# Two Energy-Efficient Routing Algorithms for Wireless Sensor Networks

Hung Le Xuan, Sungyoung Lee
Department of Computer Engineering, Kyung Hee Unversity, Korea
{lxhung, sylee}@oslab.khu.ac.kr

**Abstract:** *Power Conservation is one of the most important challenges in wireless sensor networks. In this paper, we present two minimum-energy routing algorithms. Our main goal is to reduce power consumed and prolong the lifetime of the network. The first protocol, named CODE: **CO**ordination-based **D**ata dissemination for s**E**nsor networks, addresses the sensor networks consisting of mobile sinks. CODE considers energy conservation not only in communication but also in idle-to-sleep state. This protocol is based on GAF protocol and grid structure to reduce energy consumed. The second protocol, named SIDE: **SI**nk cluster – based data **D**issemination for s**E**nsor networks, addresses the sensor networks consisting of large number of stationary sinks. SIDE considers loosely resource-constrain of the sinks to ease the cost burden of sensor nodes. Our simulation results show that CODE and SIDE gain energy efficiency and prolong the network lifetime.*

## I. Introduction

A wireless sensor network is randomly deployed by hundreds or thousands of unattended and untethered sensor nodes in an area of interest. These networking sensors collaborate among themselves to collect, process, analyze and disseminate data. In the sensor networks, a *data source* is defined as a sensor node that either detects the stimulus or is in charge of sensing requested information. The sources are usually located where environment activities of interest take place. A *sink* is defined as a user's equipment such as PDA, laptop, etc. which gathers data from the sensor network.

Limitations of sensors in terms of memory, energy, and computation capacities give rise to many research issues in the wireless sensor networks. In recent years, a bundle of data dissemination protocols have been proposed [3]-[5], [8], [9], [12]. Most of these efforts focus on energy conservation due to the energy limitation and the difficulty of recharging batteries of thousands of sensors in hostile or remote environment. Generally, the power consumption of sensors can be used for three functionalities (a) the power consumed for transmission of packets (b) the power consumed for reception of packets and (c) the power consumed when the network is idle. Besides, recent studies have shown that radio communication dominates energy consumption in the sensor networks, rather than computation [30]. Therefore, power conservation is an especially important challenge at the communication layers.

Each sensor network possesses its own characteristics to cater different applications. An example of such applications is monitor and control of safety-critical military, environmental, or domestic infrastructure systems. Depends on each application, the sinks may be mobile while the sensors are stationary. On the other hand, a number of sinks may be large since many users may simultaneously access the sensor networks. In this paper, we propose two energy-efficient data dissemination approaches which have been built in our previous work [31], [32]. These protocols individually address the sensor networks consisting of mobile sinks and the sensor networks consisting of a large number of sinks.

The first algorithm, Coordination- based Data Dissemination protocol (or CODE for short), addresses mobile sinks. We are motivated by the fact that handling mobile sinks is challenge of large-scale sensor network research. Though many researches have been published to provide efficient data dissemination protocols to mobile sinks [3]-[5], [8], [9], [12], they have proposed how to minimize energy consumed for network communication, regardless idling energy consumption. In fact, energy consumed for nodes while idling can not be ignored [13]-[15]. M.Stemm *et al* [14] and Y.Xu *et al* [15] show that energy consumption for *idle:receive:send* ratios are *1:1.05:1.4*, respectively. Consequently, they suggest that energy optimizations must turn off the radio. Doing this not only simply reduces number of packets transmitted but also conserves energy both in overhearing due to data transfer, and in idle state energy dissipation when no traffic exists, especially in sensor networks with high node density. In CODE, we take into account of energy for both communication and idle. CODE provides an energy efficient data dissemination path to mobile sinks for coordination sensor networks. CODE is based on grid structure and coordination protocol GAF [11].

The second algorithm, Sink Cluster-based Data Dissemination protocol (or SIDE for short), addresses a large number of stationary sinks. We are motivated by the fact that future sensor networks will consist of high sink density, from several to tens and they are often far away from phenomena.

While existing protocols focus only on sensor node processing (e.g. in-network aggregation, query and data aggregation) to reduce the amount of data transmitted to sinks, SIDE exploits capacities of not only nodes but also sinks in order to reduce communication cost. Receiving data, a sink can act as a source's *Agent* to relay data to the other nearby sinks. Since the sink is not as tightly resource-constrained as sensor nodes, they can talk directly to each other or through a few hops in order to ease the cost burden for sensor networks.

To better understand the rest of the paper, we first describe the general protocol design goals of sensor networks in Section 2. Since the paper is composed of two different approaches which target different sensor network models, we separately present each protocol and its performance evaluation in Section 3 and Section 4. The discussion about benefit of each proposed approach is given right after its evaluation. Section 5 concludes the paper.

## II. Protocol Design Goals

The wireless sensor network has its own constrains that differs from ad hoc networks. Such constrains make designing routing protocol for sensor networks is very challenging [26]. First of all, sensor nodes are limited in power, processing capacities and memory. Those require careful resource management. Second, sensor nodes may not have global identifications (IDs). Therefore, classical IP-based protocol can not be applied to the sensor networks. Third, sensor nodes might be deployed densely in the sensor networks. Unnecessary nodes should be turned off its radio while guaranteeing connectivity of the entire sensor field. Fourth, generated data traffic has significant redundancy in it since multiple sensors may generate same data within the vicinity of a phenomenon. Such redundancy needs to be exploited by the routing protocols to improve energy and bandwidth utilization [28].

In order to design a good protocol for the sensor networks, such constrains should be managed in efficient manner. In this paper, we emphasize on three major design goals in data dissemination protocol for wireless sensor networks.

**Energy Efficiency/Network Lifetime**

Energy efficiency is the most important consideration due to the power constrain of sensor nodes. Recent studies have shown that radio communication is dominant consumer of energy in the sensor networks. Most of recent publications mainly focus on how to minimize energy consumption for sensor networks. Besides, multi-hop routing will consume less energy than direct communication, since the transmission power of a wireless radio is proportional to distance squared or even higher order in the presence of obstacle. However, multi-hop routing introduces significant overhead for topology management and medium access control [28]. Another characteristic of the common sensor networks is that sensor nodes usually generate significant redundant data. Therefore similar packets from multiple nodes should be aggregated so that the number of packets transmitted would be reduced [29]. Several work [11], [19] suggest that unnecessary nodes should be turned off to conserver energy and reduce collision.

**Latency**

The user is interested in knowing about the phenomena within a given delay. Therefore, it is important to receive the data in a timely manner [27], [29].

**Scalability**

Scalability is also critical factor. For a large scale sensor network, it is likely that localizing interactions through hierarchical and aggregation will be critical for ensure scalability [27]

Keeping these design goals in mind, in this paper we propose tow data dissemination protocols for large-scale sensor networks to achieve energy efficiency while guaranteeing a comparable latency with existing approaches.

## III. CODE: A Coordination-based Data Dissemination Protocols to Mobile Sinks.

We first present CODE which addresses the sensor networks consisting of mobile sinks. In CODE, we rely on the assumptions that all sensor nodes are stationary. Each sensor is aware of its residual energy and geographical location. Once a stimulus appears, the sensors surrounding it collectively process the signal and one of them becomes the source to generate data report [4]. The sink and the source are not supposed to know any *a-prior* knowledge of potential position of each other. To make unnecessary nodes stay in the sleeping mode, CODE is deployed above *GAF-basic* protocol [11]. Fig.1 depicts CODE general model where the routing algorithm is implemented above the GAF protocol. In this paper, we only focus on CODE routing algorithm. Details of GAF algorithm can be referred in [11].

