2014

Energy efficient routing towards a mobile sink using virtual coordinates in a wireless sensor network

Rouhollah Rahmatizadeh
University of Central Florida

ENERGY EFFICIENT ROUTING TOWARDS A MOBILE SINK USING VIRTUAL COORDINATES IN A WIRELESS SENSOR NETWORK

by

ROUHOLLAH RAHMATIZADEH
B.S. Sharif University of Technology, 2012

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science
in the Department of Electrical Engineering and Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2014

Major Professor: Lotzi Bölöni

ABSTRACT

The existence of a coordinate system can often improve the routing in a wireless sensor network. While most coordinate systems correspond to the geometrical or geographical coordinates, in recent years researchers had proposed the use of *virtual coordinates*. Virtual coordinates depend only on the topology of the network as defined by the connectivity of the nodes, without requiring geographical information.

The work in this thesis extends the use of virtual coordinates to scenarios where the wireless sensor network has a mobile sink. One reason to use a mobile sink is to distribute the energy consumption more evenly among the sensor nodes and thus extend the life-time of the network. We developed two algorithms, MS-DVCR and CU-DVCR which perform routing towards a mobile sink using virtual coordinates. In contrast to the baseline virtual coordinate routing MS-DVCR limits routing updates triggered by the sink movement to a local area around the sink. In contrast, CU-DVCR limits the route updates to a circular area on the boundary of the local area. We describe the design justification and the implementation of these algorithms. Using a set of experimental studies, we show that MS-DVCR and CU-DVCR achieve a lower energy consumption compared to the baseline virtual coordinate

routing without any noticeable impact on routing performance. In addition, CU-DVCR provides a lower energy consumption than MS-DVCR for the case of a fast moving sink.

To my parents and whomever taught me something!

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

Wireless sensor networks (WSNs) consist of a large number of sensor nodes capable of continuously and cooperatively monitoring their surrounding environment. Sensor nodes usually send the sensed information using hop-by-hop communication to one or few collecting nodes called the sink. Sensor networks have many applications such as environmental monitoring, military surveillance, traffic control, and ambient conditions detection. Sensor nodes in a WSN use a limited source of energy such as a battery which is hard to be replaced in some applications. This opens a challenge for the researchers to design energy-optimized routing protocols to extend the lifetime of the network as much as possible.

In a multi-hop network, messages traverse a hop-by-hop path from the source to the destination. In this scenario, since every message should pass the nodes around the sink, these nodes drain their energy faster than other nodes in the network [19]. The mostly used definition of network lifetime is the time until the first node exhausts its energy. Considering this definition, one solution to extend the lifetime of a WSN is to move the sink to the areas in which nodes have more energy [18].

Using a mobile sink is useful for saving energy only when we have an efficient routing strategy towards it. Geographical coordinate system relies on the geographical location

information of the nodes to do the routing. To obtain the location of the nodes, we need GPS which is costly and infeasible in some applications. In contrast, Virtual Coordinate System (VCS) does not rely on GPS information of the nodes. Instead, it is based upon hop-by-hop distance information from a few anchor points [32]. In addition, there exist routing algorithms using VCS as effective as geographical routing algorithms. However, to the extent of our knowledge, there is no routing algorithm towards a mobile sink using virtual coordinates.

In this thesis, we design two energy-efficient routing protocols towards a mobile sink using virtual coordinate system.

1.1 Motivation

Virtual coordinate system provides most advantages of geographical coordinate system without requiring GPS location information of the nodes. On the other hand, using a mobile sink seems to be a good choice to prolong the lifetime of the network. However, we could not find a routing algorithm to a mobile sink using virtual coordinates in the literature. Hence, we found ourselves motivated to extend virtual coordinates to include routing towards a mobile sink.

1.2 Contribution

We propose two algorithms in virtual coordinate system to enable use of a mobile sink without a need to update the entire network. In more details, our contributions are:

- MS-DVCR, a routing algorithm towards a mobile sink based on the idea of limiting the update notification of the sink to a local area instead of the entire network. This algorithm saves a significant amount of energy compared to the naive idea of updating the entire network when the sink moves. The details of this algorithm is described in Chapter 4.

- CU-DVCR, an improvement to MS-DVCR with the idea of limiting the broadcast to a circular area around the sink. In fact, the circular area in CU-DVCR is the boundary of the local area in MS-DVCR. CU-DVCR can save more energy than MS-DVCR in case the sink is not moving extraordinarily fast. CU-DVCR is implemented in virtual coordinates, However, it can be generalized simply and be implemented in geographical domain with little effort. The details of this algorithm is described in Chapter 5.

- A set of experiments to show the effectiveness of the algorithms in conserving energy while maintaining the performance. In the last experiment, we show that there is an optimal size for the local area in which the energy consumption is minimized. More details are discussed in Chapter 6

1.3 Organization

The remaining of this thesis is organized as follows. Chapter 3 explains the structure of virtual coordinate system and how to transform it to retrieve a sense of directionality between two nodes. Then, DVCR routing algorithm based on the transformed virtual coordinates is explained. The MS-DVCR algorithm is presented in Chapter 4. The CU-DVCR algorithm is presented in Chapter 5. We compare and investigate different aspects of the algorithms using a series of experiments and simulation studies in Chapter 6. Finally, we conclude in Chapter 7.

CHAPTER 2
RELATED WORK

In this chapter, we discuss the related literature for both virtual coordinates and the sink mobility in wireless sensor networks.

2.1 Virtual coordinates literature

The most often used coordinate system for routing relies on the actual geographical coordinates of the nodes [2]. Having the geographical location information of the nodes, greedy forwarding can be used by each node to forward the packets to the neighbor which is the closest one to the destination. However, in networks with holes due to voids in sensor deployment or failure of some nodes, greedy forwarding fails [1]. In these networks a node can be nearer than all of its neighbors to the destination. Researchers have tried to come up with a solution for this known issue of greedy forwarding called local minimum. Greedy Perimeter Stateless Routing (GPSR) [23] is one of the pioneers to solve this problem. GPSR suggests that when a message encounters a local minimum, it should follow the perimeter of the planar graph to escape the local minimum.

In [20], directed diffusion, a data aggregation paradigm for WSNs is proposed. The main idea behind Directed diffusion is to combine the data coming from different sources

by eliminating redundancy, minimizing the number of transmissions, thus saving network energy and prolonging its lifetime. Rumor routing [5] is a variation of directed diffusion and is useful in applications where there is no geographical information available to do geographical routing.

Providing all the nodes with GPS system is expensive and infeasible in some applications. Hence, researchers have suggested networks in which only a few nodes know their location. In these networks, algorithms based on triangulation or multilateration can work well [6]. For example, in [34] a system called AHLoS (Ad-Hoc Localization System) is introduced that enables sensor nodes to discover their locations using an iterative algorithm. In this algorithm, nodes with unknown locations use ranging information and known location of beacon nodes in their neighborhood to estimate their location. Once their location is estimated, they turn into a beacon, thus, other nodes can use this information to estimate their own location.

Eliminating the need for GPS-equipped sensor nodes is favorable if we achieve equivalent performance. With the introduction of Virtual Coordinates Systems (VCS) [8] [10] the need for costly localization was resolved. In contrast to geographical coordinates, nodes are not aware of their geographical position in this coordinate system. Instead, virtual coordinates of a node is defined as its hop-by-hop distance to a set of nodes called anchors. For the nodes to acquire their virtual coordinates, network-wide flooding [7] is suggested. In this method, there is an initial setup phase in which anchors broadcast their coordinates to the

entire network. As a result, each node becomes aware of its coordinates corresponding to every anchor in the network. In another method [17], rumor routing was used to avoid a separate network setup phase. Using this technique, messages are being forwarded to random neighbors and carry the coordinates of the anchors. After some time, all the nodes become aware of the location of the anchors.

When the nodes have their virtual coordinates, they can do routing. Convex Subspace Routing (CSR) [12] is one of the routing algorithms which uses virtual coordinates. In contrast to earlier routing algorithms which usually rely on backtracking when the message encounters a local minimum, CSR dynamically selects subsets of anchors which do not cause local minimum problem. These set of anchors have the attribute of providing a convex distance function from the source to the destination. Directional Virtual Coordinate Routing(DVCR) [16] is another routing algorithm using virtual coordinates which generally outperforms some routing algorithms in geographical domain such as GPSR. We will discuss DVCR algorithm in details in Chapter 3.

In [13] researchers tried to reduce the dimension of virtual coordinates while preserving routability. Their method is based on Singular Value Decomposition (SVD) and uses novelty filtering to select effective anchors prior to SVD based compression. Using some experiments with different topologies and 40 anchors, they showed that coordinate length can be reduced by a factor of 8 on average.

In DVCR algorithm the subset of sensor nodes chosen as anchors plays an important role in routing performance. A subset of nodes can be automatically found using an algorithm [15] which significantly improves the performance of routing in most of virtual coordinate based routing schemes.

