Delay-constrained energy-efficient routing in heterogeneous wireless sensor networks

Yinying Yang and Mihaela Cardei*

Department of Computer Science and Engineering,
Florida Atlantic University,
Boca Raton, FL 33431, USA
Email: yyang4@fau.edu
Email: mihaela@cse.fau.edu
*Corresponding author

Abstract: In this paper, we propose a routing strategy for heterogeneous wireless sensor networks, which supports both periodic and event-based reporting applications. The energy-efficient network model in this paper consists of static sensor nodes, mobile and static supernodes. We propose an energy-efficient routing protocol that can satisfy the delay requirements of different types of data messages. Each source-sensor, with data to be transmitted to the sink, chooses the best relay supernode from the candidates in its routing table such that it meets the delay requirements. The message is transmitted to the selected relay supernode using sensor-to-sensor communication, and from there to the sink using supernode-to-supernode communication. Simulation results show that our routing protocol is energy-efficient and effectively satisfies different delay requirements.

Keywords: heterogeneous wireless sensor networks; energy efficiency; delay-constrained data reporting; mobile supernodes; routing protocol.

Reference to this paper should be made as follows: Yang, Y. and Cardei, M. (2010) 'Delay-constrained energy-efficient routing in heterogeneous wireless sensor networks', *Int. J. Sensor Networks*, Vol. 7, No. 4, pp.236–247.

Biographical notes: Yinying Yang is a PhD student at the Department of Computer Science and Engineering at Florida Atlantic University. She got her master's degree from Southeast University, China. Since 2006, she has been researching in wireless sensor networks.

Mihaela Cardei is Associate Professor in the Department of Computer Science and Engineering at Florida Atlantic University, and Director of the NSF-funded Wireless and Sensor Network Laboratory. She received her PhD and MS in Computer Science from the University of Minnesota, Twin Cities, in 2003 and 1999 respectively. Her research interests include wireless networking, wireless sensor networks, network protocol and algorithm design, and resource management in computer networks. She is a recipient of the 2007 Researcher of the Year Award at Florida Atlantic University. She is a member of IEEE and ACM.

1 Introduction

Heterogeneous wireless sensor networks consist of a large number of resource-constrained sensor nodes used for data measurements (Akyildiz et al., 2002) and fewer resource-rich wireless devices (or supernodes), which can be used for complex computations, decision making and data relaying (Cardei et al., 2008). Supernodes can move via wheels (Dantu et al., 2005) or they can be attached to transporters, such as robots (Dantu et al., 2005) and vehicles (Lee et al., 2006), to act as mobile relay nodes.

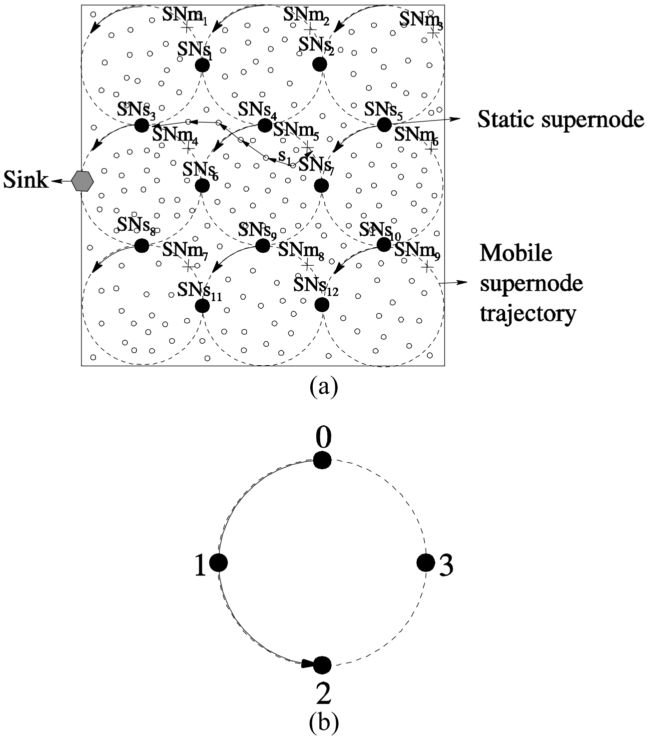
The use of supernodes is motivated by the observation that in a Wireless Sensor Network (WSN) consisting of static sensors and a static sink, the sensors closer to the sink consume more energy since they forward messages on behalf of other sensors located farther from the sink. They will die first, causing network partition. As a result, messages cannot be delivered to the sink. One method to address this issue is to use more powerful supernodes as relay nodes, thus the energy spent by sensors on data forwarding is saved and network lifetime is prolonged.

Shah et al. (2003) considered one mobile relay node that communicates with the sensors within the communication range. Energy is saved by using a single-hop communication instead of the expensive multi-hop communication. However, in this case, the latency of data transfer may be high since sensors have to wait until the mobile node comes close enough. But there are applications where data delivery is time critical. In this paper, we propose an energy-efficient design that uses mobile supernodes and supports delay constrained message delivery. We design a routing scheme, which combines multi-hop sensor communication with supernode communication, and trades-off message delay with sensor energy consumption.

Our solution supports applications with both periodic and event-based data reporting. For example, consider a fire-prevention application using a WSN that measures various parameters of the environment. When no critical values are measured, only periodic data reporting is used. If smoke is detected, then an event-based reporting is triggered. If a starting fire is detected, then a more critical event is issued. In designing our model, we consider that periodic data reporting is delay-tolerant, thus an energy-efficient mechanism can be used for data delivery. On the other hand, event-based data reporting is delay-sensitive, and the required lower delay is achieved at the expense of consuming more energy.

We divide the square monitored area into small grids (see Figure 1a). Each mobile supernode takes care of one grid. The mobile supernodes move along circular trajectories. Static supernodes act as mailboxes. Mobile supernodes drop messages off in mailboxes. A message stored in a mailbox waits to be picked up by the corresponding mobile supernode and carried closer to the sink.

Figure 1 Trajectories of the mobile supernodes and positions of the static supernodes. (a) Hierarchical structure of the network. (b) Movement of a mobile supernode and possible positions for the static supernodes on the circular trajectory



(a)

(b)

When a sensor has a data message to transmit, it decides the delivery method based on the message delay requirement. For a larger delay requirement, a sensor delivers the message using supernode-to-supernode communication. The message will be sent to a chosen relay supernode through a sensor-to-sensor multi-hop path, which includes only sensors as relay nodes. The supernode takes care of delivering the message to the sink without any sensor relaying, using only supernode-to-supernode communication. If the delay requirement cannot be met this way, the source sensor will choose a pure sensor-to-sensor path directly from itself to the sink. Note that using supernodes to relay delay-tolerant messages is more energy efficient and as a result, the network lifetime is prolonged.

2 Related works

There are recent research works that exploit node mobility to prolong the network lifetime (Shah et al., 2003; Luo et al., 2005; Wang et al., 2005a; Wang et al., 2005b; Soma et al., 2006; Ngai et al., 2007). Some of them are limited to certain applications that do not require a stringent latency requirement (e.g. Shah et al., 2003; Soma et al., 2006), while others are concerned with designing routing protocols that minimise the delivery delay (e.g. Ngai et al., 2007).