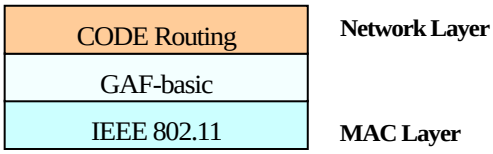



Fig.1.CODE system model

The basic idea of CODE is to divide sensor field into grids. Grids are indexed based on its geographical location. According to GAF, each grid contains one coordinator which acts as an intermediate node to cache and relay data. CODE consists of three phases: *data announcement*, *query transfer* and *data dissemination*. As a stimulus is detected, a source generates a *data-announcement* message and sends to all coordinators using simply flooding mechanism. Each

coordinator is supposed to maintain a piece of information of the source including the stimulus and the source's location. As a mobile sink joins in the network, it selects a coordinator in the same grid to act as its *Agent*. When it needs data, it sends a query to this *Agent*. The *Agent* is in charge of forwarding the query to the source based on the target's location and grid IDs. An efficient data dissemination path is established while the query traverses to the source. Receiving a query, the source sends the data to the sink along the data dissemination path. The Agent helps the sink to continuously keep receiving data from the source when the sink moves around. Periodically, the sink checks its location. If the sink moves to another grid, it first sends *cache-removal* message to clear out the previous data dissemination path and then re-sends a query to establish a new route.

## 1. CODE Theory

### *A. Grid Indexing*

We assume that we have partitioned the network plane in virtual *MxN* grids (for example in Fig.2 that is 3x2 grids). Each grid ID which has a typed [CX.CY] is assigned as follows: at the first row, from left to right, the grid IDs are [0.0], [1.0], and [2.0]. Likewise, at the second row, grid IDs are [0.1], [1.1], and [2.1] and so forth. To do this, based on the coordinate (x, y), each node computed itself CX and CY:

$$CX = \left\lfloor \frac{x}{r} \right\rfloor, \; CY = \left\lfloor \frac{y}{r} \right\rfloor \qquad (1)$$

where $r$ is the grid size and $\lfloor x \rfloor$ is largest integer less than $x$.

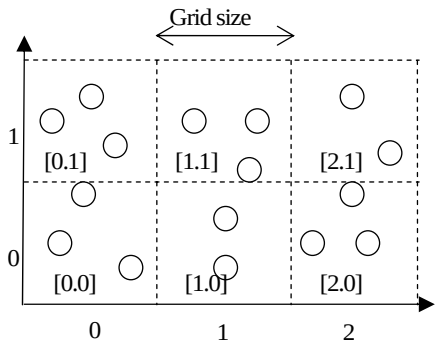



Fig.2.Grid Indexing

### *B. CODE Algorithms*

a*) Data Announcement*

When a stimulus is detected, the source propagates a *data-announcement* message to all coordinators using simply flooding mechanism. Every coordinator stores a few piece of information for the data dissemination path discovery, including the information of the stimulus and the source location. In this approach, the source location does not mean the precise location of the source, but its grid ID. Since the coordinator role might be changed every time, the grid ID is the best solution for nodes to know the target it should relay the query to. To avoid keeping data-announcement message at each coordinator indefinitely, a source includes a *timeout* parameter in *data-announcement* message. If this timeout expires and a coordinator does not receive any further *data-announcement* message, it clears the information of the stimulus and the target's location to release the cache.

*b) Query Transfer*

Every node is supposed to maintain a *Query INformation Table* (hereafter called QINT) in its cache. Each entry is identified by a tuple of (*query, sink, uplink*) (*sink* is the node which originally sends the *query*; *uplink* is the last hop from which the node receives the query). We define that two entries in QINT are identical if all their corresponding elements are identical. For example in Fig.3, node n1 and node n2 receive a query from sink1 and sink2, therefore it maintains a QINT as Fig.4.

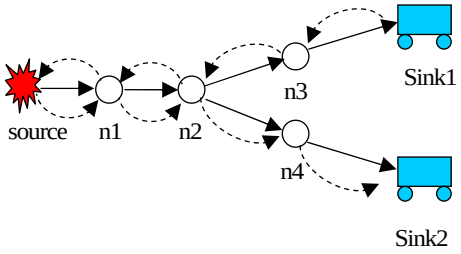



Fig.3.Query transfer and data dissemination path setup

| **Node n1** | | |
|---|---|---|
| **query** | **sink** | **uplink** |
| A | sink1 | n2 |
| A | sink2 | n2 |

| **Node n2** | | |
|---|---|---|
| **query** | **sink** | **uplink** |
| A | sink1 | n3 |
| A | sink2 | n4 |

Fig.4.Query information table maintained at nodes n1 and n2

Receiving a query from an uplink node, a node first checks if

the query exists in its QINT. If so, the node simply discards the query. Otherwise, it caches the query in the QINT. Then, based on target's location stored in each coordinator, it computes the ID of next grid to forward the query. This algorithm is described in Fig.5. In this figure, *NODE* is the current node handling the query packet and *src_addr* contains the target's location. If NODE is the source, it starts sending data along the data dissemination path. Otherwise, it finds the next grid which closest to the target to relay the query. In case the next grid contains no node (so-called *void grid*) or the next grid's coordinator is unreachable, it tries to find a round path. To do this, it first calculates the disparity $\delta_{CX}, \delta_{CY}$ as:

$$\Delta_{CX} = p->src\_addr.CX - NODE.CX, \quad \delta_{CX} = \frac{\Delta_{CX}}{|\Delta_{CX}|}$$

$$\Delta_{CY} = p->src\_addr.CY - NODE.CY, \quad \delta_{CY} = \frac{\Delta_{CY}}{|\Delta_{CY}|}$$

Then, the next grid ID will be:

$$NextGrid.CX = NODE.CX + \delta_{CX}$$

$$NextGrid.CY = NODE.CY + \delta_{CY}$$

```
Find_Next_Grid(NODE, packet* p)
{
If (NODE is Source)
    NODE.send_data();
Else{
    Δ_CX = p->src_addr.CX - NODE.CX;
    Δ_CY = p->src_addr.CY - NODE.CY;
    δ_CX = (Δ_CX == 0) ? 0 : Δ_CX / |Δ_CX|;
    δ_CY = (Δ_CY == 0) ? 0 : Δ_CY / |Δ_CY|;
    NextGrid.CX = NODE.CX + δ_CX;
    NextGrid.CY = NODE.CY + δ_CY;
    If (lookup_neighbor_table(NextGrid) == TRUE)
        return NextGrid;
    Else
        find_round_path();
  }
```

Fig.5.Pseudo-code of finding next grid ID algorithm

Each node is supposed to maintain a *one-hop-neighbor table*. (i.e. information about its one-hop neighbors). If a node can not find the next grid's coordinator in this table, it considers that grid as a void grid.