In virtual coordinates, we do not have access to geographical location information of the nodes. Hence, we cannot have a geographical map of the network. However, we can retrieve the topology preserving maps of the network using an algorithm [14] using the virtual coordinates of the sensor nodes. Given the topology preserving maps of the network, researchers have suggested a method to accurately track an predict the mobility in virtual coordinates [21].

2.2 Sink mobility literature

The simple and traditional form of a wireless sensor network consists of some sensors and a single sink node with a fixed position. However, using mobile sinks is suggested in the literature to prolong the network lifetime. In [39] the performance and trade-offs associated with the cases of using a mobile sink and making the network all static are investigated. The results show that in some situations mobile relays or mobile sinks can be used to improve network lifetime.

Using robots as mobile sinks solved the problem of practicality of deploying such a system as suggested in [24]. In this work, a network with robots as mobile sinks is deployed to care for houseplants in the office environment.

In [9] saving power in sensor networks based on predictable mobility of the observer (mobile sink) is investigated. They indicate that their model fits well for public transportation vehicles since their movement is predictable. They results show that the power savings over a static sensor network are significant.

The idea of exploiting the sink mobility to increase the lifetime of a WSN is also investigated in [40]. In WSNs with up to 256 nodes, their model produces sink movement patterns which lead to a network lifetime up to almost five times longer than a network with a static sink. They suggest that in a network with a static sink, the nodes in the proximity of the sink drain their energy faster than other nodes.

Mobility is sometimes enforced by the requirement of the application. For example, in [4] an underwater sensor network is investigated in which an autonomous underwater vehicle is responsible for collecting data from the nodes throughout the network.

Routing towards mobile sinks in an energy optimized way is emphasized in the literature. Considering the fact that mobile sink is mostly used to increase the lifetime of the network, an energy optimized routing towards a mobile sink seems to be essential to reach that goal. In [27] mobility is suggested to improve the lifetime of the network. First, they show that in a circle shaped network, the best mobility strategy might be for the sink to follow the periphery

of the network. Second, they suggest that considering this experimental conditions, a better routing strategy uses a combination of round routes and short paths.

MobiRoute [28] considers the problem of practicality of routings which support sink mobility and investigates an approach to use a mobile sink to balance the traffic load and thus improve the lifetime of the network. They consider different scenarios with nodes located in point lattices and a special in-building network with nodes forming a ring. Their results show that MobiRoute in most cases, improves the network lifetime with only a modestly degraded reliability in packet delivery.

A three-tier architecture for collecting sensor data in sparse sensor networks using mobile nodes is presented in [35] which assumes random walk as the mobility pattern. This work focuses on a simple analytical model for understanding performance as system parameters such as number of mobile elements. In a more formal work [30], researchers have defined the sink mobility problem in a linear programming form. They divide the problem of maximizing the lifetime of the network into two subproblems: a scheduling problem that determines the sojourn times of the sink at different locations, and a routing problem in order to deliver the sensed data to the sink in an energy-efficient way. They claim that their model provides the optimal solution to these problems and thus gives the best achievable network lifetime.

Large scale sensor networks need extra considerations for energy efficient routing especially when mobile sinks are being used. An architecture for large scale sensor networks is proposed in [36] which uses mobile agents. They compare their architecture with a static ad

hoc sensor network using some experiments and show a substantial gain in energy efficiency. In another work Two-Tier Data Dissemination (TTDD) [26] is proposed to solve the problem of collecting data in large scale WSNs with mobile sinks. Their Two-Tier Data Dissemination approach provides scalable and efficient data delivery to multiple mobile sinks.

A simple solution to the problem of routability is for a mobile sink to consecutively inform the sensors of its new location when the movement occurs. Declarative Routing Protocol (DRP) [11] is using the idea in which the sink continuously propagate its location information throughout the entire network as it moves. However, this scenario is not suitable when the sink moves relatively fast and the thus the number of required broadcasts increases.

Integrated Location Service and Routing (ILSR) [25] is a geographical routing protocol towards a mobile sink. In this protocol, the sink floods location updates to sensors in its neighborhood when a link breaks or is created. Hence, this algorithm is suitable for situations in which the sink speed is slow thus the number of required updates is small. The experiments show that ILSR generates routes close to shortest paths.

Local Update-based Routing Protocol (LURP) [37] is a routing protocol towards a mobile sink. In this algorithm, the basic idea is to limit the update notification broadcasted by the sink to a limited local area. This algorithm is proposed for geographical coordinates domain. An extension to LURP called Adaptive Local Update-based Routing Protocol (ALURP) [38] is proposed to reduce the local area size if possible as the sink moves. This technique can work slightly better than LURP in energy savings.

A movement pattern for the sink and its effect on energy consumption is studied in [29]. The suggested strategy is to move the sinks on a predetermined path, along the perimeter of a hexagonal tiling when their neighbor sensors' energy becomes low. Simulations show the effectiveness of this method in extending the lifetime of the network.

CHAPTER 3
VIRTUAL COORDINATES

In this chapter we introduce Virtual Coordinate System(VCS) which is the base for the proposed algorithms in this thesis. We will see how virtual coordinate provides a decent routing algorithm without any need for geographical location information of the nodes in a network.

3.1 General principles

A coordinate system gives every node a unique identifier to make it distinguishable from the other nodes. In virtual coordinates nodes are defined by their hop-count distance to a set of nodes called *anchor nodes*. When a sensor network is deployed, nodes can find their coordinates using a setup phase using network-wide flooding [7]. In another method, the nodes can find their coordinates gradually without any need for a setup phase using Rumor Routing [17].

In hop by hop routing, when a node receives a packet, it should decide to which of its neighbors it should forward the packet. The goal is that the packet should traverse the minimum number of hops before reaching its destination. Greedy forwarding suggests that each node should choose one of its neighbors which is closest to the destination. To compare

the nodes based on how far they are located to the destination, we should have a sense of distance between two nodes in a network. In geographical coordinates, we can use Euclidean distance for this purpose. In virtual coordinate, we should have a similar concept to be able to do routing based on greedy forwarding. The distance metrics in VCs are based on the L^1 or L^2 norms defined over VCs.

One of the problems we face by choosing greedy forwarding as a base for the routing algorithm is local minima. Consider a node which is closer to the destination among its neighbors based on the coordinates we use. This problem can occur in networks of a non-convex shape in both geographical and virtual coordinates but it is more probable in VC domain. As the VCs propagate radially away from the anchors, the L1 and L2 do not provide good estimates of distance in VC systems, causing many local minima and poor routing performance. Therefore, new coordinates are derived from VCs to overcome this disadvantage and generate more Cartesian like coordinate systems. Directional Virtual Coordinate System (DVCS) is such an example.

3.2 Directional Virtual Coordinate System Routing

Directional virtual coordinate routing protocol (DVCR) [16] is a routing protocol based on virtual coordinate system. Since VCs propagate radially away from the anchors, the directionality of coordinates is lost. DVCR applies a mathematical transformation to restore the directionality of the ordinates.

In a sensor network with N nodes and M anchors, $h_{N_iA_j}$ shows the minimum hop distance between node N_i and anchor A_j. Thus, the virtual coordinate of node N_i is $[h_{N_iA_1}, \ldots, h_{N_iA_M}]$. Since $h_{N_iA_j}$ is the same for all the nodes within a certain distance from A_j in all directions, it does not provide a sense of directionality. We should use more than one anchor to decrease the number of nodes with a similar coordinates. If we use two anchors, the nodes on the intersection of the circles centered to anchors will have the same coordinates. Hence, at least three anchors are needed to have a unique coordinate for each node in the network.

To give an intuition of how anchors are used to make a coordinate system, we show a simple one dimensional network with two anchors in Table 3.1 . This simple example illustrates that $h_{N_iA_1} + h_{N_iA_2}$ does not provide directionality for the nodes between the anchors since all the values are the same. On the other hand, $h_{N_iA_1} - h_{N_iA_2}$ is the same for all the nodes between the anchors. So, we need a more complicated formula such as Equation 3.1 which uses both sum and difference to return a sense of directionality for all nodes and have a directional coordinate for a one dimensional network with two anchors:

$$f(h_{N_iA_1}, h_{N_iA_2}) = \frac{1}{2h_{A_2A_1}}(h_{N_iA_1} - h_{N_iA_2})(h_{N_iA_1} + h_{N_iA_2}) \tag{3.1}$$

where $\frac{1}{2h_{A_2A_1}}$ is used for normalization.