In work of Ngai et al. (2007), mobile actuators are used to collect data and they consider delay requirements when deciding their moving path. Sensors are static and they locally store the sensed data, which are uploaded when an actuator approaches. Sensors are assigned different weights and an actuator visits the sensors with higher weights more frequently in order to minimise the inter-arrival time. First, a priori route is formed by constructing a travelling salesman path, which contains all locations to be visited. The actuator determines whether to visit the next node on the route by generating a random number. If the random number is smaller than the weight of the node, the actuator visits the node. Otherwise, the actuator skips it and determines whether to visit the following node using the same mechanism.

In work of Luo et al. (2005), only one mobile sink is adopted to improve network lifetime. They first fix the routing strategy to the shortest path routing and search for the optimal mobility pattern of the base station. Through mathematical analysis, the authors argue that the best mobility strategy is to follow the periphery of the network. Then based on the optimal mobility pattern, they propose a routing strategy that combines round routing and the shortest path routing, resulting in better performance.

In work of Wu et al. (2007), the monitored area is divided into grids. A store-carry-forward mechanism is studied. The paper is concerned with designing a hierarchical trajectory and rendezvous points for ferries (relay nodes). Each trajectory is a square loop with four rendezvous points. The trajectory of a ferry in the dense mode is regarded as the trajectory of a public bus that is fixed, and the trajectory of a ferry in the sparse mode is regarded as the trajectory of a car-pool taxi that is determined on demand by customers.

In this paper, the network supports different applications that have different delay requirements. We focus on the trade-off between energy efficiency and delay requirements. The heterogeneous network model introduced in the paper effectively prolongs the network lifetime. The routing strategy proposed for this model delivers messages according to their delay requirement, in an energy-efficient manner.

3 Network model and problem definition

3.1 Network model

We consider a dense network deployed in an $R \times R$ square area, which consists of N static sensor nodes, N_m mobile supernodes, N_s static supernodes and a static sink. The supernodes and sensors are synchronised and are aware of their positions through GPS (Mcneff et al., 2002) or other localisation techniques (He et al., 2003; Huet al., 2004).

Figure 1a illustrates the network model. The sink is located in the middle of the left edge of the monitored area. The monitored area is divided into grids, with one mobile supernode responsible for each grid.

Similar as Liu et al. (2007) and Luo et al. (2005), we consider pre-defined circular repetitive trajectories for mobile supernodes, represented by dashed circles in Figure 1a. This is a realistic scenario since the motion of many objects has repetitive patterns and their positions at particular times can be roughly estimated. For example, consider vehicle-assisted networks (Chen et al., 2001; Zhao et al., 2006; Wu et al., 2007), where the monitored area is traversed by a public transportation system. Buses can carry the supernodes, therefore the speed and trajectory can be scheduled.

As mentioned in Section 2, there are related works focusing on similar mobile networks. In work of Luo et al. (2005), there is only one mobile sink that moves along a circular trajectory. In work of Liu et al. (2007), mobile nodes move along circular trajectories with different periods. In work of Wu et al. (2007), monitored area is divided into grids and different relay nodes cover different grid areas. Our goal in this paper is to design a routing strategy considering both the delay constraint and energy efficiency. To solve the problem for a general case and to simplify the analysis, we use the following assumptions for the mobility model.

We assume that there is at least one mobile supernode in the network. The number of grids and mobile supernodes is adjustable. When there is only one mobile supernode, this reduces to the case as in Luo et al. (2005), where the trajectory is considered to be optimum for network lifetime elongation. When the network is equipped with multiple mobile supernodes, more sensors can find shorter sensor-to-sensor paths to the sink through supernodes. Simulation results in Figure 7a compare the impact of different numbers of mobile supernodes.

We assume that all mobile supernodes have the same schedule, which means that they move with the same velocity in the same direction (counter clockwise) starting from the position 0, as shown in Figure 1b. We also assume that the mobile supernodes begin to move at the same time so that they are in the same relative location of the circular trajectory. Our algorithm can be extended to other cases, such as when mobile supernodes use different time periods to traverse a circle or when the relative locations of mobile supernodes are different.

The time period to traverse a circle is denoted by T. Mobile supernodes in Figure 1a are labelled SNm_1, SNm_2, …, SNm_9. The crosses in Figure 1a show their positions at a particular time instance. The dark dots represent the static supernodes. They are located at the intersection of two circular trajectories and they serve as mailboxes. Figure 1b shows the four possible positions 0,1, 2, 3, for static supernodes on a circular trajectory. Static supernodes in Figure 1a are labelled SNs_1, SNs_2, …, SNs_{12}.

Mobile supernodes are used to carry data from one static supernode (mailbox) to another. Data packets are dropped to mailboxes and stored until they are picked up by other mobile supernodes. Using supernode-to-supernode communication, messages are delivered to the sink. We assume that after a data message reaches a supernode, only supernode-to-supernode communication is used to deliver the message to the sink, which means no additional sensor relaying is involved. To reach the first supernode on the path, a sensor-to-sensor communication is used. To summarise, a source-sensor first tries to use supernodes for data delivery, decision based on the message delay requirement. If using supernodes for data delivery does not meet the delay requirement, a sensor-to-sensor delivery from the source-sensor to the sink is employed.

The supernodes and sink energy consumption is not taken into account since in the general case they are resource-rich devices. For example, they could be carried by public buses and therefore no additional energy is consumed. Static supernodes cannot communicate with each other directly, assuming that the distance between them is longer than the communication range.

Our network model is more energy-efficient compared to a static WSN. Using mobile supernodes, our routing protocol reduces the overall number of sensors involved in data relaying, and this energy saving has a high impact on prolonging network lifetime.

3.2 Problem definition

Data messages have different delay requirements. A source-sensor labels each packet with a specific delay requirement D, depending on the criticality of the message. Thus the message has to be delivered to the sink within time D. Our objective is to design a routing protocol that satisfies the message delay requirements while maximising the network lifetime.

The problem that we address in this paper is the *Delay-constrained Energy-efficient Routing Problem (DERP): Given the trajectories of the mobile supernodes and the positions of the sink and the static supernodes, the goal is to design a routing mechanism such that the sensed data are delivered to the sink in the specified delay bound D while maximising the network lifetime.*

In Figure 1a, sensor node s_1 has multiple routing choices, two of them are shown. The first choice is to buffer the message and send it to the mobile supernode SNm_5 when it moves within direct communication range. This is the most energy-efficient way since no other sensors are involved in forwarding. Another choice is to deliver the message to the static supernode SNs_3 using a sensor-to-sensor path, and from here to the sink via the mobile supernode SNm_4. This choice consumes more energy compared to the first approach since more sensors are involved in data forwarding. s_1 has other routing choices not shown in Figure 1a; for example, it can send the message to SNs_6, and from here to the sink using SNm_4.

The routing decision depends on the delay bound D, the time the message is generated, and the mobile supernodes' location at that time. In a delay tolerant application, the first approach is preferred. In a delay sensitive application, if SNm_4 is approaching SNs_3 and the message can be delivered to SNs_3 before SNm_4's arrival, then the second choice may be used if it satisfies the delay requirement. In this case, the message is picked up and carried faster to the sink by SNm_4. On the other hand, if a message arrives at SNs_3 after SNm_4's departure, then it has to wait almost another time period T before delivery. The fastest alternative is to use a pure sensor-to-sensor path from the source-sensor to the sink. However, this is the most expensive choice in terms of energy consumption, and it should be avoided.