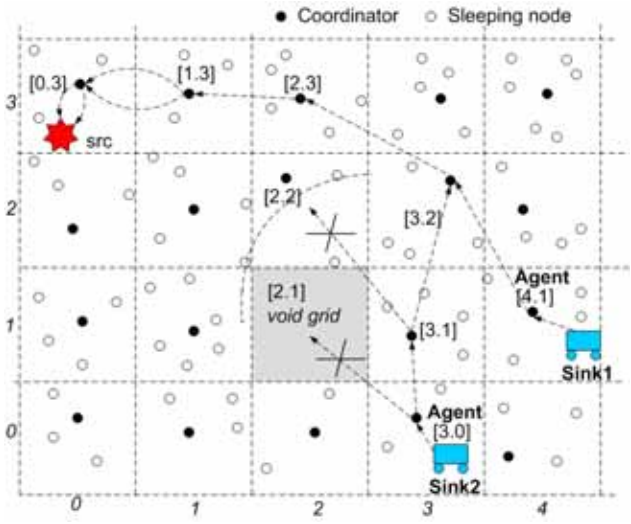



Fig.6.Multi-hop routing through coordinators

For example in Fig.6, the sink1 sends query to the source *src* along the path [4.1], [3.2], [2.3], [1.3], [0.3]. However, with the sink2, the grid [3.0]'s coordinator can not find grid [2.1]'s neighbor (due to void grid) and grid [3.1]'s coordinator also can not find grid [2.2]'s neighbor (due to unreachable node) in its one-hop-neighbor table. Therefore, it finds the round path as [3.1], [3.2], [2.3], [1.3], [0.3]. A data dissemination path is discovered by maintaining a QINT at each intermediate node. A query from a sink is re-transmitted when the sink moves to another grid.

*c) Data Dissemination*

A source starts generating and transmits data to a sink as it receives a query. Receiving data from another node, a node on the dissemination path (including the source) first checks its QINT if the data matches to any query and to which uplinks it has to forward. If it finds that the data matches several queries but with the same uplink node, it forwards only one copy of data. Doing this reduces considerable amount of data transmitted throughout the sensor network. For example in Fig.4, node n1 receives the same query A of sink1 and sink2 from the same uplink node (n2). Therefore, when n1 receives data, it sends only one copy of data to n2. Node n2 also receives the same query A of sink 1 and sink 2 but from different uplink nodes (n3, n4). Thus, it must send two copies of data to n3 and n4. Likewise, the data is relayed finally to the sinks.

*C. Handling Sink Mobility*

CODE is designed for mobile sinks. In this section, we describe how a sink keeps continuously receiving updated data from a source while it moves around within the sensor field.

Periodically, a sink checks its current location to know which grid it is locating. The grid ID is computed by the formula (1). If it is still in the same grid of the last check, the sink does nothing. Otherwise, it first sends a *cache-removal* message to its old Agent. The *cache-removal* message contains the query's information, the sink's identification and the target's location. The old Agent is in charge of forwarding the message along the old dissemination path as depicted in Fig.7. Receiving a *cache-removal* message, a node checks its QINT and removes the matched query. When this message reaches the source, the whole dissemination path is cleared out, i.e. each intermediate node on the path no longer maintains that query in its cache. Consequently, the source stops sending data to the sink along this dissemination path. After old dissemination path is removed, the sink re-sends a query to the target location. A new dissemination path is established as described in section (b) above. By doing this, the number of queries which is needed to be re-sent is reduced significantly compared with other approaches. Hence, collision and energy consumption is reduced. Also, the number of loss data packet is decreased. In case the sink moves into a void grid, it selects the closest coordinator to act as its Agent.

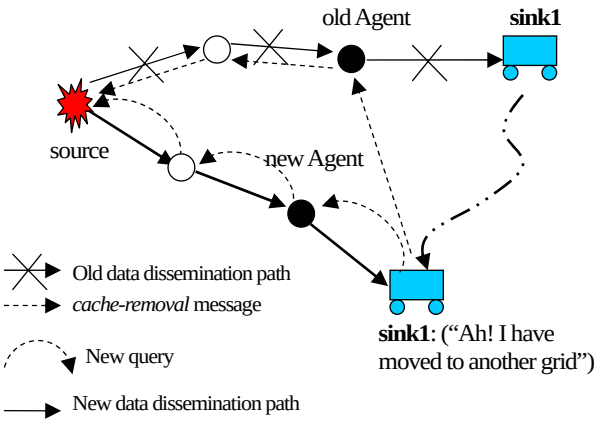



Fig.7.Handling sink mobility

## 2. CODE Performance

### *A. Simulation Model*

We developed a simulator based on SENSE simulator [22] to evaluate and compare CODE to other approaches such as Directed Diffusion (DD) [3] and TTDD [4]. To facilitate comparisons with TTDD and DD, we use the same energy model used in ns2 [25] that requires about 0.66W, 0.359W and 0.035W for transmitting, receiving and idling respectively. The simulation uses MAC IEEE 802.11 DCF that SENSE implements. The nominal transmission range of each node is 250m [11].

Our goal in simulating CODE is to examine how well it actually conservers power, especially in dense sensor networks. In the simulation, we take into account of total energy consumed for not only transmitting, receiving but also idling. The sensor network consists of *400* sensor nodes, which are randomly deployed in a *2000mx2000m* field (i.e. one sensor node per *100mx100m* grid). *Two-ray ground* is used as the radio propagation model and an omni-directional antenna having unity gain in the simulation. Each data packet has *64* bytes, query packet and the others are *36* bytes long. The default number of sinks is 8 moving with speed *10* m/sec (i.e. the fastest human speed) according to *random way-point* model [21]. Two sources generate different packets at an average interval of *1* second. Initially, the sources send a data-announcement to all coordinators using flooding method. When a sink needs data, it sends a query to its Agent. As a source receives a query, it starts generating and sends data to the sink along the data dissemination path. The simulation lasts for *200* seconds.

We use four metrics to evaluate the performance of CODE. The energy consumption is defined as the total energy network consumed. The success rate is the ratio of the number of successfully received packets at a sink and the total number of packet generated by a source, averaged over all source-sink pairs. The delay is defined as the average time between the time a source transmits a packet and the time a sink receives the packet, also averaged over all source-sink pairs. We define the network lifetime as the number of nodes alive over time.

### *B. Performance Results*

#### *a) Impact of Sink Number*

We first study the impact of the sink number on CODE. In the default simulation, we set the number of sink varying from 1 to 8 with the max speed 10m/s and a 5-second pause time.

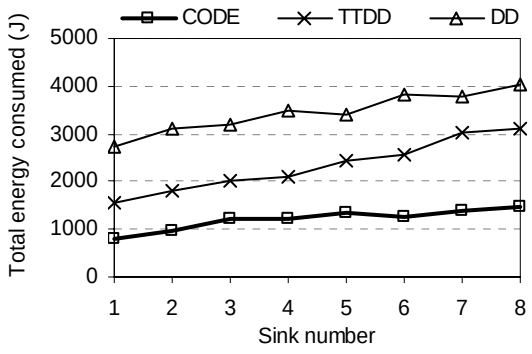



Fig.8.Energy consumption for different numbers of sinks

Fig.8 shows total energy consumption of CODE. It demonstrates that CODE is more energy efficient than other

protocols. This is because of two reasons. First, CODE uses QINT to efficiently aggregate query and data along data dissemination path. This path is established based on grid structure. Hence CODE can find a nearly straight route between a source and a sink. Second, CODE exploits GAF protocol, so that nodes in each grid negotiate among themselves to turn off its radio. Therefore, it reduces significantly energy consumption. In contrast, DD always propagates the new location of sinks throughout the sensor field in order for all sensor nodes to get the sink's location. In TTDD, the new multi-hop path between the sink and the grid is rebuilt. Also, data dissemination path of TTDD is along two sides of a right triangle.

Fig.9 demonstrates the average end-to-end delay of CODE. As shown in this figure, the delay of CODE is shorter than TTDD and slightly longer than DD. In Fig.10, it shows that the success rate of CODE is always above 90 percent. It means that CODE delivers most of data successfully to the multiple sinks.