Table 3.1: A one dimensional network consisting of consecutively connected nodes: N_1, N_2, A_1, N_3, N_4, A_2, N_5, N_6

	N_1	N_2	A_1	N_3	N_4	A_2	N_5	N_6
$h_{N_iA_1}$	2	1	0	1	2	3	4	5
$h_{N_iA_2}$	5	4	3	2	1	0	1	2
$h_{N_iA_1} + h_{N_iA_2}$	7	5	3	3	3	3	5	7
$h_{N_iA_1} - h_{N_iA_2}$	-3	-3	-3	-1	1	3	3	3
$f(h_{N_iA_1}, h_{N_iA_2})$	-3.5	-2.5	-1.5	-0.5	0.5	1.5	2.5	3.5

To define a vector in a direction which starts from an anchor and ends to another anchor, for every two arbitrarily chosen anchors from all the anchors in the network, say A_j and A_k, lets define vector $\vec{f}(h_{N_iA_j}, h_{N_iA_k})$ to be:

$$\vec{f}(h_{N_iA_j}, h_{N_iA_k}) = f(h_{N_iA_j}, h_{N_iA_k})\vec{u}_{A_jA_k} \tag{3.2}$$

where $\vec{u}_{A_jA_k}$ is called the virtual direction and is the unit vector in direction of A_jA_k. The magnitude of the vector $f(h_{N_iA_j}, h_{N_iA_k})$ is given as

$$f(h_{N_iA_j}, h_{N_iA_k}) = \frac{1}{2h_{A_jA_k}}\left(h^2_{N_iA_j} - h^2_{N_iA_k}\right) \tag{3.3}$$

The virtual distance between two nodes N_p and N_q in this direction would be defined as:

$$F_{A_jA_k}(N_p, N_q) = f(h_{N_pA_j}, h_{N_pA_k}) - f(h_{N_qA_j}, h_{N_qA_k}) \tag{3.4}$$

Let us now introduce a definition of the distance metric used in DVCR. Let suppose that the transformed ordinates of the source node x and the sink node y are given as $N_x \equiv [n_{x1} \cdots n_{xy} \cdots n_{xP}]$ and $N_y \equiv [n_{y1} \cdots n_{xy} \cdots n_{yP}]$. Here P is the cardinality of the ordinates and can be selected from C_2^M combinations given M randomly select anchors. Using the L^2 distance between the source and the destination nodes we find the distance as:

$$D_{N_xN_y} = \sqrt{\sum_{\forall j}(n_{xj} - n_{xy})^2}; j = 1 : J \leq C_2^M \tag{3.5}$$

This distance metric allows us to perform *greedy forwarding:* when a node needs to transmit a message to the destination (usually, the sink), it will forward the message to the neighbor which is closest to the destination in terms of the defined distance D.

To avoid messages getting stuck in a local minima in networks with a concave shape or networks with holes, DVCR uses the ordinate difference between the nodes and its neighbors. Let us consider the ordinate difference set $\Delta_{A_1A_2}$ with reference to anchor nodes A_1 and A_2. Therefore,

$$\Delta_{A_1A_2} = |F_{A_1A_2}(N_i, N_k)|; N_k \in K \tag{3.6}$$

where K is the total number of neighbors of node N_i. Let the maximum ordinate difference be $\alpha_{12} = \max(\Delta_{A_1A_2})$ and the minimum ordinate difference be $\beta_{12} = \min(\Delta_{A_1A_2})$. Therefore, the approximate ordinate difference between current node and destination is given as:

$$\alpha_{12}n + \beta_{12}m = |F_{A_1A_2}(N_i, N_d)| \tag{3.7}$$

Similarly using reference anchor nodes A_3 and A_4 we get,

$$\alpha_{34}n + \beta_{34}m = |F_{A_3A_4}(N_i, N_d)| \tag{3.8}$$

By solving Equations (3.7) and (3.8), we are able to find $n+m$ which gives us an estimate of minimum number of hops to the destination. Similar calculations are performed by all of the neighbors of current node i and the node having the minimum number of hops is selected for forwarding.

CHAPTER 4
THE MS-DVCR ROUTING PROTOCOL

4.1 General description

Consider a sensor network with a mobile sink in which we want to use DVCR as a routing algorithm. Before the sink starts to move, all the nodes know the shortest hop distance to each of the anchors and to the sink. Also, each node knows VC coordinate of its neighbors to compare the distance to destination of each of its neighbors with itself. So, routing is possible until the time in which the sink starts to move. When the sink moves, its virtual coordinates will change, and the messages routed to the old coordinates will not reach the sink. A simple solution would be to notify all the nodes about the sink's new coordinates. This solution, however is expensive in terms of the number of messages, and the corresponding energy consumption especially when the sink moves relatively fast.

An idea which is proposed in geographical routing to deal with this problem is Local Update-Based Routing Protocol(LURP) [37]. We take this idea and propose an algorithm in virtual coordinate domain. The idea is that when the sink starts to move, it will notify the nodes within a local area. If the sink exited the local area while moving, it will broadcast its coordinates to the entire network and form a new area centered to its current location. By this scenario, each node outside the area forwards the packets to somewhere in the area.

As soon as the packet enters the area, it will be forwarded to the sink because all nodes in the area have the updated coordinates of the sink.

4.2 MS-DVCR algorithm

In order to specify the nodes inside the local area, we need to have a definition which distinguishes them from the other nodes in the network. In VCs we do not have the geographical location information of the nodes. So, for defining the local area, we use hop distance to the center of the area. Let us consider that at the time of the creation of the local area L the sink is at the location $N_s^c = [n_{s1}^c \dots n_{sP}^c]$. Thus, the local area will be defined as all the nodes which are closer than L2 distance r to the initial location of the sink:

$$R = \left\{ n \subset N \;\middle|\; D_{N_n N_s} = \sqrt{\sum_{i=1}^{P} (n_{ni} - n_{si})^2} \leq r \right\} \tag{4.1}$$

Sink is at the center of this area when the area is being created. However, when time passes, it can be anywhere in the area depending on its movement strategy. Based on the definition of the local area, sink movement can be labeled as two different types:

1. Local move: in this type of movement, the sink stays inside the local area. So, the sink needs to update only the nodes inside the local area about its new location, and the local area will not change.

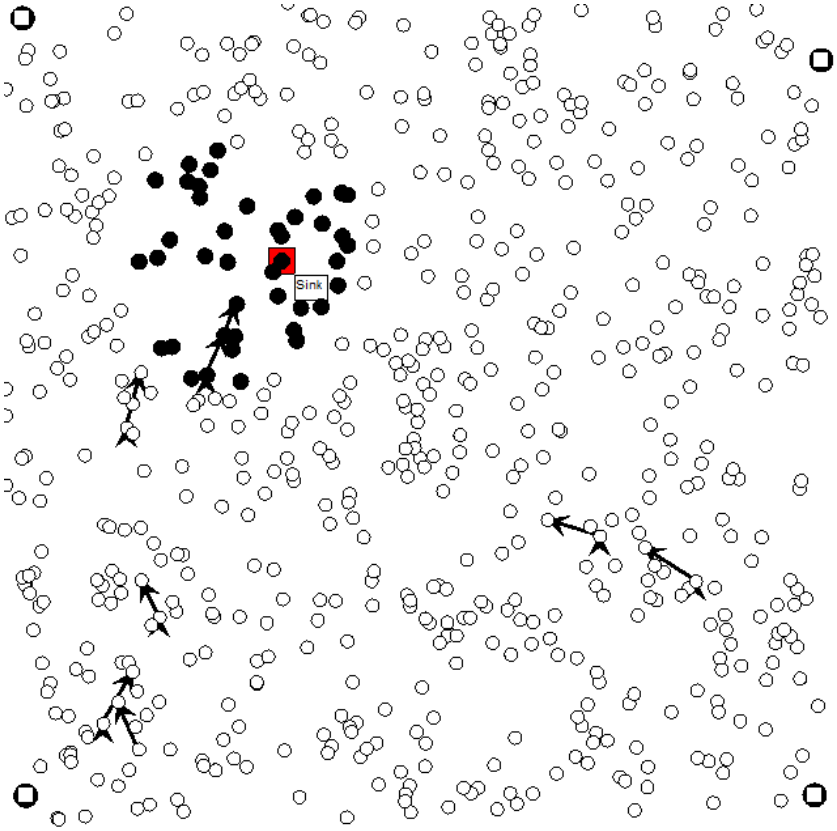

Figure 4.1: Operation of MS-DVCR between sink moves, black nodes: current local area, thick circles in the corner: anchor nodes

2. External move: this type of movement is occurred when the sink leaves the current local area R. As a result, the sink must (a) create a new local area R' and (b) notify the whole network about its new virtual coordinates.

You can see screenshots of the operation of a network in Figures 5.1, 5.2, 5.3.