4 Routing strategy

The routing process contains two phases. In the first phase, called *set-up phase*, beacon messages are broadcast and each sensor collects and stores routing information in its routing table. In the second phase, called *data reporting phase*, data messages are generated from sensors and sent to the sink. Each source-sensor makes routing decisions according to the information in its own routing table to satisfy the delay requirements and optimise the energy consumption. If supernodes are used in data delivery, a routing path has two subpaths:

1 a sensor-to-sensor path from the source-sensor to a supernode

2 a supernode-to-supernode path from the supernode to the sink.

If no supernodes are involved in data delivery, then a sensor-to-sensor path is used from the source-sensor to the sink.

4.1 Set-up phase

In the set-up phase, each static supernode first computes its transmission time to the sink T_{SNs}-$_S$, assuming a message is picked up by the corresponding mobile supernode immediately. The values will be propagated and stored in sensor routing tables.

4.1.1 Computation of T_{SNs}-$_S$

Static supernodes are divided into three groups and labelled according to their locations in Figure 2a: group G including supernodes $G(i,j)$, group N including supernodes $N(i,j)$, and group S including supernodes $S(i,j)$, where i is the row number (from the base line) and j is the column number. We use the following representation in the figure: a grey square for a $G(i,j)$ supernode, a black square for a $N(i,j)$ supernode, and a black circle for a $S(i,j)$ supernode. Each static supernode computes $T^{(i,j)}_{SNs-S}$ according to Theorem 1 assuming that a message is being immediately picked up by a mobile supernode.

Theorem 1: *(Computation of $T_{SNs\text{-}S}$) The transmission time from a static supernode to the sink, assuming a message is being immediately picked up by the corresponding mobile supernode, is computed according to the following formulas*:

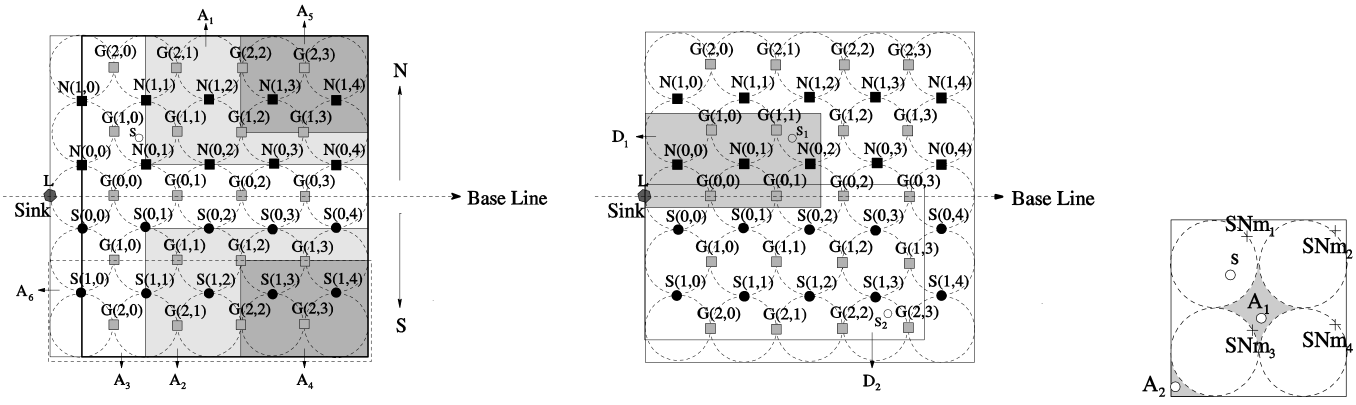
- $T^{G(i,j)}_{SNs-S} = \frac{T}{2} + (i+j) \times T$, *for a supernode in group G*
- $T^{N(i,j)}_{SNs-S} = \frac{T}{4} + (i+j) \times T$, *for a supernode in group N*
- $T^{S(i,j)}_{SNs-S} = \frac{3T}{4} + (i+j) \times T$, *for a supernode in group S.*

The transmission time from a static supernode a to the sink is:

$$T^{a}_{SNs-S} = \frac{T}{2} + \min_{nh \in NH(a)} \left\{ T^{nh}_{SNs-S} + T_{a-nh} \right\} \tag{1}$$

where $NH(a)$ is the set of next exchange-points of the static supernode a. A next exchange-point is a possible next-hop static supernode where the messages picked up from a can be dropped off by the mobile supernode. T_{a-nh} denotes the transmission time from the static supernode a to the static supernode nh.

Figure 2 Broadcast area and delivery area. (a) Static supernodes broadcast area. (b) Sensors delivery area towards static supernodes. (c) Mobile supernodes broadcast area outside the circular trajectories



(a) (b) (c)

For example, for $G(0,1)$ in Figure 2a, one next exchange-point is $G(0,0)$. It takes $T_{a-nh} = \frac{T}{2}$ from $G(0,1)$ to $G(0,0)$. Since all mobile supernodes have the same circular trajectory and the same velocity, all static and mobile supernodes know the current location of any mobile supernode. When the message reaches $G(0, 0)$, the mobile supernode that will pick up and deliver $G(0,0)$'s messages to the sink is located at L. Thus, it takes an additional $\frac{T}{2}$ time for this mobile supernode moving from L to $G(0,0)$ to pick up the message. Since the transmission time from $G(0, 0)$ to the sink is $\frac{T}{2}$, the total transmission time for $G(0,1)$ is computed. If there are several possible next exchange-points, the one resulting in the shortest transmission time is selected.

Next, we prove that Theorem 1 holds for static supernodes in the northern part of the base line, for the groups G and N. $T_{SNs-S}^{G(0,0)} = \frac{T}{2}$ and $T_{SNs-S}^{N(0,0)} = \frac{T}{4}$ can be computed easily and they comply with the theorem formulas.

Let us compute T_{SNs-S} for the static supernodes in the first row of groups G and N. For both $G(0, j)$ $(j \neq 0)$ and $N(0,j)$ $(j \neq 0)$, the best next exchange-point that provides the shortest transmission time is $G(0,j-1)$. This complies with Theorem 1 when the transmission time is computed according to formula 1. The best next exchange-point for static supernodes in the first column of N and G, $N(i,0)$ and $G(i,0)$ with $(i \neq 0)$, is $N(i-1,0)$. This result also complies with the theorem. Based on the known results of the supernodes in the first row and the first column, the transmission time of the other supernodes can be computed one by one.

For any supernode $G(i,j)$, $(i \neq 0$ and $j \neq 0)$, the best next exchange-point is $G(i,j-1)$ or $N(i-1,j)$. If it is $G(i,j-1)$, then according to formula 1, $T_{SNs-S}^{G(i,j)} = \frac{T}{2}+\frac{T}{2}+\frac{T}{2}+(i+j-1)\times T = \frac{T}{2}+(i+j)\times T$. The same result is obtained when $N(i-1, j)$ is selected as the next exchange-point.

For $N(i, j)$ $(i \neq 0$ and $j \neq 0)$, the best next exchange-point is $G(i,j-l)$ or $N(i-1,j)$. Using similar computations, the theorem result is verified. The proof for the static supernodes in the southern part of the base line is similar.

Each static supernode records in its routing table its T_{SNs-S} value and the best next exchange-point as the next hop.