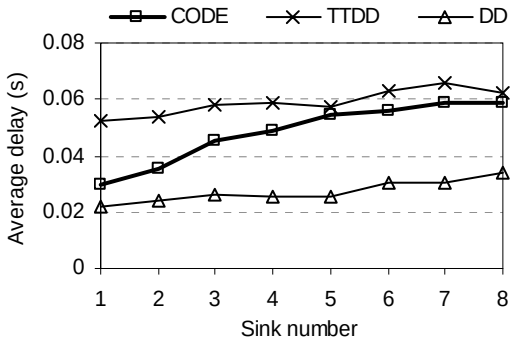



Fig.9.Delay for different numbers of sinks

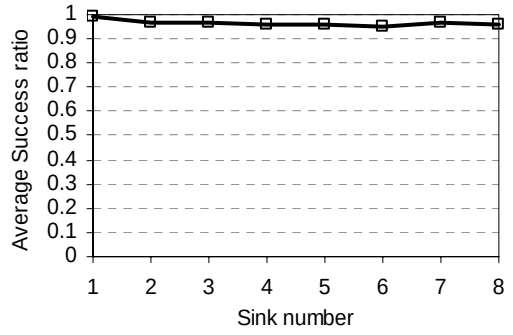



Fig.10.Success rate for different numbers of sinks

*b) Impact of Sink Mobility*

In order to examine the impact of sink mobility, we measure CODE for different sink speeds (0 to 30 m/sec). In this experiment, the network consists of 8 mobile sinks and 400 sensor nodes.

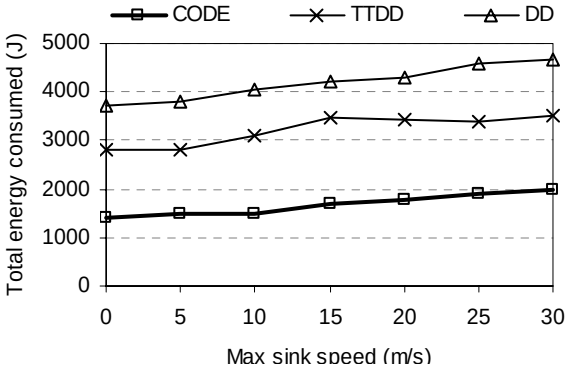



Fig.11.Energy consumption for different sink speeds

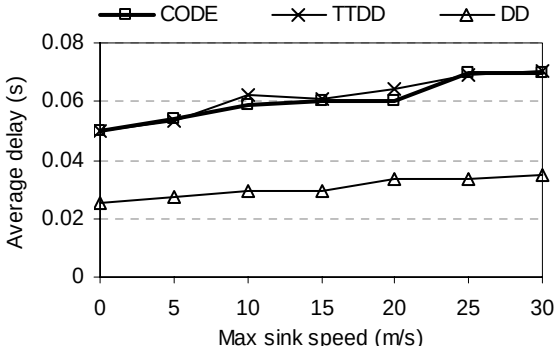



Fig.12.Delay for different sink speeds

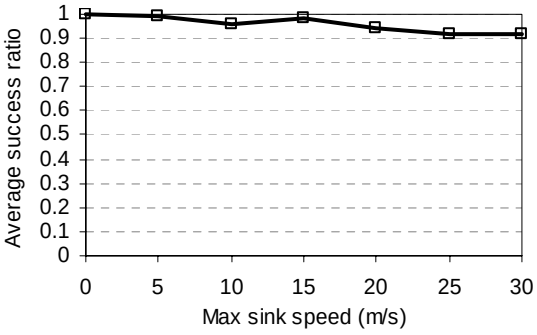



Fig.13.Success rate for different sink speeds

Fig.11 demonstrates total energy consumed as the sink speed changes. In both low and high speeds of the sinks, CODE shows the total energy consumed is better than other protocols, about twice less than TTDD and three times less than DD. The reason is that, aside from above explanations, the mobile sink contact with the coordinator to receive data while it is moving. Thus, the query only needs to resend as it

moves to another grid. Fig.12 shows the delay of CODE which is comparable with TTDD and longer than DD. In Fig.13, the success rate is also above 90 percent. These results show that CODE handles mobile sinks efficiently.

*c) Impact of Node Density*

To evaluate the impact of node density on CODE, we vary the number of nodes from 300 (1 node/cell on average) to 600 nodes (2 nodes/cell). Eight sinks move with speed 10m/sec as default. Fig.14 shows the energy consumption at different node densities. In this figure, CODE demonstrates better energy consumption than other protocols. As the number of nodes increase, the total energy consumption slightly increases. This is because of turning off node's radio most of the time. Therefore, energy is consumed mostly by the coordinators. While in TTDD and DD, nodes which don't participate in communication still consume energy in sleeping mode.

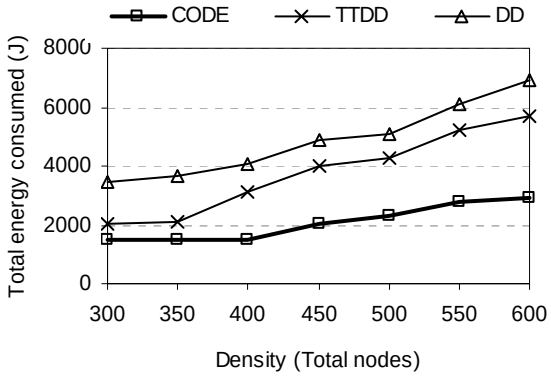



Fig.14.Energy consumption for different node density

*d) Network Lifetime*

In this experiment, the number of sinks is 8 moving with speed 10 m/sec. The number of sensor nodes is 400. A node is considered as a dead node if its energy is not enough to send or receive a packet. Fig.15 shows that number of nodes alive of CODE is about 60 percent higher than TTDD at the time 600sec. This is because of two reasons. The first is that CODE focus on energy efficiency. The second is that rotating coordinators distributes energy consumption to other nodes, thus nodes will not quickly deplete its energy like other approaches. TTDD concentrates on dissemination nodes to deliver data, therefore such nodes will run out of energy quickly. We do believe that when the node density is higher, the lifetime of CODE will be prolonged much more than other approaches.

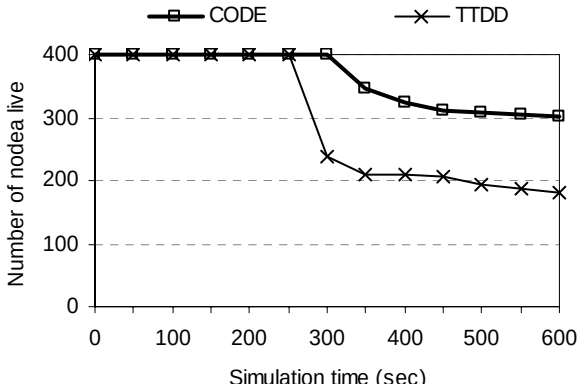



Fig.15.Number of node alive over time

## 3. Related Work

Many sensor network protocols have been developed in recent years. [3], [5]-[9], [12]. One of the earliest work, SPIN [5], addresses efficient dissemination of an individual sensor's observation to all the sensors in the network. SPIN uses meta-data negotiations to eliminate the transmission of redundant data. Directed Diffusion [3] and DRP [10] are similar in that they both take the data-centric naming approach to enable in-network data aggregation. In Directed Diffusion, all nodes are application-aware. This enables diffusion to achieve energy saving by selecting empirically good paths and by caching and processing data in-network. DRP exploits application-supplied data descriptions to control network routing and resource allocation in such a way as to enhance energy efficiency and scalability. GRAB [9] targets at robust data delivery in an extremely large sensor network made of highly unreliable nodes. It uses a forwarding mesh instead of a single path, where the mesh's width can be adjusted on the fly for each data packet. GEAR [6] uses energy aware neighbor selection to route a packet towards the target region. It uses Recursive Geographic Forwarding or Restricted Flooding algorithm to disseminate the packet inside the destination regions.