Before proposing the MS-DVCR algorithm, we need to enumerate different types of messages used in it:

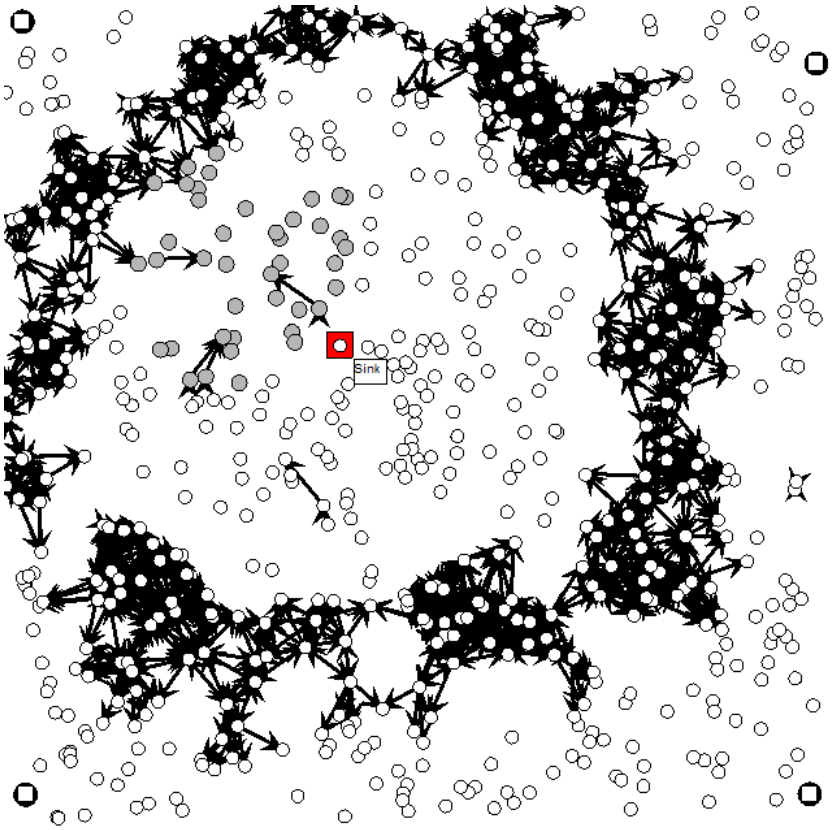

Figure 4.2: Network notification after an external move in MS-DVCR, black nodes: current local area, gray nodes: previous local area, thick circles in the corner: anchor nodes

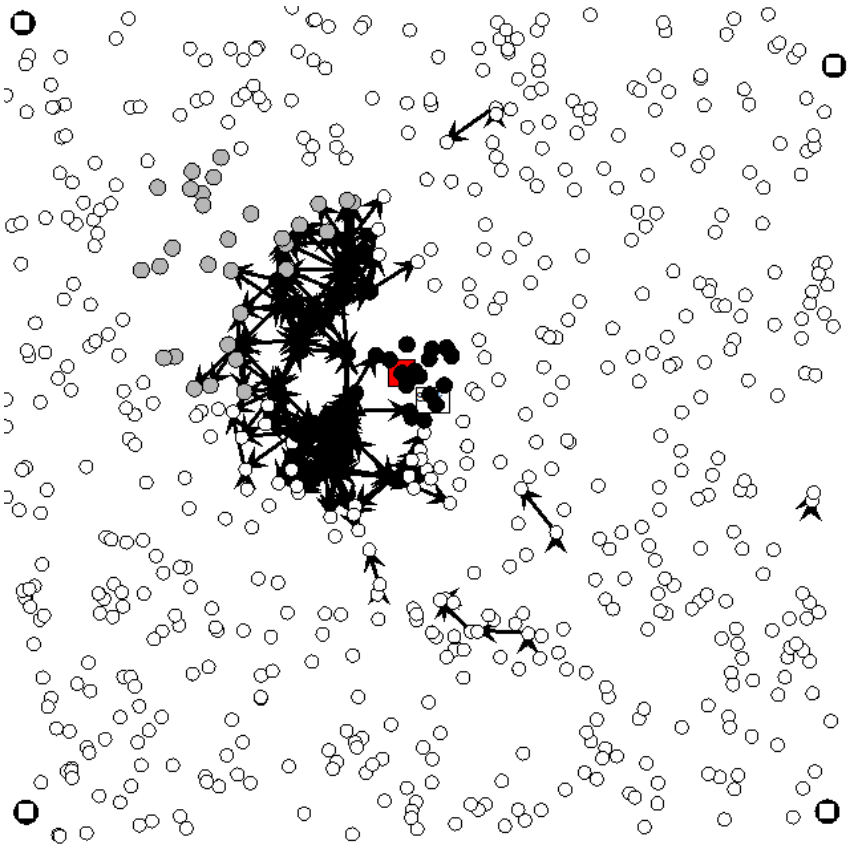

Figure 4.3: A local move after formation of a new local area in MS-DVCR, black nodes: current local area, gray nodes: previous local area, thick circles in the corner: anchor nodes

- LOCAL messages are sent by the sink and carry its location information to update the nodes inside the local area. These messages are broadcasted but they are limited to the nodes inside the local area.

- EXTERNAL messages are broadcasted to the entire network and carry the location information of the sink to update all the nodes in the network.

- SENSING messages are sent by the sensor nodes and carry sensed data to the sink. When they reach a node inside the local area, that node knows the current location of the sink and will forward the message to the sink.

The behavior of the sink in MS-DVCR is described in algorithm 1. First, the sink starts to move and replaces the new location with its current location. When it reaches to the new location, the sink checks whether it has exited the current local area or not. This is done by comparing its new location with the local area center. If it is further than the radius of the local area, it has exited the local area. In this case, the sink will broadcast and EXTERNAL message informing all the nodes in the network. If the result of the check is that the sink has not left the current local area yet, it broadcasts a LOCAL message to all the nodes inside the local area to update them. The sink also handles SENSING messages received from sensors throughout the network.

As described in Algorithm 2, a node handles three types of incoming messages and also does the sensing task. When the node receives a LOCAL message, it checks if the node is located inside the current local area. Based on the result of this check, it broadcasts the

Algorithm 1 Sink behavior in MS-DVCR

```
when move do
  new-location := current location of sink
  if (D_L2(new-location, local-area-center)) < r  then
    broadcast(msg(LOCAL, new-location))
  else
    local-area-center := new-location
    broadcast(msg(EXTERNAL, local-area-center))
  end if
end when
when receives(msg(SENSING, data)) do
  update local model with data
end when
```

message to the neighbor nodes. The node updates the next hop to the sink independent of its location when it receives a LOCAL message or an EXTERNAL message. When the node receives an EXTERNAL message, it broadcasts it without checking whether the node is located inside the local area or not. The node simply forwards the incoming SENSING messages using the next-hop field it is keeping and updating when receiving the new location of the sink. When it observes a new event, the node creates a new message and forwards it to the next hop to the sink.

Algorithm 2 Node behavior in MS-DVCR

```
when receives(message(LOCAL, new-sink-location)) do
  nexthop := closest neighbor to new-sink-location
  if (D_L2(local-area-center, nodelocation) < r) then
    broadcast(msg(LOCAL, local-area-center))
  end if
end when
when receives(message(EXTERNAL, local-area-center)) do
  nexthop := closest neighbor to local-area-center
  broadcast(msg(EXTERNAL, local-area-center))
end when
when receives(message(SENSING, data)) do
  send(msg(SENSING, data), nexthop)
end when
when sensor-captures(observation) do
  data = report-formation(observation)
  send(msg(SENSING, data), nexthop)
end when
```

CHAPTER 5
THE CU-DVCR ROUTING PROTOCOL

5.1 General description

In the previous chapter we saw that the main idea behind MS-DVCR is to limit the radius of broadcasting to the nodes inside a local area while the sink is inside the area. We will introduce CU-DVCR algorithm which takes one step further by limiting the broadcasting to the nodes on the boundary of local area. This idea comes from the fact that only nodes on the boundary of the local area are involved in correcting the assumption of incoming messages to the local area about the current location of the sink.

To have a better intuition, consider that most of the packets are being generated outside the local area. If these packets know the exact location of the sink as soon as they enter the local area, they would be able to find their way towards it. So, updating the nodes other that the ones on the boundary of the local area does not seem to be necessary. If the incoming messages to the local area carry the updated location of the sink while traveling towards it, they can also update the nodes inside the local area on their way. By this scenario, the nodes inside the local area will be informed gradually of the new location of the sink.

We will give a more accurate explanation of this algorithm in next part.

5.2 The CU-DVCR algorithm

Similar to MS-DVCR, we can define the local area R as all the nodes in which their L2 distance to the initial location of the sink is smaller than r:

$$R = \left\{ n \subset N \;\middle|\; D_{N_n N_s} = \sqrt{\sum_{i=1}^{P} (n_{ni} - n_{si})^2} \leq r \right\} \tag{5.1}$$

We should also distinguish the nodes on the boundary of the local area from other nodes in this area. For this purpose, we can define B as the set of nodes on the boundary of the local area in which their L2 distance to the initial location of the sink is between r and $r - c$ where c is the width of the boundary area.

$$B = \left\{ n \subset N \;\middle|\; r - c \leq D_{N_n N_s} = \sqrt{\sum_{i=1}^{P} (n_{ni} - n_{si})^2} \leq r \right\} \tag{5.2}$$

Given this definition, we can imagine two types of movements for the sink:

1. Local move: if the sink stays inside the local area, it needs to send a message to one of the nodes on the boundary area B and that message will be broadcasted to the rest of the nodes in B.

2. External move: when the sink leaves the current local area R, it needs to (a) create a new boundary area B' and (b) notify the whole network about its new virtual coordinates.