4.1.2 Beacon messages broadcasting

At the set-up phase, each static supernode broadcasts one beacon message *Beacon* (*SNsID*, *TTL*, *timeMark*, *height*, T_{SNs-S}), containing the identifier of the supernode, *TTL* (Time-To-Live), the time the message is generated, the number of hops to the static supernode (initialised to 0), and T_{SNs-S}. A static supernode's beacon message is flooded only a specific *broadcast area*, which is defined as follows.

Definition 1: Broadcast area of static supernodes

1. The broadcast area of a static supernode $G(i,j)$ $(i > 0)$ in the northern part of the base line is a rectangle with four boundaries: the line connecting $N(i-1,j)$, $N(i-1,j+1)$, the line connecting $N(i-l,1)$, $N(i,j)$, and two boundaries of the monitored area.
2. The broadcast area of a static supernode $G(i,j)$ $(i > 0)$ in the southern part of the base line is a rectangle with four boundaries: the line connecting $S(i-1,j)$, $S(i-1, j+1)$, the line connecting $S(i-1,j)$, $S(i,j)$, and two boundaries of the monitored area.
3. The broadcast area of a static supernode $G(i,j)$ $(i = 0)$ located on the base line is a rectangle with four boundaries: the line connecting $N(i,j)$, $S(i,j)$, and three boundaries of the monitored area.
4. The broadcast area of a static supernode $N(i,j)$ or $S(i,j)$ $(i > 0)$ is a rectangle with four boundaries: the line connecting $G(i,j-1)$, $G(i,j)$, the line connecting $G(i,j-1)$, $G(i+1, j-1)$, and two boundaries of the monitored area.
5. The broadcast area of a static supernode in the first column of N or S, $N(i,j)$ or $S(i,j)$ $(j = 0)$, is a rectangle with four boundaries: the line connecting $G(i,j)$, $G(i,j+1)$, and three boundaries of the monitored area.

Flooding only inside the broadcast area reduces the area where Beacon messages propagate, reducing the unnecessary overhead. TTL mechanism is optional and can be used to further reduce the broadcast area and overhead. When the broadcast area is still too large and broadcasting in the whole area generates too much overhead, TTL can be adopted to reduce the number of transmissions.

The beacon message is broadcasted using controlled flooding and TTL is used to limit the maximum distance the message is propagated. Every sensor records a *minHeight*, initialised to infinite. We assume that each sensor knows its location through GPS or other localisation protocols. When a sensor s receives a beacon message, it compares its location with the broadcast area, if it is inside the broadcast area and if the *height* value in the message is less than *minHeight*, then s records the information in the beacon, and the sensor from which the message was received is recorded as the next hop to the supernode. Sensor s replaces *minHeight* with *height* and decreases *TTL* by 1. If *TTL* is greater than 0, then s increases *height* by 1 and broadcasts the message (with the updated *TTL* and *height*) to its neighbours. Otherwise, if the sensor is not inside the broadcast area, it drops the received message. Supernodes do not forward other supernodes' beacon messages.

Figure 2a shows some examples of broadcast areas of static supernodes. Light grey areas A_1 and A_2 are the broadcast areas for $G(1, 1)$ in the northern and southern part, respectively. A_3 with thick boundary lines is the broadcast area of $G(0, 0)$. A_4 and A_5 are the broadcast areas for $S(1, 3)$ and $N(1, 3)$. A_6, with dashed boundaries, is the broadcast area for $S(1,0)$. It is not necessary to send the static supernode's beacon message to the sensors outside the corresponding broadcast area considering transmission delay and energy consumption. In the Theorem 2, we prove it from these two aspects.

Theorem 2: *(Validation of the broadcast area of static supernodes) Broadcasting beacon messages from static supernodes within their broadcast areas does not affect the transmission delays or the energy consumption.*

Using supernodes, the propagation delay to the sink has three parts, as presented in formula 2. If the first relay supernode is static, $T_{s-SN||S}$ is the transmission time from the sensor to this supernode, $T_{waiting}$ is the waiting time in the static supernode until the message is picked up by a mobile supernode, and T_{SN_8-S} is the transmission time from the static supernode to the sink (computed according to the Theorem 1).

$$delay = T_{s-SN||S} + T_{waiting} + T_{SN_8-S} \qquad (2)$$

Taking $G(1,1)$'s broadcast area A_1 as an example in Figure 2a, we show that any sensor outside A_1 can find a better static supernode than $G(1,1)$. Let us take the sensor s located near $N(0,1)$, see Figure 2a. Regarding the energy consumption or the delay criteria, s can always choose a static supernode better than $G(1,1)$, such as $G(1,0)$ or $N(0,1)$.

Since s is closer to $G(1,0)$ or $N(0,1)$ compared to $G(1,1)$, it saves energy by forwarding the messages through $G(1,0)$ or $N(0,1)$. Regarding the delay, $G(1,0)$ and $N(0,1)$ have shorter T_{SNs-S}, which are $1\frac{1}{2}T$ and $1\frac{1}{4}T$, respectively, according to formula 1, compared to $T_{SNs-S}^{G(1,1)} = 2\frac{1}{2}T$, and shorter $T_{s-SN||S}$ since s is closer to $G(1,0)$ and $N(0,1)$.

In the worst case, $T_{waiting}$ is larger for $G(1,0)$ and $N(0,1)$, depending on the current position of the mobile supernode. For $G(1,0)$, the longest waiting time is when the mobile supernode has just passed $G(1,0)$, thus $T_{waiting} = T$. In this case, the waiting time for $G(1,1)$ is $T_{waiting} = \frac{T}{2}$, so $G(1,0)$ has a $\frac{T}{2}$ longer waiting time. Similarly, when the mobile supernode is at $G(1,1)$, $N(0,1)$ has a $\frac{3}{4}T$ longer waiting time. But the waiting time difference does not overweight the benefit gained from T_{SNs-S}, and according to formula 2, the total delay when choosing $G(1,0)$ or $N(0,1)$ is shorter than choosing $G(1,1)$. The static supernodes outside A_1 are closer to the sink and thus provide better choices than $G(1,1)$.

A similar analysis can be done for other sensors outside A_1, and in general for the other static supernodes. Broadcasting beacon messages only inside the broadcast areas instead of the whole network helps reducing the overall overhead.

Definition 2: A sensor's delivery area towards static supernodes is a rectangle with edges defined by the following lines: the base line (and its supernodes), the left edge of the monitored area, the line formed by the nearest column of supernodes at the right of the sensor, and the line formed by the nearest row of supernodes at the north (or south) of the sensor. For a source-sensor, only static supernodes in this area are stored in its routing table.

Figure 2b shows examples of delivery areas. D_1 (the grey area) is the delivery area for s_1 and D_2 is the delivery area for s_2 Taking s_1 as an example, according to Theorem 2, we can exclude $N(1,0)$, $N(1, 1)$, $N(1, 2)$, $G(1,2)$, $G(0,2)$, $S(0,0)$, $S(0,1)$, $S(0,2)$ and other static supernodes located farther away from the sink. Then the possible candidates are limited to those in the delivery area of s_1. Based on Theorem 2, each sensor stores in its routing table only paths to the static supernodes in its *delivery area.*

Broadcast areas and the *TTL* parameter are used to reduce the number of static supernodes in sensors' routing tables. To make further reductions, a sensor could store only beacon messages from static supernodes with the smallest T_{SNs-S}.