While such previous work only addresses the issue of delivering data to stationary sinks, other work such as TTDD [4], SEAD [7], and SAFE [8] target at efficient data dissemination to mobile sinks. TTDD exploits local flooding within a local cell of a grid which sources build proactively. Each source disseminates data along the nodes on the grid line to the sink. However, it does not optimize the path from the source to the sinks. When a source communicated with a sink, the restriction of grid structure may multiply the length of a strait-line path by $\sqrt{2}$ . Also, TTDD frequently renews the entire path to the sinks. It therefore increases energy consumption and the connection loss ratio. SAFE uses flooding that is geographically limited to forward the query to nodes along the direction of the source. SAFE uses

geographically limited flooding to find the gate connecting itself to the tree. Considering the large number of nodes in a sensor networks, the network-wide flooding may introduce considerable traffic. Another data dissemination protocol, SEAD, considers the distance and the packet traffic rate among nodes to create near-optimal dissemination trees. SEAD strikes a balance between end-to-end delay and power consumption that favors power savings over delay minimization. SEAD is therefore only useful for applications with less strict delay requirements.

CODE differs from such protocols in three fundamental ways. First, CODE exploits GAF protocol [11] to reduce energy consumption and data collision while the nodes make decision to fall into sleeping mode. Second, based on grid structure, CODE can control the number of transmitted hops and disseminates data along a path shorter than others such as TTDD. Third, the number of re-transmitted queries is reduced by maintaining an Agent to relay data to the sink when it moves within a grid. In addition, CODE takes into account of query and data aggregation [1], [2] to reduce the amount of data transmitted from multiple sensor nodes to sinks like other approaches.

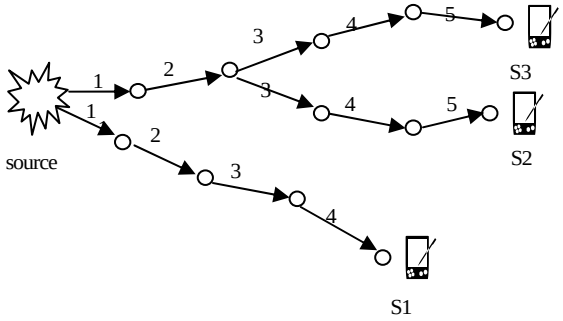
## IV. SIDE: A Sink Clustering- based Data Dissemination Protocol to Large Number of Sinks

In this section, we present SIDE, which addresses the sensor networks consisting of large number of sinks. This scheme is based on Geographic and Energy Aware Routing (GEAR) [6]. SIDE exploits capacities of not only nodes but also sinks in order to reduce communication cost. Receiving data, a sink can act as a source's *Agent* to relay data to the other nearby sinks. Since the sink is not as tight resource-constrained as sensor nodes, therefore they can talk directly to each others or through a few hops in order to ease the cost burden for sensor networks. SIDE is based on following assumptions:

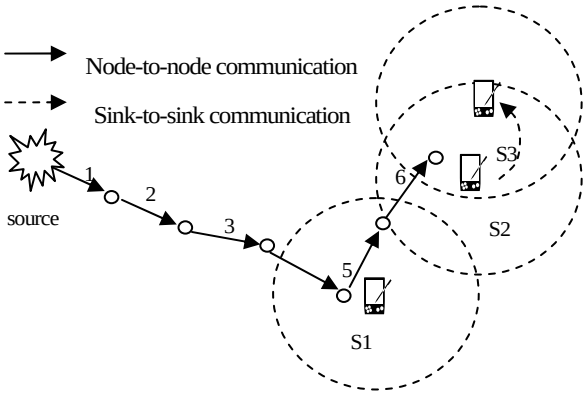
- After having been deployed, sensor nodes remain stationary in their initial locations.
- Each sensor node is assumed to be aware of its geographical location and residual energy.
- Sinks are immobile. The number of sinks may be from several to tens and they are not resources-constrained.
- When receiving a query, sensor nodes in the target region agree with each other so that only one node acts as the source to aggregate, pre-process and disseminate useful data to sinks.

### 1. SIDE Theory

Most current protocols use query and data aggregation to reduce communication cost while disseminating data to multiple sinks as illustrated in Fig.16a. In this case, a source propagates its data to multiple sinks along a reverse path or by simply flooding [3]-[5][31]. Each node is supposed to maintain some routing state so that whenever it receives data, it relays to an appropriate neighbor towards a sink. For example, Directed Diffusion [3] uses *interest* and *gradient* caching to establish a data dissemination path. SPIN [5] uses *negotiation* among one-hop neighbors to ensure that only useful information will be transmitted. In TTDD [4], each dissemination node is supposed to maintain a grid structure state to relay data. GPSR [29] is based on *greedy* and *perimeter forwarding* in a *planarized graph* to find next hop.




(a) Total Cost: 12, Maximum Delay: 5




(b) Total Cost: 6, Maximum Delay: 7

Fig.16.Two approaches of data dissemination to multiple sinks.

Although such protocols provide reliable and energy efficient data dissemination to sinks, yet they do not take account of sink capacities. Our idea, as illustrated in Fig.16b, is that instead of simultaneously disseminating data to all sinks, a source first sends data to the closest sink. If a sink can communicate with others, it can itself relay data directly to them instead of through multi-hop sensor nodes. For example in Fig.16b, receiving identical queries from the sinks S1, S2 and S3, the source first sends data to the closest sink S1. Then, S1 sends data to S2 via a few hops. S2 is notified that it can communicate with S3, so it directly

forwards data to S3. By doing this, it can reduce communication overhead and energy consumption of sensor nodes. However, this may cause longer delay. For example in Fig.16a (using query and data aggregation), we assume that the communication cost and delay between two neighboring nodes is of unit size. Therefore total cost to deliver data to three sinks is 12 and the maximum delay is 5 (delay of sending data to the sinks S2 or S3). In Fig.16b, which illustrates our idea, the total communication cost is 6, while maximum delay is 7 (delay of sending to S3). It's evident that this is not efficient for all cases of sink location (especially when sinks are far away from each other), thus we should provide multiple-option mechanisms for a source so that it knows when it should use the option of Fig1a or Fig.16b to achieve energy efficiency. Hereafter, we call such options as *Option 1* and *Option 2*, respectively.

**Definition 1**: In SIDE, we define two options to minimize energy consumed for data dissemination. *Option1* is defined as an approach which uses query and data aggregation. *Option2* is defined as a sink cluster-based approach, i.e. nearby sinks are grouped into cluster so that the source sends data to the closest sink first, then this sink is in charge of forwarding to the others by direct communication or via a few sensor node hops.

*A. How to Select the Minimum-Energy Option?*

In [1], C.Intanagonwinwat *et al* defined a task description as a list of attributed-value pairs that describes a task. In our approach, we also put some description of the sink which propagates the task. This description includes sink location and its communication range. It is necessary for a source to decide which option to use to disseminate data. For example, the animal tracking task mentioned in Section 1 might be described as:

```
type = four-legged animal
interval = 100 ms
duration = 20 seconds
dest = [-100,100,200,400] //destination region
sink_loc = [300,550]//sink location
sink_range = 250//sink communication range (m)
```

This named task description is called *interest*. An interest is injected into the network by sinks. Since sensor network interests may often contain the target geographical location, we use *Geography and Energy Aware* algorithm [6] to directly propagate interests to the target. A source relies on the interest communication cost to decide how to disseminate data. Before describing how a node decides data dissemination choice, we first mention briefly about the communication cost in GEAR [6].