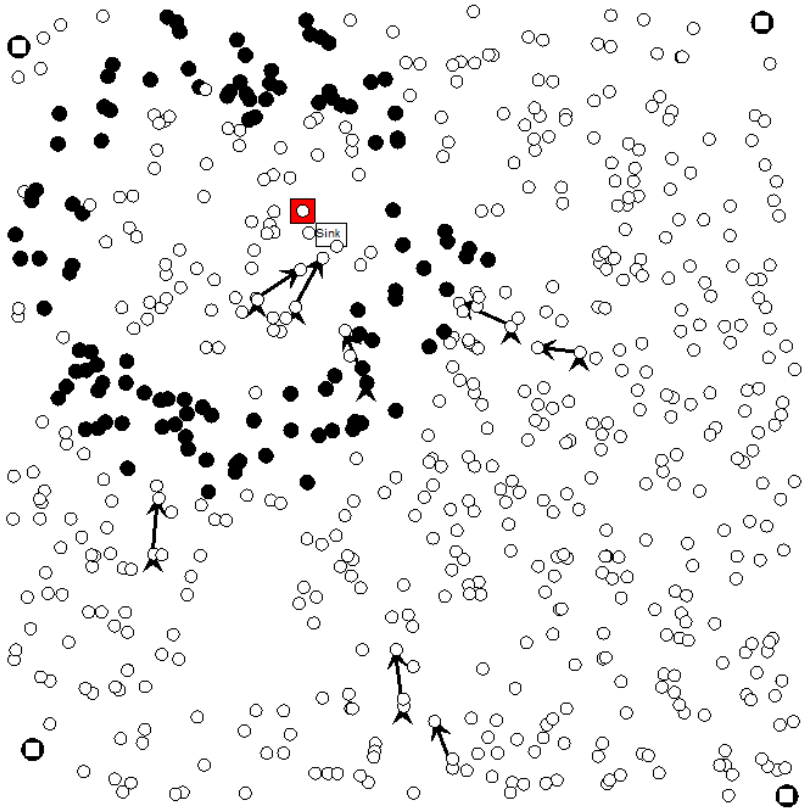

Figure 5.1: Operation of CU-DVCR between sink moves, black nodes: current boundary area, thick circles in the corner: anchor nodes

You can see three screenshots in Figures 5.1, 5.2, 5.3 to have a better understanding of the operation of CU-DVCR.

Similar to MS-DVCR, we can define three types of messages used in CU-DVCR:

- LOCAL messages are sent by the sink to one of its neighbors and will be forwarded to the boundary area B. The first node in B which gets these messages, broadcasts them to all the nodes inside B. They contain the current location information of the sink.

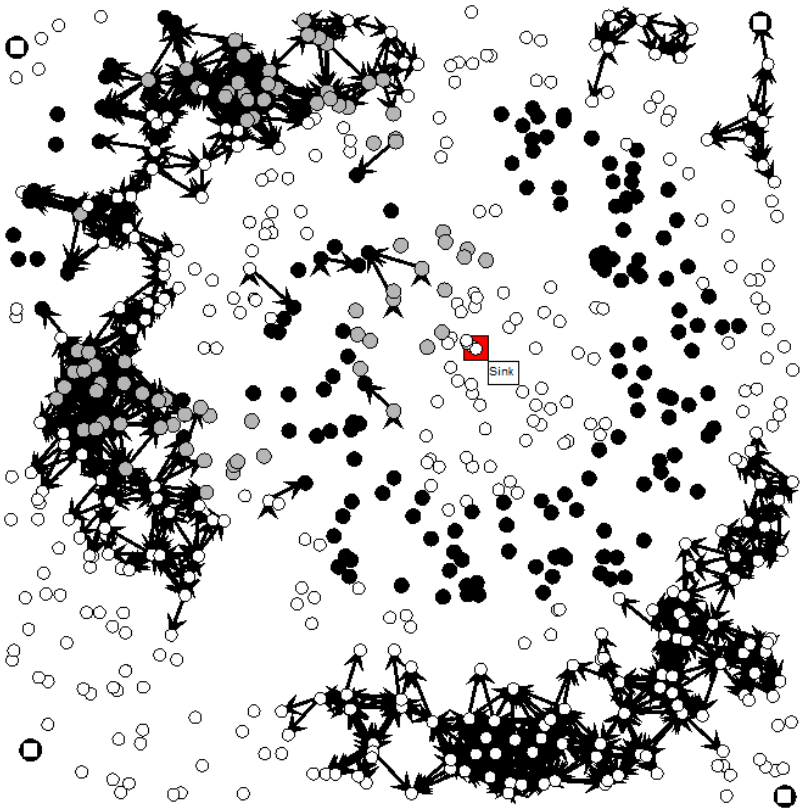

Figure 5.2: Network notification after external move in CU-DVCR, black nodes: current boundary area, gray nodes: previous boundary area, thick circles in the corner: anchor nodes

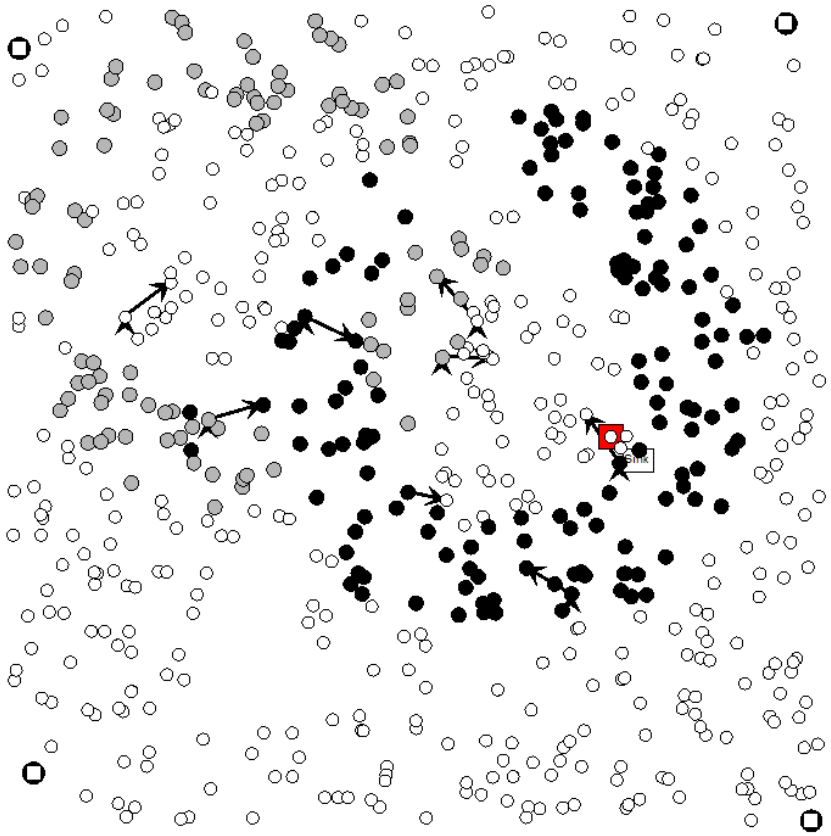

Figure 5.3: New and old local areas in CU-DVCR, black nodes: current boundary area, gray nodes: previous boundary area, thick circles in the corner: anchor nodes

- EXTERNAL messages are broadcasted to the entire network when the sink does an external move and carry its new location information.

- SENSING messages are sent by the sensor nodes and carry sensed data to the sink. They reach a node on the boundary area B, obtain the current location of the sink from that node, and update the other nodes inside the local area on their way to the sink.

Algorithm 3 describes the behavior of the sink in CU-DVCR. When the sink moves, it updates its new location based on the coordinates of its neighbors. Then it checks whether the new location is inside the current local area; i.e., it has done a Local or External move. In case of Local move, it sends a LOCAL message to one of its neighbors to be forwarded to the boundary area. When the sink does an External move and leaves the local area, it creates a new local area. It sets the center of the new local area as its location. Then, it broadcasts an EXTERNAL message containing this location information. The sink also handles SENSING messages based on the specification of the network.

Algorithm 4 describes the behavior of a node in CU-DVCR. When a node receives a LOCAL message containing the new location of the sink, it updates the next hop to the destination which is one of its neighbors. Then it checks whether it is located inside the boundary area or not. If yes, it broadcasts the message, otherwise, in the case of being inside the local area and outside of the boundary area, it will forward the message to the furthest node from the sink to finally reach a point on the boundary of the local area.

Algorithm 3 Sink behavior in CU-DVCR

```
when move do
  new-location := current location of sink
  if (D_L2(new-location, local-area-center)) < r  then
    send(msg(LOCAL, new-location), random-neighbor)
  else
    local-area-center := new-location
    broadcast(msg(EXTERNAL, local-area-center))
  end if
end when
when receives(msg(SENSING, data)) do
  update local model with data
end when
```

In case of receiving an EXTERNAL message, local-area-center and next-hop fields will be updated and the node will broadcast the message to the other nodes in the network. A sensor node forwards a SENSING message to the next hop to the sink. It also creates a message after sensing data and forwards it to the next hop.

