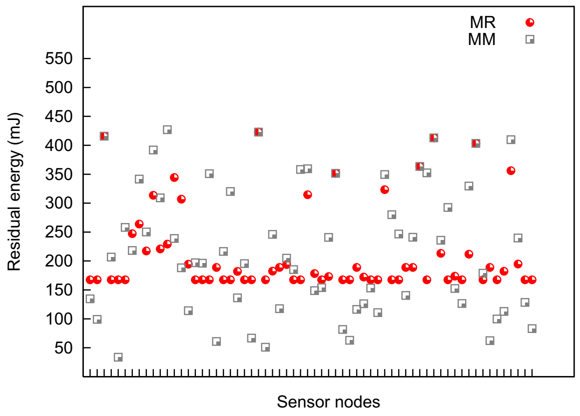


Fig. 18. Round No. 7.

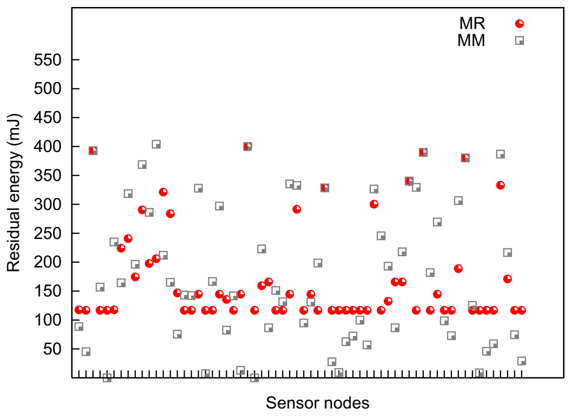


Fig. 19. Round No. 8.

of the corresponding sensor node at the end of the round using the MM (MR) scheme. Note that the MR scheme tries to maintain a bound on the minimum residual energy which helps in balancing the residual energy amongst all sensor nodes and, hence, prolonging the lifetime of the network. For example, at the end of the fifth round, the lowest energy level using the MR scheme is 261 mJ and the lowest energy level using the MM scheme is 138 mJ. Furthermore, at the end of the last round, the lowest energy level using the MR scheme is 116 mJ and 37.5% of the nodes using the MM scheme have energy levels lower than 116 mJ.

5. Conclusion

In this paper, we address the problem of unbalanced energy expenditure in WSNs resulting from using a stationary sink and multi-hop relaying. To alleviate the effect of this problem, we argue for using multiple, mobile data collectors, and propose a scheme for finding near-optimal placement of mobile data collectors together with the routing patterns to deliver data to data collectors. The novelty of our approach stems from:

1. Solving a general problem in which a data collector can be placed anywhere in the sensing field and another problem in which data collectors can move along and be placed on tracks spanning the sensing field.
2. Finding a complete, finite search space for data collector locations for each problem.

Linear programming is used to find near-optimal solutions from the obtained search space. We use an objective function that takes into account both the current residual energy and future energy expenditure of each sensor node. Experimental results show that our scheme has the potential to prolong the lifetime of a WSN significantly as compared with another static scheme and a mobile scheme that does not consider the residual energy of nodes.

We are currently extending our scheme to a 3D Underwater WSN where data collectors buoy on the surface of the water and sensor nodes float at different depths underwater. We also consider other dimensions to the problem: variable transmission range and bounded delay.

Acknowledgements

We thank Robert Benkoczi for his helpful discussions and insightful comments. We also thank the Natural Sciences and Engineering Research Council of Canada and King Saud University in Saudi Arabia for their support.

References

[1] S. Gandham, M. Dawande, R. Prakash, S. Venkatesan, Energy efficient schemes for wireless sensor networks with multiple mobile base stations, in: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM), December 2003.
[2] A. Azad, A. Chockalingam, Mobile base stations placement and energy aware routing in wireless sensor networks, in: Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), April 2006.
[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Personal Communications Magazine 40 (8) (2002) 102–114.
[4] W. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, in: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, January 2000.
[5] S. Olariu, I. Stojmenovic, Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting, in: Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM), April 2006.
[6] X. Wu, G. Chen, S. Das, On the energy hole problem of nonuniform node distribution in wireless sensor networks, in: Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS), October 2006.
[7] Y. Shi, Y.T. Hou, A. Efrat, Algorithm design for base station placement problems in sensor networks, in: Proceedings of the 3rd International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine), August 2006.
[8] C. Papadimitriou, K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Dover Publications, 1998.
[9] N. Priyantha, A. Chakraborty, H. Balakrishnan, The cricket location-support system, in: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, August 2000.
[10] N. Bulusu, J. Heidemann, D. Estrin, GPS-less low cost outdoor localization for very small devices, IEEE Communications Magazine 7 (5) (2000) 28–34.
[11] J. Jaromczyk, G. Toussaint, Relative neighborhood graphs and their relatives, Proceedings of the IEEE 80 (9) (1992) 1502–1517.
[12] http://lpsolve.sourceforge.net/5.5/.

Waleed M. Alsalih received his Ph.D. degree from the School of Computing at Queen's University, Ontario, Canada in 2009. He is currently an assistant professor in the Computer Science Department at King Saud University. Waleed's research interests include wireless sensor networks, ad hoc networks, and energy-aware design.

Hossam S. Hassanein is a leading researcher in the School of Computing at Queen's University in the areas of broadband, wireless and variable topology networks architecture, protocols, control and performance evaluation. Before joining Queen's University in 1999, he worked at the department of Mathematics and Computer Science at Kuwait University (1993–1999) and the department of Electrical and Computer Engineering at the University of Waterloo (1991–1993). Dr. Hassanein obtained his Ph.D. in Computing Science from the University of Alberta in 1990. He is the founder and director of the Telecommunication Research (TR) Lab http://www.cs.queensu.ca/~trl in the School of Computing at Queen's. Dr. Hassanein has more than 300 publications in reputable journals, conferences and workshops in the areas of computer networks and performance evaluation. Dr. Hassanein has organized and served on the program committee of a number international conferences and workshops. He also serves on the editorial board of a number of International Journals. He is a senior member of the IEEE and serves as the Secretary of the IEEE Communication Society Technical Committee on Ad hoc and Sensor Networks (TC AHSN). Dr. Hassanein is the recipient of Communications and Information Technology Ontario (CITO) Champions of Innovation Research award in 2003. In March 2007, he received a best paper award at the IEEE Wireless Communications and Networks (a flagship IEEE communications society conference).

Selim G. Akl received his Ph.D. degree from McGill University in Montreal in 1978. He is currently a Professor of Computing at Queen's University, Kingston, Ontario, Canada. His research interests are in parallel computation. He is author of Parallel Sorting Algorithms (Academic Press, 1985), The Design and Analysis of Parallel Algorithms (Prentice Hall, 1989), and Parallel Computation: Models and Methods (Prentice Hall, 1997), and a co-author of Parallel Computational Geometry (Prentice Hall, 1992). Dr. Akl is editor in chief of Parallel Processing Letters and presently serves on the editorial boards of Computational Geometry, the International Journal of Parallel, Emergent, and Distributed Systems, and the International Journal of High Performance Computing and Networking.