Each node *N* is supposed to maintain a *learned cost h(N, R)* to destination *R*. If a node does not have $h(N_i, R)$ state for a neighbor $N_i$, it computes the *estimated cost* $c(N_i, R)$ as default value for $h(N_i, R)$. This estimated cost $c(N_i, R)$ of $N_i$, as described in [6], is defined as follows:

$$c(N_i, R) = \alpha d(N_i, R) + (1 - \alpha) e(N_i)$$

where $\alpha$ is a *tunable weight*, $d(N_i, R)$ is the distance from $N_i$ to the destination R, and $e(N_i)$ is the consumed energy at node $N_i$. A node picks up next-hop neighbour $N_{\min}$ which has the minimum cost compared with the other neighbors. Then, it sets its own $h(N, R)$ to $h(N_{\min}, R) + C(N, N_{\min})$ where the latter term is the cost of transmitting a packet from $N$ to $N_{\min}$. By computing this cost, a node can also recognize and avoid a *hole*.

We now describe how a source disseminates data to multiple sinks based on the communication cost. Let's assume that there are three sinks and one source as illustrated in Fig.16. We also assume that two sinks S2 and S3 are within the communication range of each other, so that they can directly talk to each other. Those sinks propagate identical interests to the source along different path according to GEAR algorithm. Each interest includes its communication cost from the sink to the source. Based on distances between sinks and source and the communication costs, we define an *average cost* $C_{avg}$ as follow:

$$C_{avg} = \frac{\sum_{i=1}^{n} cost(s_i)}{\sum_{i=1}^{n} dis(s_i)}$$

where *n* is the number of sinks, $cost(s_i)$ is total cost to deliver interest from sink $s_i$ to the source and $dis(s_i)$ is the geographical distance between sink $s_i$ and the source. For example in Fig.16, we assume that the distance between two neighboring hops is one unit, thus:

$$C_{avg} = \frac{\sum_{i=1}^{3} cost(s_i)}{\sum_{i=1}^{3} dis(s_i)} = \frac{cost(s_1) + cost(s_2) + cost(s_3)}{dis(s_1) + dis(s_2) + dis(s_3)}$$
$$= \frac{4 + 5 + 5}{4 + 5 + 5} = 1$$

A source has no information to compute the actual communication cost between two sinks, thus it computes the *estimated cost* between two sinks based on their position as:

$$e(s_i, s_j) = \begin{cases} 0 & ; if\ s_i\ and\ s_j\ can\ talk\ to\ each\ other \\ dis(s_i, s_j).C_{avg} & ; otherwise \end{cases}$$

where $dis(c_i, c_j)$ is distance between sink $s_i$ and $s_j$, which is computed based on sink locations in the received interests. For example

$$e(s_1, s_2) = dis(s_1, s_2).C_{avg} = 2.1 = 2;\ \ e(s_2, s_3) = 0.$$

We assume that S1 has the minimum communication cost of delivery the *interest* to the source. Therefore the source first sends data to S1. From S1, data is relayed to S2, S3, so on and so forth. We then define the *estimated cost* to disseminate data from the source to a sink $s_i$ as follow:

$$e(s_i) = cost(s_1) + \sum_{i=1}^{n} e(s_i, s_{i+1})$$

For example,

$e(s_1) = cost(s_1) = 4$ ,

$e(s_2) = cost(s_1) + e(s_1, s_2) = 4 + 2 = 6$ ,

$e(s_3) = cost(s_1) + e(s_1, s_2) + e(s_2, s_3) = 4 + 2 + 0 = 6$

Finally, the source comes to conclusion that:

$$C_1 = \sum_{i=1}^{n} cost(s_i);$$

$$C_2 = cost(s_1) + \sum_{i=1}^{n-1} e(s_i, s_{i+1});$$

$$if\ (C_1 > C_2)\ then\ Call\ Option2;$$

$$else\ Call\ Option1;$$

The algorithm is summarized as in Fig.17.

**SIDE-Algorithm**

1. **Source receives queries**
2. **Compute average cost**

$$C_{avg} = \frac{\sum_{i=1}^{n} cost(s_i)}{\sum_{i=1}^{n} dis(s_i)}$$

3. **Compute *estimated cost* between two sinks**

$$e(s_i, s_j) = \begin{cases} 0 & ;if\ s_i\ and\ s_j\ can\ talk\ to\ each\ other \\ dis(s_i, s_j).C_{avg} & ;otherwise \end{cases}$$

4. **Compute *estimated cost* from a source to a sink**

$$e(s_i) = cost(s_1) + \sum_{i=1}^{n} e(s_i, s_{i+1})$$

5. **Compute *total cost***

$$C_1 = \sum_{i=1}^{n} cost(s_i);$$

$$C_2 = cost(s_1) + \sum_{i=1}^{n-1} e(s_i, s_{i+1});$$

6. **Select which option to use**

$$if\ (C_1 > C_2)\ then\ Call\ Option2;$$

$$else\ Call\ Option1;$$

Fig.17.SIDE algorithm

For example in Fig.16a, we have:

$$C_1 = \sum_{i=1}^{3} cost(s_i) = 4 + 5 + 5 = 14$$

In Fig.16b, we have:

$$C_2 = cost(s_1) + \sum_{i=1}^{n-1} e(s_i, s_{i+1}) = 4 + 2 + 0 = 6$$

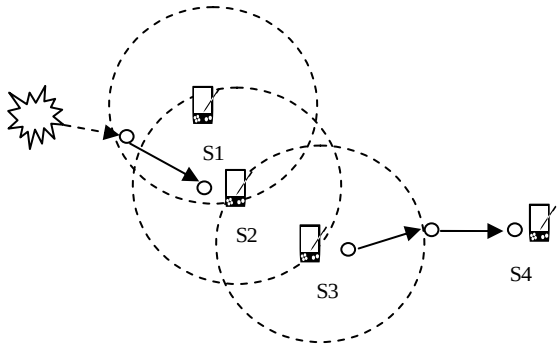
In $C_1$ , since queries from S2 and S3 are aggregated at some nodes on the path to the source, thus the source is notified that it has to subtract some cost from the total cost. As depicted in Fig.16, that subtracted cost is 2, therefore:

$$C_1 = 14 - 2 = 12$$

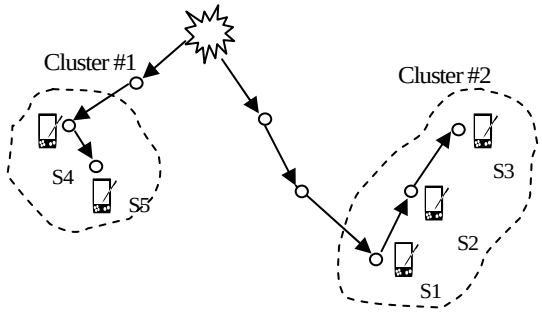
As the $C_1 > C_2$ , *Option2* is selected as the minimum-energy option to disseminate data to multiple sinks.

*B. Arbitrary Sink Position and Exceptional Cases*

So far, we have described the ideal case that all sinks are nearby. In fact, a sink is located at an arbitrary position in a sensor field. It's evident that there are several exceptional cases of sink position that we are going to mention.