Besides static supernodes, mobile supernodes also broadcast beacon messages. A mobile supernode periodically broadcasts *Beacon* (*SNmID*, *TTL*, *timeMark*, *height*, T_{SNs-S}, *dropPos*) during the first period *T*, where *height* is the number of hops to the location where the mobile supernode broadcast this beacon, *dropPos* is the drop-off position of the mobile supernode (which is defined below), and T_{SNs-S} is the delivery time from *dropPos* to the sink.

A mobile supernode will drop off a message to the static supernode that minimises the delivery time to the sink:

$$\text{drop-off position} = \begin{cases} \text{position 2, case 1} \\ \text{position 1, case 2} \\ \text{position 0, otherwise} \end{cases}$$

where position numbers are the ones shown in Figure 1b. Case 1 applies to trajectories which are completely above the base line. Case 2 is for trajectories that intersect or are located below the base line and have a static supernode in position 1.

When defining the broadcast areas of mobile supernodes, we distinguish two cases, when a sensor is located inside a circular region or inside a star-shaped region (e.g. the grey area A_1 in Figure 2c). In the first case, a sensor only keeps and forwards beacons from the mobile supernode of that circular region. Sensor s in Figure 2c only records and forwards beacons from the mobile supernode SNm_1. In the second case, a sensor stores beacons from the mobile supernodes whose quarter circular trajectories form the star-shaped region. In Figure 2c, boundaries of the grey area A_1 are consecutively covered by four mobile supernodes SNm_1, SNm_2, SNm_3 and SNm_4. Sensors inside A_1 record beacon messages sent by these four mobile supernodes when they move along the boundaries of A_1. Sensors inside the gray area A_2 receive beacon messages only from the mobile supernode SNm_3 when it moves along the left quarter of its circular trajectory.

The sink broadcasts a beacon message *Beacon* (*timeMark*, *height*), where *height* is the number of hops to the sink, in the whole network, so that each sensor will know a path to the sink without involving supernodes. This information is stored as a back-up and is used in cases when delay constraints do not allow using supernodes for data forwarding.

Using the information in the beacons, a sensor learns about possible relay supernodes. An entry in a sensor's routing table has the following fields: (*SNsID/SNmID/Sink*, *timeMark*, *nextHop*, *minHeight*, $T_{SNs\text{-}S}$, *dropPos*), where *minHeight* is the minimum number of hops to the supernodes or to the sink.

4.2 Data reporting phase

When a data message is generated, the routing path is decided such that it satisfies the delay requirement and is energy efficient (minimum number of sensor relaying). The routing strategy decides the type of data delivery: (a) using pure sensor-to-sensor communication, or (b) using supernodes. If data delivery is done using supernodes, then the mechanism selects the first supernode, which can be a mobile supernode or a static supernode. The decision on the data delivery type is described next.

Let us denote with d the maximum delay between two one-hop away sensors. A source-sensor uses formula 2 to compute the total delivery delay for every candidate in its routing table. In formula 2, $T_{s-SN\|S}$ is computed using formula 3, where the *minHeight* is the corresponding field in the routing table.

$$T_{s-SN\|S} = minHeight \times d \tag{3}$$

The mobile supernode location on the trajectory when a message reaches the first relay supernode is computed as:

$$location = \left(current\_time + T_{s-SN\|S}\right)\%T \tag{4}$$

where % means a residue, which can be a decimal. We assume that at the time 0 all mobile supernodes are in the position 0 (see Figure 1b).

If the first relay supernode is a static supernode, then the sensor knows the $T_{SNs\text{-}S}$ value from its routing table. $T_{s-SN\|S}$ is estimated using formula 3. The source-sensor also computes $T_{waiting}$ since it knows the position of the static supernode and the position of the mobile supernode which picks up the message (computed using formula 4). The total transmission time *delay* is computed according to formula 2.

If the first relay supernode is a mobile supernode, then $T_{s-SN\|S}$ is estimated using formula 3. The message will be sent to a sensor near the trajectory, from which the beacon was transmitted in the set-up phase. $T_{waiting}$ is the sum of the time taken in waiting in the sensor node near the trajectory for the mobile supernode to come close enough, the time taken by the mobile supernode to carry the message to its drop-off position and the time taken in waiting in the drop-off position for another mobile supernode to pick it up. $T_{SNs\text{-}S}$ is the delivery time from the drop-off position to the sink, which is stored in its routing table.

If the message is sent through a pure sensor-to-sensor path, $T_{SNs\text{-}S}$ and $T_{waiting}$ are both 0. $T_{s-SN\|S}$ is estimated using formula 3.

Using the information in its routing table, a source-sensor can find multiple paths to the sink. It computes the *delay* of each candidate path and compares it with the delay bound D of the message. The path with the smallest *delay* is the pure sensor-to-sensor path. If this is the only path that has a delay smaller than or equal to D, then the message will be delivered using the pure sensor-to-sensor path. The message will be transmitted by sensors using the *nextHop* field for the sink entry in the routing tables.

If one or more supernode paths meet the delay requirement, then the transmission will be done using supernodes. The most energy-efficient path in terms of minimum number of forwarding sensors (*minHeight*) is selected. If there are more paths with the same *minHeight*, then any of them can be selected randomly. Once a source-sensor has selected a supernode path, it uses a sensor-to-sensor path to the first supernode. Sensors use their routing tables to decide the next hop where the message will be relayed. They use the *nextHop* field for the corresponding supernode entry in the table. The first supernode will then deliver the message to the sink using supernode-to-supernode communication.

4.3 Updating the routing table information

During the set-up phase, a sensor routing table records information to several static supernodes, up to four mobile supernodes, and to the sink. To prolong network lifetime and guarantee coverage and connectivity, sensors with energy less than a predefined threshold e_{th} do not participate in data relaying. To ensure this, the sink and supernodes periodically broadcast *Update* messages. These messages are used by sensors to update their paths to supernodes and to the sink. *Update* messages are similar to the *Beacon* messages, and they are broadcasted periodically. Only sensor nodes with energy at least e_{th} will forward *Update* messages, and thus only these sensors can be part of future message delivery paths.

A sensor node that receives an *Update* message will refresh the *timeMark*, *nextHop* and *minHeight* fields in its routing table. An entry in the routing table that has not been refreshed is removed, meaning that there is no delivery path to that supernode or to the sink.

The routing mechanism is summarised in the Algorithm 1.

Algorithm 1 DERP-Algorithm

1: Set-up phase
   - Static supernodes record the best next exchange point and mobile supernodes record their drop-off positions in their routing tables. $T_{SNs\text{-}S}$ is computed (Theorem 1).
   - The sink and all supernodes broadcast *Beacon* messages in their broadcast areas. Sensors record paths to them.

2: Data reporting phase
   - Data messages are generated.
   - A source-sensor computes the *delay* for every possible candidate path using formula 2. If only the pure sensor-to-sensor path meets the delay constraint then this path is used. If one or more supernode paths meet the delay constraint, then the one involving a minimum number of sensor relaying is selected.

3: Updating the routing table information
   - The sink and all supernodes periodically broadcast *Update* messages in their broadcast area. Only sensors with energy greater than a threshold forward *Update* messages.

5 Simulations

5.1 Simulation environment

Sensors are randomly deployed in a square of 30 × 30 area units. Network lifetime is organised in rounds. In each round, every sensor which is alive sends one periodic data reporting message. In addition, 20% of the sensors send an event-based reporting message. We take the maximum delay between two neighbour sensors $d = 1$.