Algorithm 4 Node behavior in CU-DVCR

```
when receives(message(LOCAL, new-sink-location)) do
  nexthop := closest neighbor to new-sink-location
  if (D_L2(local-area-center, nodelocation) < r) then
    if (D_L2(local-area-center, nodelocation) > r-c) then
      broadcast(msg(LOCAL, new-sink-location))
    else
      send(msg(LOCAL, new-sink-location), farthest-neighbor-from-sink)
    end if
  end if
end when
when receives(message(EXTERNAL, new-local-area-center)) do
  local-area-center := new-local-area-center
  nexthop := closest neighbor to local-area-center
  broadcast(msg(EXTERNAL, local-area-center))
end when
when receives(message(SENSING, data)) do
  send(msg(SENSING, data), nexthop)
end when
when sensor-captures(observation) do
  data = report-formation(observation)
  send(msg(SENSING, data), nexthop)
end when
```

CHAPTER 6
EXPERIMENTAL STUDY

In this chapter, we do some experiments to analyze different aspects of the algorithms we introduced. We compare MS-DVCR and CU-DVCR with a naive solution which is to update all the nodes in the network when the sink moves. This algorithm is called Update All-DVCR UA-DVCR) [31]. By the way, this is an expensive solution in terms of energy consumption considering the number of messages needed to update all the nodes in the network.

6.1 Performance analysis

Before jumping to the experiments, we want to analyze the energy consumption of each of the algorithms. The difference between the algorithms we compare in this chapter appears when the sink moves:

- Update All-DVCR (UA-DVCR) [31] - Update all the nodes in the network

- Mobile Sink-DVCR (MS-DVCR) [31] - Update the nodes in a local area around the sink

- Circular Update-DVCR (CU-DVCR) - Update the nodes on the boundary of a local area around the sink

To have a better understanding of the energy consumption, we need to divide it into three parts:

1. Updating the nodes inside the local area
2. Updating the nodes outside the local area
3. Forwarding the events to the sink

In first part, CU-DVCR is expected to consume less amount of energy compared to MS-DVCR and UA-DVCR since it is not using broadcasting to update all the nodes inside the local area. For the second part, considering the fact that in MS-DVCR and CU-DVCR updating of the nodes when the sink does an external move is limited to the nodes inside the local area, we expect less energy consumption for them compared to UA-DVCR.

In CU-DVCR, the events created at the nodes inside the local area may traverse a lengthier path due to lack of awareness about the exact location of the sink. These packets go to the previous location of the sink and after realizing that the sink has moved, they will go to the new location. So, theoretically, CU-DVCR should perform worse than other algorithms in the third part. However, We expect this to have a small effect on the overall energy consumption due to two reasons: (a) since the nodes which are not updated when the sink does a local move constitute a small portion of all the nodes in the network, the probability of an event to be created in them is small (b) these nodes receives the updated location of

the sink as soon as an incoming packet from outside of local area passes the boundary of the local area and comes to them.

In brief, we expect less energy consumption for CU-DVCR compared to MS-DVCR. In addition, the energy consumption of UA-DVCR should be significantly higher than the other algorithms because of the important effect of the second part on overall energy consumption. This point should be mentioned that if the sink moves very slowly, the effect of third part may become sensible.

6.2 Experimental setup

In our simulations, we have used a sensor network with square area of $L \times L$ meters. The simulations are implemented in the Java-based extensible simulator YAES [3]. Sensor nodes are randomly and uniformly distributed with the average density of 0.005 nodes/m^2. The transmission range of each sensor node is 30 meters. We use virtual coordinates with 4 sensor nodes at each corner of the square area as the anchor points. When simulation time passes, events are generated randomly at all sensor nodes. The generated events will be forwarded to the sink using one of the routing algorithms.

For the sink mobility model, we use Random Waypoint [22] which helps in distributing the energy consumption throughout different areas of the network to increase the network lifetime. First, sink chooses one of the sensor nodes as its destination. To reach that

Table 6.1: Experimental parameters

Parameter	Values
General	
Sensor network size $L \times L$	400...1000 m
Node deployment	random uniform
Deployment density	0.005 nodes / m^2
Number of sensor nodes	800-5000
Transmission range	30 m
Sink movement	random waypoint, no stops
Sink speed	1-5 m/s
Experiment length	4000 messages
Protocols	
Routing protocol	UA-DVCR, MS-DVCR, CU-DVCR
Coordinates	directed virtual
Anchors	4, extreme corners
local area radius	$r \in \{5, \ldots, 12\}$
boundary area width	$c = 4$

destination, in every step it moves from one of the sensor nodes to one of its neighbors. Once it reaches the destination, it will choose another random destination and moves toward it. Table 6.1 lists the parameters of the experimental setup. In the following sections, we demonstrate the results of the experiments and explain them.

6.3 Energy consumption and average path length function of the size of the sensor network

Our goal in this set of experiments was to compare the algorithms in energy consumption and the average path length messages traversed by varying the width of the sensor network. The sensor network width was varied between 400 and 1000 meters with the constant de-

ployment density of 0.005 nodes/m^2. So, the generated networks contained between 800 and 5000 sensor nodes. The speed of the sink was set to 4m/s. Rappaport communication model [33] was used to calculate the energy consumption. The total energy consumption is the sum of energy consumption of all the nodes in the network. The average path length was calculated as the average number of hops successfully routed messages traversed to reach the sink. The experiments had been averaged over 20 different runs with different random seeds for the deployment of the nodes and the movement of the sink.

The radius of the local area is being calculated using the following formula. This makes the size of the local area relative to the size of the network.

$$r = L \div 100 + 4 \tag{6.1}$$

Figure 6.1 compares the overall energy consumption of the protocols during the simulation. As expected, UA-DVCR is consuming significantly higher amount of energy compared to other algorithms. In larger networks this difference in energy consumption is more visible. MS-DVCR shows a significant reduction in energy consumption compared to UA-DVCR. Finally, CU-DVCR shows best results among the algorithms. It is also less sensitive to increase in the size of the network. This shows the effect of broadcasting on the energy consumption.

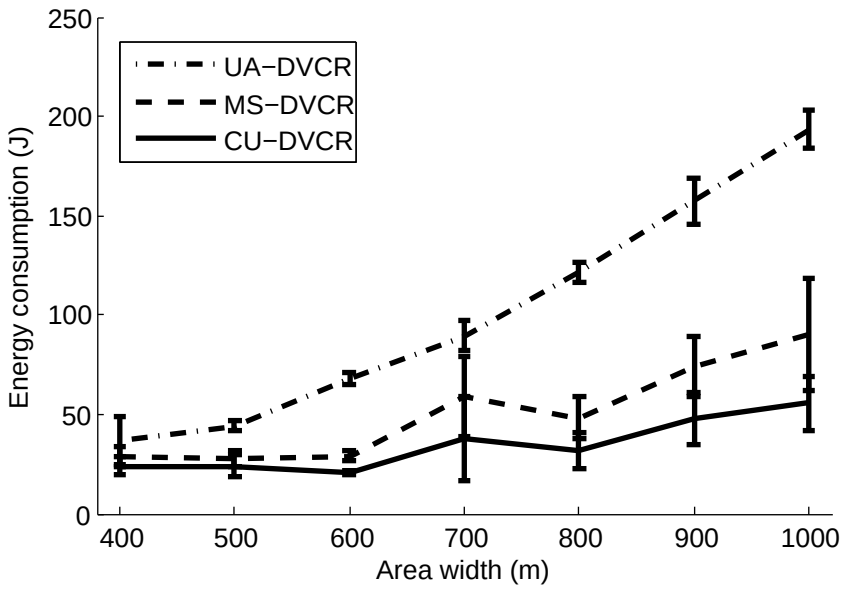

Figure 6.1: Overall energy consumption for UA-DVCR, MS-DVCR, and CU-DVCR function of area size.

In CU-DVCR we have limited the broadcasting to the boundary of the local area. So, we see a huge difference between the algorithms as the size of the network increases.

We consider path length to be the number of hops a message traverses to reach the sink. Using this definition, we can average path length traversed by all the packets and reach Figure 6.2. The differences between the path lengths in these protocols are barely visible and are essentially masked by the randomness of the generated network. As expected, for all the algorithms, the average path length increases with increase in size of the area.

We can conclude this part with the finding that CU-DVCR is outperforming other algorithms in energy consumption especially in larger networks. This achievement is reached without barely visible difference in average path length traversed by the messages.