(a) Case One: A sink can talk to several sinks




(b) Case Two: Two group of sinks are far way

Fig.18.Arbitrary Sink Position and Exceptional Cases

*a) Case One: A sink can talk to several sinks (Fig.18a)*

As depicted in Fig.18a, the sink S2 can talk directly to S1 and S3. S1 is the closest sink that the source should send data first. However, by comparing the communication range of S2, the source finds that S2 can talk directly to S1 and S3, thus it first sends to S2. Then the sink S2 is in charge of forwarding data to S1 and S3 directly. In turn, S3 relays data through multi-hop to S4. This is a trade-off between communication cost and delay. Though sending along the path source $\rightarrow S1 \rightarrow S2 \rightarrow S3$ may take less cost than sending along source $\rightarrow$ S2 $\rightarrow$ (S1,S3) , yet the delay of S3 to receive data is longer. Consequently, it also causes S4 to have a longer delay.

*b) Case Two: Groups of sinks are far away from each other (Fig.18b)*

This case often occurs in sensor networks wherein only some sinks are nearby, and far away from the others. The best solution of this case to minimize communication cost is that the source groups nearby sinks together. We define that a set of sinks $\{\Re\}$ is a group if they satisfy the following definition:

**Definition 2:** Given a couple of sinks $s_i, s_j \in \{\Re\}$. There exist a set of sinks $\{s_k\} \in \{\Re\}$ such that communication cost between $s_i, s_j$ via $\{s_k\}$ is less than a predetermined number of hops H.

*H* is decided according to node density, i.e. in a dense network *H* can be greater than in less dense network. For example in Fig.18b, three sinks S1, S2, and S3 are in the same group while S4 and S5 are in another group.

**Sink-Clustering-Algorithm**
**Input:** set of sinks $\Gamma$
**Output:** groups of nearby sinks $\{\Re\}$

1. $\Gamma = \{s_i, i = \overline{1,n}\}$ *;//set of sinks*
2. **while** $(\Gamma \neq \Phi)$ {*//loop until every sink is clustered*
3. $\Re = \{\Phi\}$ *;//initiating a group of nearby sinks*
4. **while** $(\Gamma \neq \Phi)$ {*//loop till all nearby sinks are clustered in group* $\Re$
5. $\Re \leftarrow s_k \mid s_k \in \Gamma$ *;//put an arbitrary sink into group*
6. $\Gamma \setminus \{s_k\}$; *//remove* $s_k$ *from set of sinks*
7. **for each** $s_i \in \Gamma$ {
8. **if** $(\exists s_j \in \Re \mid e(s_i, s_j) \leq H)$ {*// $s_i$ is nearby at least one sink in group* $\Re$
9. $\Re \leftarrow s_i$;
10. $\Gamma \setminus \{s_i\}$;

}}}}

Fig.19.Pseudo-code of Sink-Clustering-Algorithm

The *Sink-Clustering-Algorithm* is described in Fig.19. This algorithm is performed at the source-side. The main idea of this algorithm is that for a given group, we first pick up an arbitrary sink $s_k$ in the set of remaining sinks $\Gamma$ to put into a group $\Re$ (line 5), and remove $s_k$ from $\Gamma$ (line 6). Then, for each sink $s_i$ in the set $\Gamma$, we check if it is nearby to any sink of $\Re$ (line 8). If so, we put $s_i$ into $\Re$ (line 9) and remove it from $\Gamma$ (line 10). It's evident that this algorithm terminates, i.e. every sink is clustered, and no sink belongs to more than one group.

## 2. SIDE Simulation

*A. Simulation Model*

To evaluate SIDE performance, we simulate a sensor network which consists of *200* sensor nodes randomly deployed in a *2000mx2000m* field. The source generates different packets at an average interval of *5* second. Initially, the sink sends a query to the source. As the source receives a query, it calculates on query information to group sinks. Then it starts generating and sends data to the sinks. The simulation lasts for *500* seconds. The simulation result is depended on the scenario, i.e. the more near the sinks are, the less energy the network consumes. Therefore, we run our simulation on several scenarios with different position of sinks, and then we get the results averaged of all.

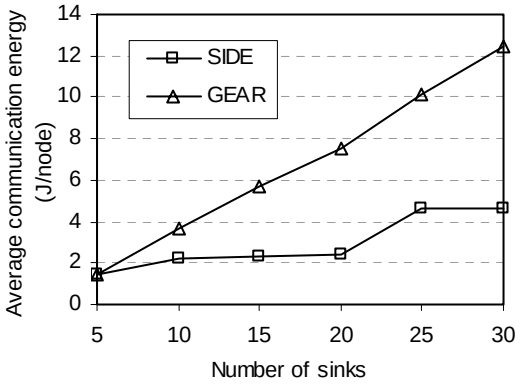



Fig.20.Energy consumption with different number of sinks

Fig.20 plots the energy consumed by SIDE. In comparison with GEAR, SIDE reduces the energy consumed significantly as the number of sinks increases. As the number of sinks is five, energy consumed by SIDE is almost the same as GEAR. This is because sinks are scatted far away from each other. Therefore, the source has to individually send data to sinks like GEAR. However, as the number of sinks increases, they can talk to others so energy consumed by sensor nodes is reduced. By comparing $C_1$ and $C_2$ in advance at the source-side to choose minimum-energy option (line 6 in Fig.19), the energy consumed is always

equal or less than GEAR. These results demonstrate that SIDE meets our design goal, i.e. reduces energy consumed of sensor nodes

Fig.21 shows the end-to-end delay of SIDE. This delay depends on sink's positions. However, in most cases, it is comparable with pure GEAR. This result is because of trade-off between communication cost and end-to-end delay as mentioned above. Fig.22 demonstrates the success ratio of SIDE. In this figure, it shows that the success rate of SIDE is always above 90 percent. It means that SIDE delivers most of data successfully to the multiple sinks.

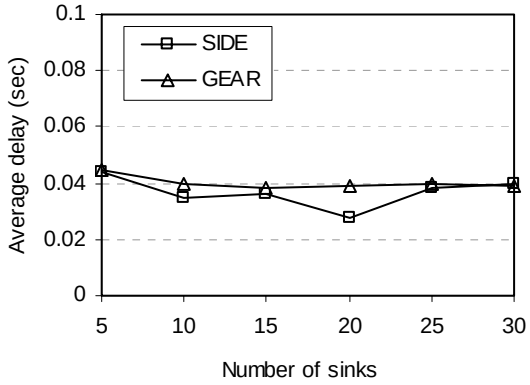



Fig.21.Average delay with different number of sinks

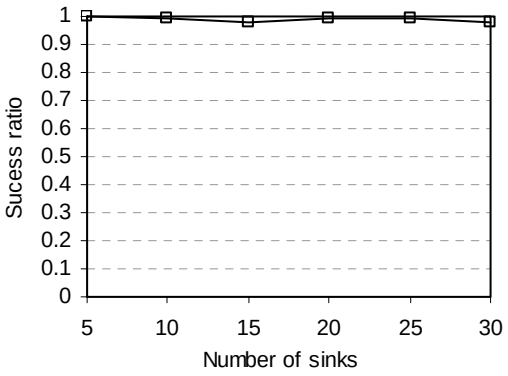



Fig.22.Success ratio with different number of sinks

### 3. Discussion

Recent advances in MEMS technology have result in large-scale sensor networks with hundreds or thousands of sensor nodes. Therefore, we believe that such networks will be dense, not only in terms of sensor nodes but also sinks in the near future. Exploiting sink capacities brings remarkable benefit to ease the cost burden for sensor networks due to the following reasons. First, as the size of sensor networks grows, the distance between the sink and sources become larger. Though in-network aggregation helps to reduce communication cost, it still needs source-to-sink data dissemination. SIDE groups nearby sinks so that the source reports data to the closest sink, then this sink acts as the source's Agent to disseminate data to the others. Second, the number of sinks may be large in future sensor networks. It is possible that some of them are often nearby with less resource-constrained than sensor nodes. They can talk directly to each other or via a few sensor nodes. Sharing out the communication cost to such sinks is worth to ease the cost burden of sensor networks. By selecting optional data forwarding approach at the source-side based on cost estimation, this approach also works well in sensor networks with low number of sinks. We do believe that, the approach brings an efficient scheme for sensor networks and can be employed in conjunction with other data dissemination protocols such as GPSR, Directed Diffusion, SPIN, etc.