In our simulations, we compute the energy consumption similar to LEACH (Heinzelman et al., 2000). The energy consumption for transmitting and receiving a message is $E_{Tx} = 50 \times 10^{9} \times msg\_length + 100^{-12} \times msg\_length \times r^2$ and $E_{Rx} = 50 \times 10^{-9} \times msg\_length$, where r is the transmission range and *msg_length* is the length of the message. We take the length of a data message to be eight times the length of a control message (*Beacon* and *Update* message). The initial energy of each sensor is 10,000 units. To transmit and receive a control message, a sensor consumes about 1 unit of energy. To transmit and receive a data message, a sensor consumes about 8 units of energy.

We study the following metrics. *Network lifetime* is defined as the number of rounds the network lasts until the first message cannot be delivered to the sink. *Number of delivered messages* represents how many data messages are successfully delivered to the sink during the network lifetime. *Miss ratio* measures the percentage of data messages that do not meet the delay requirements. *Average delay* is computed as $\frac{\sum delay}{numOfDataMsg}$, i.e. the sum of actual delivery delays of all data messages including periodic and event-based reporting, divided by the total number of delivered data messages. When a data message is to be delivered to a mobile supernode, it may need to be buffered by a sensor near the trajectory until the mobile supernode comes close enough. This is reflected when we study the *number of buffered messages.*

The *percentage of data messages through supernodes* is a metric used to show how many event-based data messages are delivered through supernodes instead of pure sensor-to-sensor paths and is computed as $\frac{numThroughsupernodes}{totalMsgEventNum}$. The *overhead* is defined as the total number of control messages (*Beacon* and *Update* messages) that are produced and forwarded in the network.

We conduct the simulation on a custom discrete event simulator. The initial sensor deployment is generated randomly. In the simulation, we use the following parameters:

- *NumberOfSensors* is the total number of sensors in the network.
- *TransRange* is the transmission range of a sensor, which is 3 units.
- *T* is the period during which a mobile supernode has completed a circular trajectory.
- *MAX_TTL* is the initial *TTL* in the control messages.
- *Radii* is the radii of a circular trajectory.
- *BroadcastPeriod* is the period after which mobile supernodes broadcast control messages. If *BroadcastPeriod* = 0, then mobile supernodes do not broadcast control messages; they are used only to carry messages between static supernodes. If $BroadcastPeriod = \frac{1}{8}T$, then the mobile supernodes broadcast every $\frac{1}{8}T$, excluding the positions where static supernodes are located.
- Periodic data reporting has no delay requirement. Event-based reporting uses three delay classes, DB_1, DB_2, and DB_3. Class DB_3 is for the most urgent events, and DB_1 is for the least urgent events.
- The update process is triggered every one round, two rounds, or three rounds, using *UpdatePeriod* = 1, 2, or 3, respectively. A sensor with residual energy less than *E_TH_DEAD* = 200 units is considered dead and will not send any messages. A sensor with residual energy less than *E_TH_FORWARD* will issue data messages but does not forward messages on behalf of other sensors. *E_TH_FORWARD* takes the values 3000, 4000, or 5000 units for the three *UpdatePeriod* values.

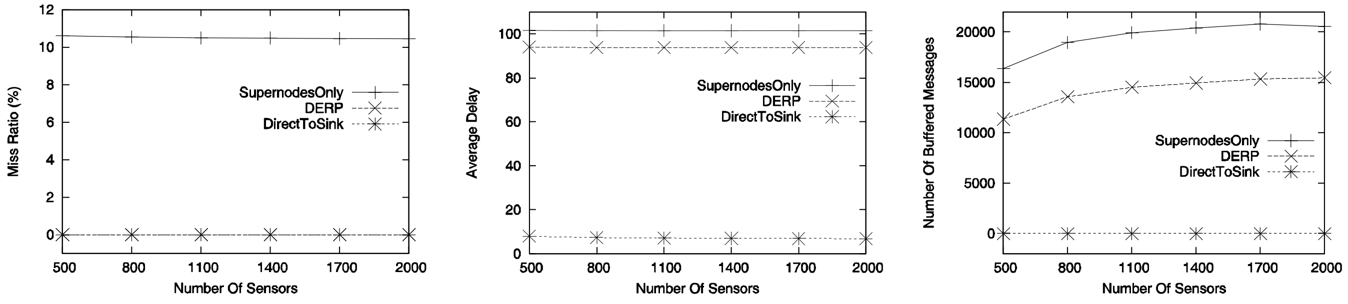
All the tests are repeated 100 times. The collected data is averaged and reported in the following figures.

5.2 Simulation results

In Figures 3 and 4, $T = 40$ units, $BroadcastPeriod = \frac{1}{8}T$, *Radii* = 5 units, *MAX_TTL* = 50 units, $DB_1 = 120$, $DB_2 = 80$, $DB_3 = 40$ units, and *UpdatePeriod* = 1. In the *SupernodesOnly* algorithm, each sensor sends its data messages to the closest supernode, without considering the delay requirements. The *SupernodesOnly* algorithm is similar to many cluster-based data delivery mechanisms where sensors always send data to their closest cluster heads. The *DirectToSink* algorithm applies to homogeneous sensor networks which do not use supernodes. Every data message is delivered to the sink using a pure sensor-to-sensor path. *DERP* is the complete DERP-Algorithm.

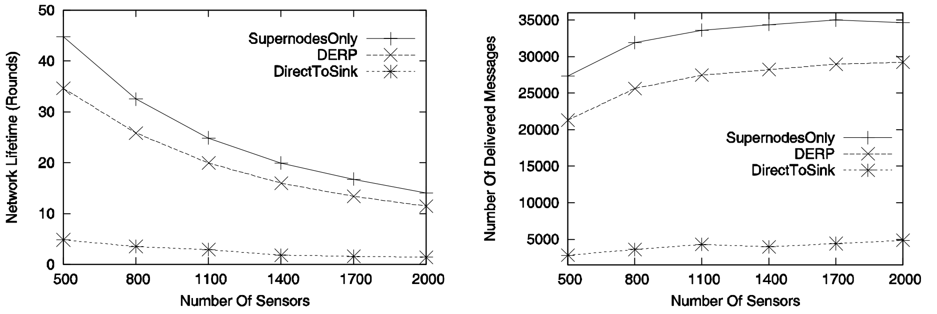
Figure 4a shows that *SupernodesOnly* has the longest network lifetime and *DirectToSink* has the shortest one. This happens because the *SupernodesOnly* uses the smallest number of sensor data relaying. Thus the energy consumed in data forwarding is saved. The network lifetime decreases when the number of sensors increases. This happens because in the simulation, in each round, each sensor which is alive sends one periodical data and besides, 20% of the sensors send one event-based report. In this case, when there are more sensors deployed in the area, in each round, more reports are generated and delivered. Therefore in general, sensors which are closer to the sink forward more messages when there are more sensors and they are prone to die faster, which causes a shorter network lifetime (smaller number of rounds).

Figure 3 Comparison between the three algorithms. (a) Miss ratio (b) Average delay (c) Number of buffered data messages



(a) (b) (c)

Figure 4 Comparison between the three algorithms. (a) Network lifetime (b) Number of data messages that are successfully sent to the sink.