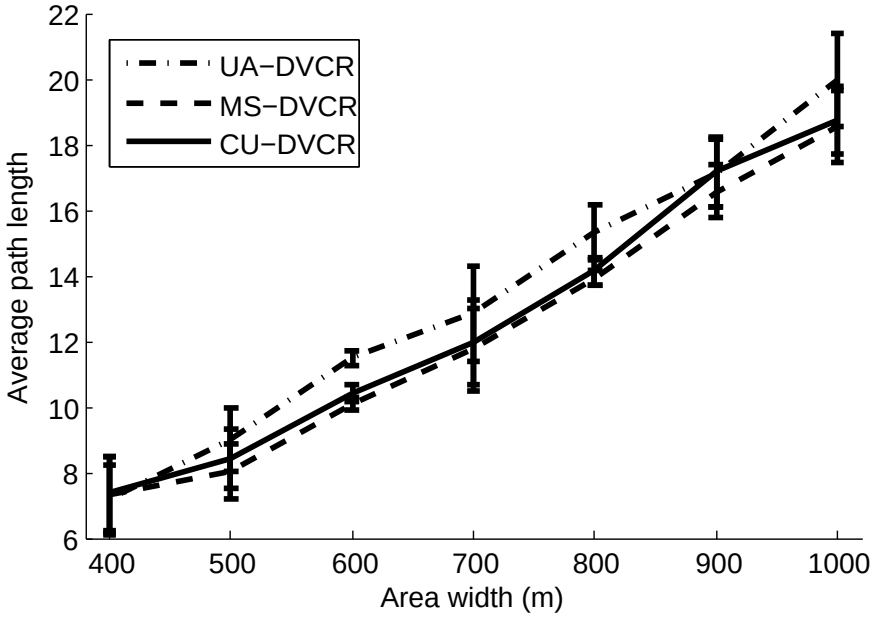

Figure 6.2: Average path length for UA-DVCR, MS-DVCR, and CU-DVCR function of area size.

6.4 Energy consumption and routability function of the sink speed

In this section, our goal is to see the effect of changing the sink speed on energy consumption and routability. For this purpose, we set up a $1000m \times 1000m$ network. We set the radius of the local area to 10 hops. in each run 5000 messages were generated and sent to the sink. The results are averaged over 15 runs with different random seeds for the deployment of the nodes and the movement of the sink. Figure 6.3 shows energy consumption as a function of the sink speed. CU-DVCR outperforms MS-DVCR especially when the sink moves faster. This was expected because more updates occur inside the local area as the

sink moves faster. As a result, more packets will be sent to notify the nodes in the local area in MS-DVCR and more energy will be consumed.

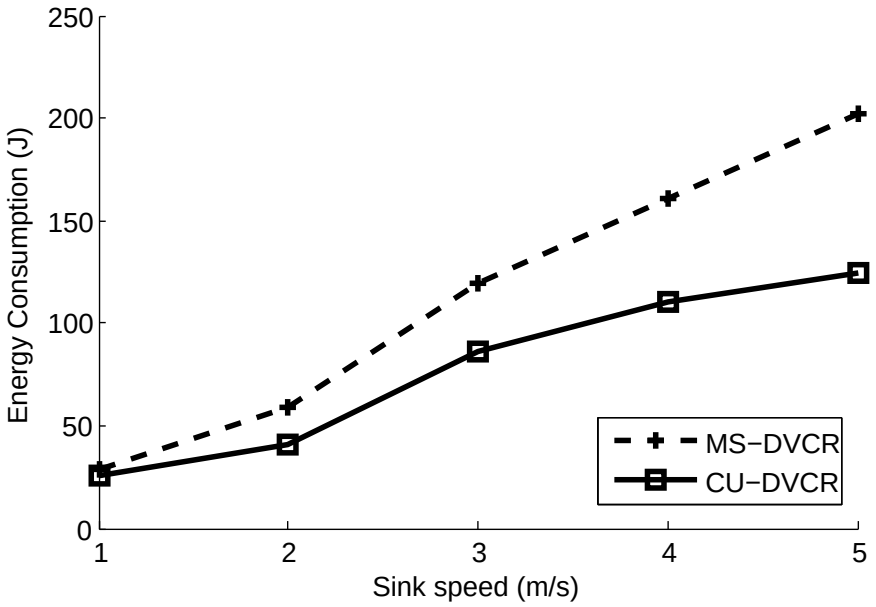

Figure 6.3: Energy consumption for MS-DVCR and CU-DVCR function of the sink speed.

Now, we want to compare the MS-DVCR and CU-DVCR in delivery ratio of the packets. We also want to see if increasing the sink speed affects delivery ratio of the packets or not. Figure 6.4 shows the number of successfully delivered messages function of the sink speed. More than 99.8% of packts were successfully delivered to the sink and the difference between the two algorithms is negligible. Thus, CU-DVCR has successfully decreased the energy consumption without any noticeable effect on the packet delivery ratio compared to MS-DVCR. We also see that delivery ratio is not changing as the sink speed increases.

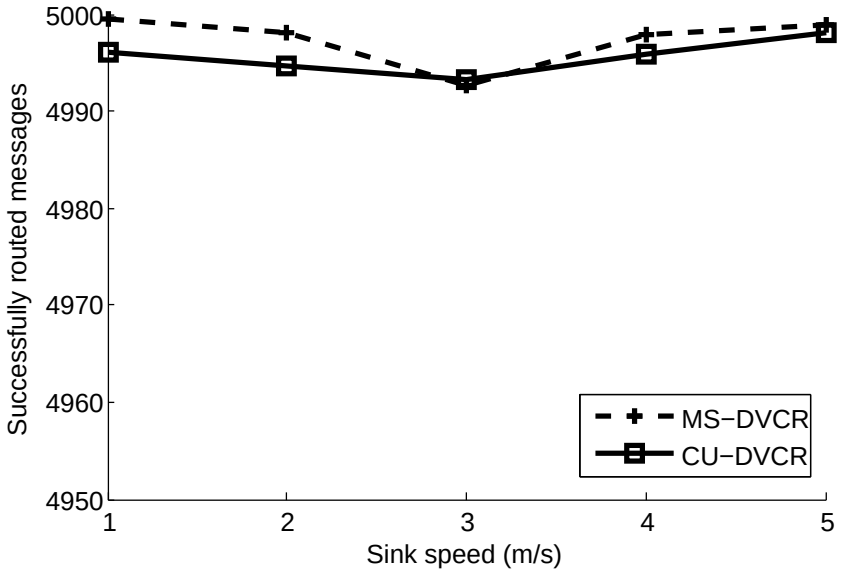

Figure 6.4: Number of successfully routed messages for MS-DVCR and CU-DVCR function of the sink speed. Total number of transmitted messages were 5000.

6.5 Energy consumption function of the size of the local area

In this section, we want to investigate the effect of varying the radius of local area on energy consumption. We set the side length of the network to 600m and the sink speed to 4m/s. Total of 4000 messages had been generated in each of these experiments and the results were averaged over 10 runs with different random seeds for the deployment of the nodes and the movement of the sink. Figure 6.5 shows the results of these runs. In general, choosing a small value for the radius of local area causes an increase in energy consumption. This is because the sink leaves a smaller local area more frequently than a larger one. On the other hand, making local area larger requires more number of packets to update local area.

This results in an optimal radius for energy consumption in the middle of the graph. For the network with these parameters, the optimal hop value for both algorithms is 10 hops.

CU-DVCR is more efficient since it needs less number of packets to update a boundary area compared to MS-DVCR which updates the entire local area.

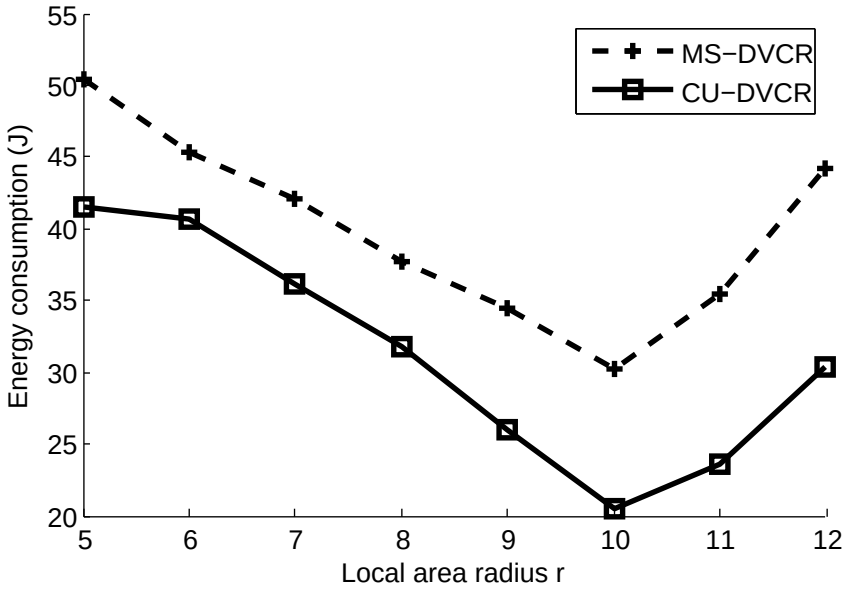

Figure 6.5: Energy consumption function of the radius of local area

CHAPTER 7
CONCLUSIONS

In this thesis, we introduced MS-DVCR and CU-DVCR, protocols for routing messages towards a mobile sink using virtual coordinates system. The main idea behind MS-DVCR is to limit the location updates sent by the mobile sink to a local area around the sink. The CU-DVCR algorithm takes a step further and limits the broadcasting to a circular area on the boundary of the local area. Experimental studies show that these algorithms consume significantly lower amount of energy compared to the naive solution of updating the entire network when the sink moves. In addition, CU-DVCR conserves more energy compared to MS-DVCR without causing the path length traversed by the messages to become longer. Another experiment showed that when the sink moves faster, CU-DVCR can save more energy without sacrificing routability. Finally, we found an optimal radius of local area for a network in which the energy consumption is minimized.