## V. Conclusion

In wireless sensor networks where nodes have very limited power, the energy conservation is very important research issue. In this paper, we proposed two algorithms, named CODE and SIDE, to reduce energy consumed for sensor networks. CODE is deployed above GAF to take advantages of coordination protocols. In other words, CODE exploits coordination protocols to achieve energy efficiency while disseminating data successfully to mobile sinks. This approach is based on grid structure to find an energy-efficient data dissemination route between a source and mobile sinks. By negotiating with a coordinator, it maintains efficiently a route to mobile sinks through the coordinators while the other nodes turn its radio off to conserve energy. The other approach, SIDE, is based on GEAR. SIDE exploits capacities of sinks to ease the cost burden for sensor nodes. However, SIDE only copes with a large number of stationary sinks, rather than mobile sinks. Through our simulation results, we show that CODE and SIDE achieve energy efficiency and outperform other approaches such as Directed Diffusion, TTDD, and GEAR. However, it is evident that those networking models can exist, i.e. the number of mobiles sinks may be large in sensor networks. For our future work, we will combine CODE and SIDE to provide efficient data dissemination protocol for a large number of mobile sinks.

## References

[1] B. Krishnamachari, D. Estrin, and S. Wicker. "The impact of data aggregation in wireless sensor networks". *In Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops*, 2002.

[2] S. Maddes, R. Szewczyk, M. J. Franklin, and D. Culler. "Supporting aggregate queries over ad-hoc wireless sensor network". *In IEEE Workshop on Mobile Computing Systems and Applcations, May 2002.*

[3] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, F. Silva. "Directed diffusion for wireless sensor networking"

*Networking, IEEE/ACM Transactions* on Volume: 11 Issue: 1 , Feb. 2003 Page(s): 2 -16.

[4] Fan Ye, Haiyun Luo, Jerry Cheng, Songwu Lu, Lixia Zhang. "Sensor Networks: A two-tier data dissemination model for large-scale wireless sensor networks" *Proceedings of the Eighth Annual ACM/IEEE International Conference on Mobile Computing and Networks (MobiCOM 2002)*, Sept 2002, Atlanta, GA

[5] Joanna Kulik, Wendi Heinzelman, Hari Balakrishnan. "Negotiation-based protocols for disseminating information in wireless sensor networks" *ACM Transaction* on Volume 8 , Issue 2/3 March-May 2002.

[6] Yan Yu, Ramesh Govindan, Deborah Estrin. "Geographical and Energy Aware Routing: a recursive data dissemination protocol for wireless sensor networks (2001)" UCLA Computer Science Department Technical Report UCLA/CSD-TR-01-0023, May 2001.

[7] Hyung Seok Kim, Tarek F. Abdelzaher, Wook Hyun Kwon "Dissemination: Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks" *Proceedings of the first international conference on Embedded networked sensor systems*, November 2003

[8] Sooyeon Kim; Son, S.H.; Stankovic, J.A.; Shuoqi Li; Yanghee Choi; "SAFE: a data dissemination protocol for periodic updates in sensor networks" *Distributed Computing Systems Workshops*, 2003. Proceedings. 23rd International Conference on, 2003 Pages:228 – 234

[9] F. Ye, S. Lu, L Zhang. "GRAdient Broadcast: A Robust, Long-lived, Large Sensor Network" http://irl.cs.ucla.edu/papers/grab-tech-report.ps, 2001

[10] D. Co_n, D. V. Hook, S. McGarry, and S. Kolek. "Declarative ad-hoc sensor networking. SPIE Integrated" *Command Environments*, 2000.

[11] Y. Xu, J. Heidemannn, and D. Estrin. "Geography-informed energy conservation for ad hoc routing". *In Proc. of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001)*, Rome, Italy, July 2001.

[12] Wensheng Zhang; Guohong Cao; La Porta, T. "Data dissemination with ring-based index for wireless sensor networks" Network Protocols, 2003. *Proceedings. 11th IEEE International Conference* on , 4-7 Nov. 2003 Pages:305 – 314

[13] G. J. Pottie and W. J. Kaiser. "Embedding the internet: wireless integrated network sensors". *Communications of the ACM*, 43(5):51–58, May 2000.

[14] M. Stemm and R.H Katz. "Measuring and reducing energy consumption of network interfaces in hand-held devices". *IEICE Transaction and communication*, E80-B(8): 1125-1131, Aug. 1997

[15] Y.Xu, J.Hendemann, and D.Estrin. "Adaptive energy-conserving routing for multihop ad hoc networks". Technical Report TR-2000-527, USC/Information Sciences Institute, Oct. 2000. Available at ftp://ftp.isi.edu.isis-pubs/tr-527.pdf

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

[17] Wendi B. Heinzelman et al. "An Application-Specific Protocol Architecture for Wireless Microsensor Networks" *IEEE transactions on wireless communications*

[18] W. C. Y. Lee, "Mobile Cellular Telecommunications". McGraw Hill, 1995.

[19] S. Singh and C.S. Raghavendra. "Pamas: Power aware multi-access protocol with signalling for ad hoc networks". *ACM CCR*, July 1998.

[20] O. Kasten. "Energy consumption". ETH-Zurich, Swiss Federal Institute of Technology. Available at http://www.inf.ethz.ch/~kasten/research/bathtub/energy_consumption.html, 2001

[21] David B. Johnson and David A. Maltz. "Dynamic Source Routing in Ad Hoc Wireless Networks". *In Mobile Computing*, edited by Tomasz Imielinski and Hank Korth, chapter 5, pages 153–181. Kluwer Academic Publishers, 1996.

[22] Gang Chen et al "SENSE - Sensor Network Simulator and Emulator" http://www.cs.rpi.edu/~cheng3/sense/

[23] US Naval Observatory (USNO) GPS Operations, http://tycho.usno.navy.mil/gps.html

[24] J.Albowitz, A.Chen, and L.Shang, "Recursive Position Estimation in Sensor Networks". *ICNP'01*, 2001.

[25] The Network Simulator ns-http://www.isi.edu/nsnam/ns/

[26] I.F. Akyidiz et al, "Wireless Sensor Network: A survey", *computer networks*, Vol3.38, pp.393-422, March 2002

[27] S. Tilak et al., "A Taxonomy of Wireless Microsensor Network Models" *in ACM Mobile Computing and Communications Review (MC2R)*, June 2002.

[28] K. Akkaya, M. Younis "A Survey on Routing Protocols for Wireless Sensor Networks" (To appear in) Elsevier Ad Hoc Network Journal

[29] Brad Karp and H. T. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In Proc. ACM Mobicom, Boston, MA, 2000

[30] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S.J. Pister, "System architecture directions for networked sensors," in *Proc. Int. Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2000, pp. 93–104.

[31] L.X. Hung, SY Lee, "Minimum-Energy Data Dissemination in Coordination-based Sensor Networks" Submitted to *The International Conference on Information Networking 2005* (ICOIN), Korea

[32] L.X. Hung, SY Lee, "Sink Clustering-based Data Dissemination for Wireless Sensor Networks", Submitted to the *IEEE First International Workshop on Sensor Networks and Systems for Pervasive Computing (PerSeNS 2005)*