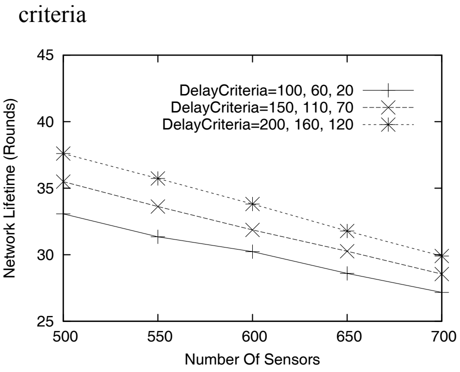
(a) (b)

Figure 4b compares the number of data messages that are successfully sent to the sink. *SupernodesOnly* has the best performance which is consistent with the results in Figure 4a. Although *SupernodesOnly* has the longest network lifetime, some of the data messages do not satisfy the delay requirements. Figure 3a shows the miss ratio. *DERP* and *DirectToSink* have the miss ratio 0 since all the messages are delivered on time. In Figure 3b, *SupernodesOnly* has the highest average delay and *DirectToSink* has the lowest. The density does not affect the average delay and the miss ratio in general. For a sparser network, increasing the sensor density can reduce the length of a sensor-to-sensor path, until it reaches the shortest path, but this is not the dominant factor in formula 2. In Figure 3c, no message has to be buffered in the *DirectToSink* approach since no mobile supernode is being used. *SupernodesOnly* buffers more messages than *DERP* since more mobile supernodes are involved in data delivery.

Figure 5 compares the network lifetime for different delay criteria. $T = 40$, $BroadcastPeriod = \frac{1}{8}T$, $Radii = 5$, $MAX\_TTL = 50$, and $UpdatePeriod = 1$. For a looser delay requirement the network lifetime is longer since more messages are delivered using supernodes. For tighter delay requirements, more messages are sent through sensor-to-sensor paths, and thus more energy is consumed by sensor data relaying.

In Figure 6, $BroadcastPeriod = \frac{1}{8}T$, $Radii = 5$, $MAX\_TTL = 50$, $UpdatePeriod = 1$, $DB_1 = 120$, $DB_2 = 80$, $DB_3 = 40$, and T varies. In Figure 6b, when T is smaller, more messages are delivered by supernodes. $T = 10$ and $T = 20$ get close curves, with the average length of the senor-to-sensor paths to supernodes being larger for $T = 20$. This explains the results from Figure 6a: the smaller the T is, the longer the network lifetime is. In Figure 7, $BroadcastPeriod = \frac{1}{8}T$, $MAX\_TTL = 50$, $UpdatePeriod = 1$, $DB_1 = 120$, $DB_2 = 80$, $DB_3 = 40$ and *Radii* varies. In all the three cases, the mobile supernodes move at the same speed, thus $T = 24$, 40 and 120, respectively. In Figure 7a, $Radii = 5$ has the longest network lifetime. When $Radii = 15$, network lifetime is longer than the case $Radii = 3$. This is because when the radii is 3, there are many supernodes in the network, and thus a large number of control messages are broadcasted in the set-up phase and during the update process. $Radii = 15$ has a small number of supernodes, T is large, and most of the messages are delivered through sensor-to-sensor paths, as inferred from Figure 7b.

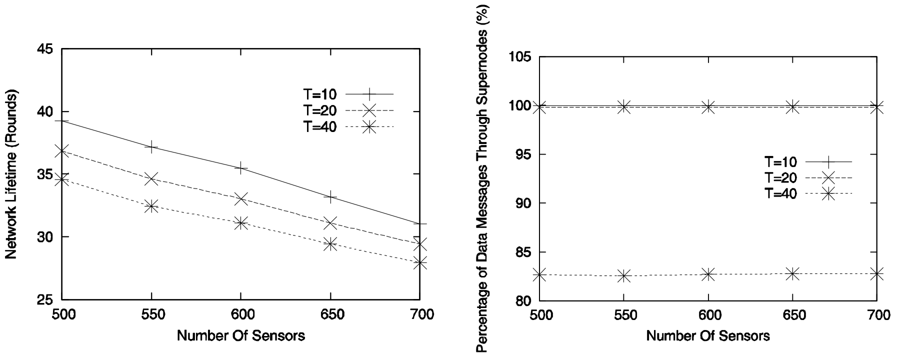
Figure 5 Comparison between the three delay requirement criteria



In Figure 8, $T = 40, BroadcastPeriod = \frac{1}{8}T$, $UpdatePeriod = 1$, $Radii = 5$, $DB_1 = 120$, $DB_2 = 80$, $DB_3 = 40$, and *MAX_TTL* varies. The figure shows that the larger the *MAX_TTL* is, the longer the network lifetime and the overhead are. When *MAX_TTL* is larger, sensors get information about more supernodes and thus have more entries in their routing tables. A data message has therefore more delivery choices and higher possibilities to be delivered using supernodes.

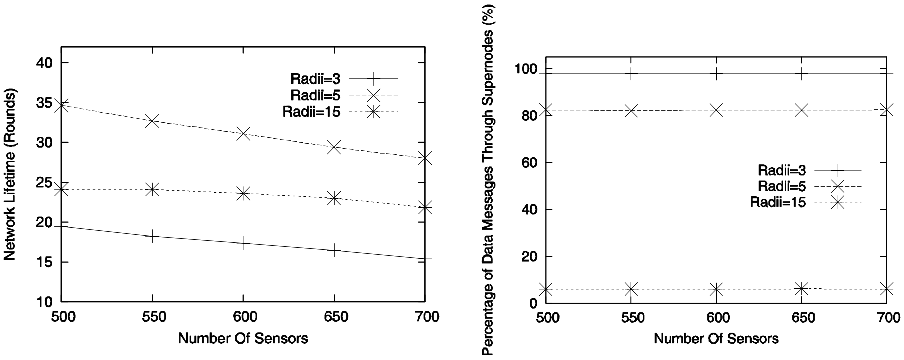
In Figure 9, $T = 40, BroadcastPeriod = \frac{1}{8}T$, $MAX\_TTL = 50$, $Radii = 5$, $DB_1 = 120$, $DB_2 = 80$, $DB_3 = 40$ and *UpdatePeriod* varies. The figure shows that for an even update period, $UpdatePeriod = 1$ produces the highest overhead and the shortest network lifetime compared with the other two cases. In practice, the frequency of update should be adaptable: lower frequency at the beginning and higher frequency when more delivery failures occur.