LIST OF REFERENCES

[1] Nadeem Ahmed, Salil S Kanhere, and Sanjay Jha. The holes problem in wireless sensor networks: a survey. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(2):4–18, 2005.

[2] Kemal Akkaya and Mohamed Younis. A survey on routing protocols for wireless sensor networks. *Ad hoc networks*, 3(3):325–349, 2005.

[3] L. Bölöni and D. Turgut. YAES - a modular simulator for mobile networks. In *Proc. of the 8th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM-05)*, pages 169–173, October 2005.

[4] L. Bölöni, D. Turgut, S. Basagni, and C. Petrioli. Scheduling data transmissions of underwater sensor nodes for maximizing value of information. In *IEEE Global Communications Conference (GLOBECOM-13)*, pages 438–444, December 2013.

[5] David Braginsky and Deborah Estrin. Rumor routing algorthim for sensor networks. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 22–31. ACM, 2002.

[6] Nirupama Bulusu, John Heidemann, and Deborah Estrin. Gps-less low-cost outdoor localization for very small devices. *Personal Communications, IEEE*, 7(5):28–34, 2000.

[7] Qing Cao and Tarek Abdelzaher. Scalable logical coordinates framework for routing in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 2(4):557–593, 2006.

[8] Antonio Caruso, Stefano Chessa, Swades De, and Alessandro Urpi. GPS free coordinate assignment and routing in wireless sensor networks. In *Proc. of 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM-05)*, volume 1, pages 150–160, March 2005.

[9] Arnab Chakrabarti, Ashutosh Sabharwal, and Behnaam Aazhang. Using predictable observer mobility for power efficient design of sensor networks. In *Information Processing in Sensor Networks*, pages 129–145. Springer, 2003.

[10] Min Chen, Xiaodan Wang, Victor Leung, and Yong Yuan. Virtual coordinates based routing in wireless sensor networks. *Sensor Letters*, 4(3):325–330, 2006.

[11] Daniel A Coffin, Daniel J Van Hook, Stephen M McGarry, and Stephen R Kolek. Declarative ad-hoc sensor networking. In *International Symposium on Optical Science and Technology*, pages 109–120. International Society for Optics and Photonics, 2000.

[12] Dulanjalie C Dhanapala and Anura P Jayasumana. Csr: Convex subspace routing protocol for wireless sensor networks. In *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*, pages 101–108. IEEE, 2009.

[13] Dulanjalie C Dhanapala and Anura P Jayasumana. Dimension reduction of virtual coordinate systems in wireless sensor networks. In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pages 1–6. IEEE, 2010.

[14] Dulanjalie C Dhanapala and Anura P Jayasumana. Topology preserving maps from virtual coordinates for wireless sensor networks. In *Proc. of the 35th IEEE Conference on Local Computer Networks (LCN-10)*, pages 136–143, October 2010.

[15] Dulanjalie C. Dhanapala and Anura P. Jayasumana. Anchor selection and serving maps in WSNs-A directional virtual coordinate based approach. In *Proc. of 36th IEEE Conference on Local Computer Networks (LCN-11)*, pages 571–579, October 2011.

[16] Dulanjalie C. Dhanapala and Anura P. Jayasumana. Directional virtual coordinate systems for wireless sensor networks. In *Proc. of IEEE International Conference on Communications (ICC-11)*, pages 1–6, 2011.

[17] Dulanjalie C Dhanapala and Anura P Jayasumana. Clueless nodes to network-cognizant smart nodes: Achieving network awareness in wireless sensor networks. In *Proc. of IEEE Consumer Communications and Networking Conference (CCNC-12)*, pages 174–179, January 2012.

[18] Shashidhar Rao Gandham, Milind Dawande, Ravi Prakash, and Subbarayan Venkatesan. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. In *IEEE Global Communications Conference (GLOBECOM-03)*, volume 1, pages 377–381, December 2003.

[19] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proc. of the 33rd Annual Hawaii International Conference on System Sciences, HICSS '00*, pages 1–10, 2000.

[20] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of*

the 6th annual international conference on Mobile computing and networking, pages 56–67. ACM, 2000.

[21] Yi Jiang, Dulanjalie C Dhanapala, and Anura P Jayasumana. Tracking and prediction of mobility without physical distance measurements in sensor networks. In *Proc. of IEEE International Conference on Communications, IEEE ICC '13*, 2013.

[22] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile computing*, pages 153–181. Springer, 1996.

[23] Brad Karp and Hsiang-Tsung Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proc. of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom-00)*, pages 243–254, August 2000.

[24] Anthony LaMarca, Waylon Brunette, David Koizumi, Matthew Lease, Stefan B Sigurdsson, Kevin Sikorski, Dieter Fox, and Gaetano Borriello. Making sensor networks practical with robots. In *Pervasive Computing*, pages 152–166. Springer, 2002.

[25] Xu Li, Jiulin Yang, Amiya Nayak, and Ivan Stojmenovic. Localized geographic routing to a mobile sink with guaranteed delivery in sensor networks. *IEEE Journal on Selected Areas in Communications*, 30(9):1719–1729, 2012.

[26] Haiyun Luo, Fan Ye, Jerry Cheng, Songwu Lu, and Lixia Zhang. Ttdd: Two-tier data dissemination in large-scale wireless sensor networks. *Wireless Networks*, 11(1-2):161–175, 2005.

[27] Jun Luo and J-P Hubaux. Joint mobility and routing for lifetime elongation in wireless sensor networks. In *Proc. of 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM-05)*, volume 3, pages 1735–1746, March 2005.

[28] Jun Luo, Jacques Panchard, Michał Piórkowski, Matthias Grossglauser, and Jean-Pierre Hubaux. Mobiroute: Routing towards a mobile sink for improving lifetime in sensor networks. In *Distributed Computing in Sensor Systems*, pages 480–497. Springer, 2006.

[29] Mirela Marta and Mihaela Cardei. Improved sensor network lifetime with multiple mobile sinks. *Pervasive and Mobile Computing*, 5(5):542–555, 2009.

[30] Ioannis Papadimitriou and Leonidas Georgiadis. Maximum lifetime routing to mobile sink in wireless sensor networks. In *Proc. SoftCOM*, 2005.

[31] R. Rahmatizadeh, S.A. Khan, A.P. Jayasumana, D. Turgut, and L. Bölöni. Routing towards a mobile sink using virtual coordinates in a wireless sensor network. In *to be presented at Proc. IEEE Int'l Conference on Communications (ICC 2014)*, 2014.

[32] Ananth Rao, Sylvia Ratnasamy, Christos Papadimitriou, Scott Shenker, and Ion Stoica. Geographic routing without location information. In *Proc. of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom-03)*, pages 96–108, September 2003.

[33] Theodore S Rappaport. *Wireless communications: principles and practice*. Publishing House of Electronics Industry, 2004.

[34] Andreas Savvides, Chih-Chieh Han, and Mani B Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 166–179. ACM, 2001.

[35] Rahul C Shah, Sumit Roy, Sushant Jain, and Waylon Brunette. Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks. *Ad Hoc Networks*, 1(2):215–233, 2003.

[36] Lang Tong, Qing Zhao, and Srihari Adireddy. Sensor networks with mobile agents. In *Proc. of IEEE Military Communications Conference (MILCOM-03)*, volume 1, pages 688–693, 2003.

[37] Guojun Wang, Tian Wang, Weijia Jia, Minyi Guo, Hsiao-Hwa Chen, and Mohsen Guizani. Local update-based routing protocol in wireless sensor networks with mobile sinks. In *Proc. of IEEE International Conference on Communications (ICC-07)*, pages 3094–3099, June 2007.

[38] Guojun Wang, Tian Wang, Weijia Jia, Minyi Guo, and Jie Li. Adaptive location updates for mobile sinks in wireless sensor networks. *The Journal of Supercomputing*, 47(2):127–145, 2009.

[39] Wei Wang, Vikram Srinivasan, and Kee-Chaing Chua. Using mobile relays to prolong the lifetime of wireless sensor networks. In *Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 270–283. ACM, 2005.

[40] Z Maria Wang, Stefano Basagni, Emanuel Melachrinoudis, and Chiara Petrioli. Exploiting sink mobility for maximizing sensor networks lifetime. In *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on*, pages 287a–287a. IEEE, 2005.