Figure 6 Comparison between three different periods. (a) Network lifetime (b) Percentage of messages delivered through supernodes



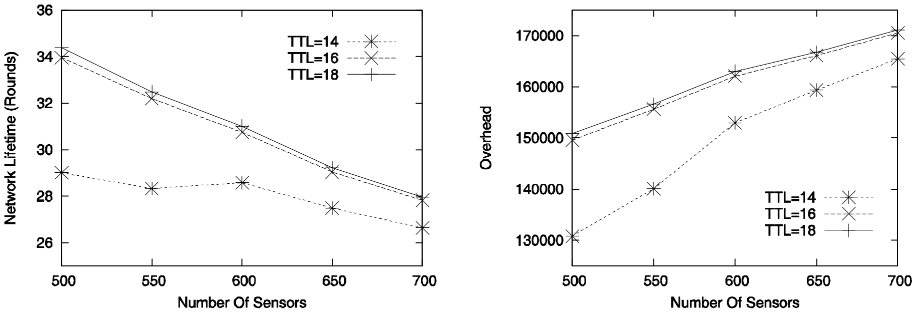
(a) (b)

Figure 7 Comparison between three different radii. (a) Network lifetime (b) Percentage of messages delivered through supernodes



(a) (b)

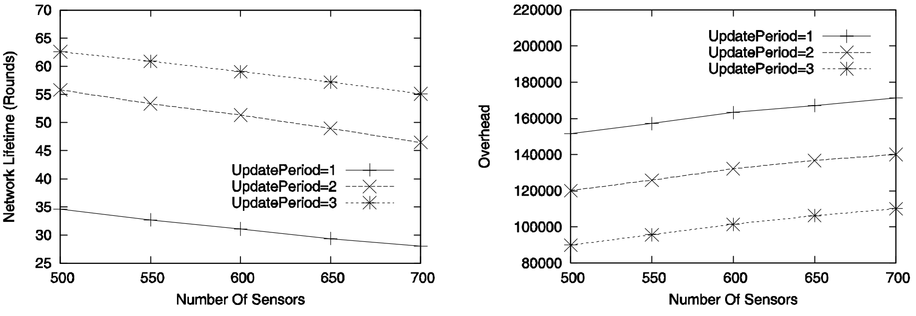
Figure 8 Comparison between three different TTLs. (a) Network lifetime (b) Overhead



(a) (b)

Figure 10 uses $T = 40$, $MAX\_TTL = 50$, $UpdatePeriod = 1$, $Radii = 5$, $DB_1 = 120$, $DB_2 = 80$, $DB_3 = 40$, and *BroadcastPeriod* varies. In Figure 10a, the longest network lifetime is obtained when $BroadcastPeriod = \frac{1}{8}T$. $BroadcastPeriod = 0$ gets the second best performance. The case when mobile supernodes broadcast more frequently gets shorter network lifetime. When the mobile supernodes do not broadcast, only paths to static supernodes are stored in sensor routing tables, and many messages will be delivered through longer sensor-to-sensor paths. On the other hand, if mobile supernodes broadcast too frequently, they produce large overhead (see Figure 10b) which results in large energy consumption.

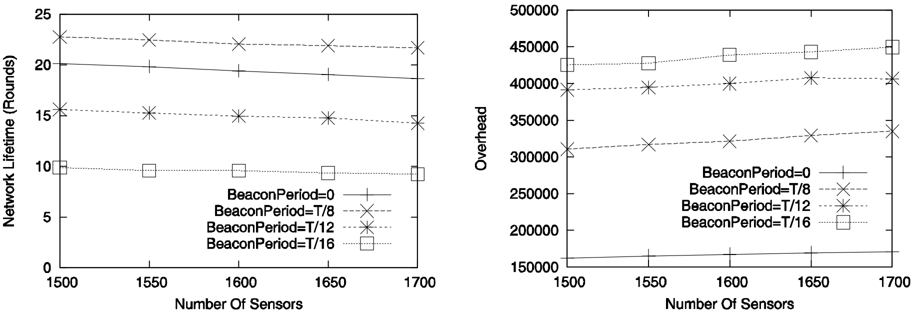
Figure 9 Comparison between three different update periods. (a) Network lifetime (b) Overhead



(a)

(b)

Figure 10 Comparison between three different periods used by the mobile supernodes to broadcast control messages. (a) Network lifetime (b) Overhead



(a)

(b)

6 Conclusions and future work

In this paper, we study the DERP problem and propose our solution, which includes two phases. In the set-up phase, the sink and supernodes broadcast *Beacon* messages, and sensors update their routing tables. In the data reporting phase, periodic and event-based messages are delivered to the sink with the help of supernodes or using pure sensor-to-sensor paths. The source-sensor makes the decision based on the message delay requirement and selects a path that involves a minimum number of sensor relaying.

For our future work, we plan to extend the routing strategy to the case when mobile supernodes' trajectory is more diverse.

Acknowledgements

This work was supported in part by the NSF grant CCF 0545488.

References

Akyildiz, I., Su, W., Sankarasubramaniam, Y. and Cayirci, E. (2002) 'Wireless sensor networks: a survey', *Computer Networks*, Vol. 38, No. 4.

Cardei, M. and Yang, Y. (2008) 'Heterogeneous wireless sensor networks', in Furht, B. (Ed.): *Encyclopedia of Wireless and Mobile Communications*, CRC Press, Taylor & Francis Group.

Chen, Z., Kung, H. and Vlah, D. (2001) 'Ad hoc relay wireless networks over moving vehicles on highways', Poster, *ACM Mobihoc.*

Dantu, K., Rahimi, M.H., Shah, H., Babel, S., Dhariwal, A. and Sukhatme, G.S. (2005) 'Robomote: enabling mobility in sensor networks', *IPSN 2005*, pp.404–409.

He, T., Huang, C., Blum, B.M., Stankovic, J.A. and Abdelzaher, T. (2003) 'Range-free localization schemes for large scale sensor networks', *ACM Mobicom.*

Heinzelman, W., Chandrakasan, A. and Balakrishnan, H. (2000) 'Energy-efficient communication protocols for wireless microsensor networks', *HICSS.*

Hu, L. and Evans, D. (2004) 'Localization for mobile sensor networks', *ACM Mobicom.*

Lee, U., Magistretti, E.O., Zhou, B.O., Gerla, M., Bellavista, P. and Corradi, A. (2006) 'Efficient data harvesting in mobile sensor platforms', *PerCom Workshops*, pp.352–356.

Liu, C. and Wu, J. (2007) 'Scalable routing in delay tolerant networks', *ACM MobiHoc.*

Luo, J. and Hubaux, J. (2005) 'Joint mobility and routing for lifetime elongation in wireless sensor networks', *IEEE INFO COM.*

McNeff, J.G. (2002) 'The global positioning system', *IEEE Transactions on Microwave Theory and Techniques,* Vol. 50, pp.645–652.

Ngai, E.C-H., Liu, J. and Lyu, M.R. (2007) 'An adaptive delay-minimized route design for wireless sensor-actuator networks', *MASS.*

Shah, R.C., Roy, S., Jain, S. and Brunette, W. (2003) 'Data MULEs: modeling a three-tier architecture for sparse sensor networks', *IEEE International Workshop on Sensor Network Protocols and Applications (SNPA).*

Somasundara, A., Kansal, A., Jea, D., Estrin, D. and Srivastava, M. (2006) 'Controllably mobile infrastructure for low energy embedded networks', *IEEE Transactions on Mobile Computing*, Vol. 5, No. 8.

Wang, W., Srinivasan, V. and Chua, K. (2005a) 'Using mobile relays to prolong the lifetime of wireless sensor networks', *MobiCom.*

Wang, Z.M., Basagni, S., Melachrinoudis, E. and Petrioli, C. (2005b) 'Exploiting sink mobility for maximizing sensor networks lifetime', *HICSS.*

Wu, J., Yang, S. and Dai, F. (2007) 'Logarithmic store-carry-forward routing in mobile ad hoc networks', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 18, No. 6.

Zhao, J. and Cao, G. (2006) 'VADD: vehicle-assisted data delivery in vehicular ad hoc networks', *INFOCOM.*