A Simple Sink Mobility Support Algorithm for Routing Protocols in Wireless Sensor Networks

Chun-Su Park, You-Sun Kim, Kwang-Wook Lee, Seung-Kyun Kim, and Sung-Jea Ko

Department of Electronics Engineering, Korea University,
Anam-Dong Sungbuk-Ku, Seoul, Korea
{cspark, yskim, kwlee, skkim, sjko}@dali.korea.ac.kr

Abstract. In order to support the sink mobility of conventional routing protocols, we propose a simple route maintaining algorithm which does not use the flooding method. In the proposed method, when the sink loses the connection with the source, it does not rebuild an entire route but simply repairs the existing route based on local information. Experimental results show that the proposed algorithm drastically improves the conventional routing protocols in terms of both energy and delay in case of mobile sink.

1 Introduction

Energy is the most crucial resource in the wireless microsensor networks due to the difficulty of recharging batteries of thousands of devices in remote or hostile environments. When a sink is mobile, the energy is consumed for building new packet forwarding route, disseminating data, and maintaining linkage between source and sink. The more frequently the sink moves, the more energy is consumed to maintain the linkage between the source and the sink. Some routing protocols have been recently proposed to support the mobile sink [1], [2], [3], [4]. Instead of flooding query packets, Scalable Energy-efficient Asynchronous Dissemination protocol (SEAD) constructs and maintains a data dissemination tree from source to multiple mobile sink [1]. A sink that wants to join the tree registers itself with the closest access node. When the sink moves out of range of the access node, the route is extended through the inclusion of a new access node. A Two-Tier Data Dissemination (TTDD) was proposed to provide scalable and efficient data delivery to mobile sinks [2]. Upon detecting a stimulus, each source node proactively builds a grid structure which enables a mobile sink to receive data continuously while moving by flooding queries within its local cell only. These protocols have some defects in their own assumptions and network model. For example, each sensor node is assumed to be aware of its own geographic location and mobile sensor nodes are not allowed. Moreover, SEAD and TTDD constrain the method of building the data forwarding route in order to support mobile sink. These constraints make them very difficult to be adopted in the other routing protocols for sink mobility.

In this paper, in order to support the sink mobility of the conventional routing protocols, we propose a simple sink mobility support (SMS) algorithm which does not use the flooding method. The proposed algorithm can be easily adopted in most existing routing protocols since it does not need to know the geometric location of sensor nodes. Moreover, the proposed algorithm incurs very few communication overhead. The rest of this paper is organized as follows. Section 2 presents network model and the proposed algorithm. Experimental results are presented in Section 3.

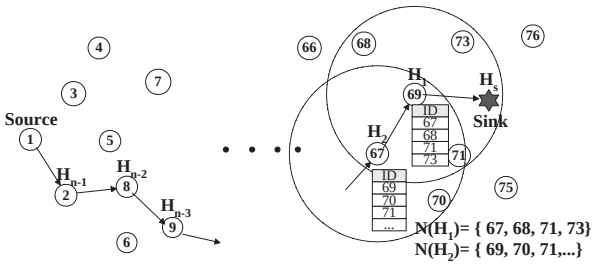


Fig. 1. Examples of H_k and $N(H_k)$.

2 Network Model and SMS

2.1 Network Model

For the SMS algorithm, we adopt the following assumptions: First, there is multihop data transmission between source and sink. Second, each sensor has a limited battery energy. Third, the speed of the mobile sink is limited. And last, the sensor network has a sufficient number of sensor nodes. For the sake of explanation, we limit our consideration to the case of a single sink. The proposed method can be easily extended to the multiple sink. The target microsensor network model is represented by a set U consisting of scattered sensor nodes. Let $R = \{H_1, H_2, ..., H_k, ..., H_{n-1}, H_n\} \subset U$ be the set of the sensor nodes existing along the data forwarding path from source to sink, where k represents the node distance from the sink on a hop scale. For example, in Fig. 1, $k = 1$ at *Node 69* and $k = 2$ at *Node 67*. Two nodes are said to be *neighbor* if they can directly communicate with each other within a single hop. Whenever a node first receives any packet from its neighboring node, it registers the ID of the neighboring node in its own *neighbor table*. However, if a node does not receives any response from a certain neighboring node during a fixed time, it removes the ID of the neighboring node from its own *neighbor table*. Let $N(H_k)$ be the set of nodes in the *neighbor table* of H_k. Fig. 1 shows $N(H_1) = \{67, 68, 71, 73\}$ and $N(H_2) = \{69, 70, 71, ...\}$. Each node in R generates the description table of

a current sensing task in which it participate [5], [6]. In most routing protocols, the sensor node relays received data packets to its own downstream node using the *task description table* that contains source ID, sink ID, data type, its own downstream node ID, and so on. If there are more than one sink and source, each data forwarding path can be distinguished using the *task description table*.

2.2 The SMS

The SMS consists of three phases; preliminary investigation, node selection, and route correction.

Preliminary Investigation Before a sink moves out of the radio range of H_1, it must store neighbor tables of some H_k's close to the sink. Let $\Omega = \{N(H_1), N(H_2), ...\}$ be a set of neighbor tables to be stored in the sink. The Ω is used for future route correction. Note that the neighbor tables of H_1 and H_2 are sufficient to support mobile sinks such as human being, robots, and tanks. Thus, we focus on the case of $\Omega = \{N(H_1), N(H_2)\}$. For example, in Fig. 1, the sink gathers the neighbor tables from *Node 69* and *Node 67*.

Node Selection Next, we describe how to maintain connectivity between source and mobile sink. When the sink finds out that it loses the connection with H_1 due to its movement, it broadcasts the *Get_Near_Node* message containing the task description table to search sensor nodes nearby. All nodes that received the *Get_Near_Node* message send a response message to the sink. Then, the sink generates or updates the neighbor table $N(\widetilde{H}_s)$ consisting of the ID's of the nodes which have transmitted response messages. Among $N(\widetilde{H}_s)$, the sink chooses a new H_1, $\widetilde{H}_1$, that has the smallest hop distance to the source by using $N(\widetilde{H}_s)$ and Ω. Let the number of elements in set G be $n[G]$. Then, $\widetilde{H}_1$ can be selected using the following procedure:

(a) If $n[R \cap N(\widetilde{H}_s)] = 1$, then $R \cap N(\widetilde{H}_s) = \{H_{k_1}\}$ and H_{k_1} becomes $\widetilde{H}_1$ for the sink. If $n[R \cap N(\widetilde{H}_s)] = m$, with $m \geq 2$, $R \cap N(\widetilde{H}_s) = \{H_{k_1}, H_{k_2}, ..., H_{k_m}\}$. Then, among H_{k_i}'s, the one that has the smallest hop distance to the source becomes $\widetilde{H}_1$. For example, if $\{H_2, H_3\} \subset R \cap N(\widetilde{H}_s)$, the sink selects H_3 as $\widetilde{H}_1$. If $R \cap N(\widetilde{H}_s) = \phi$, go to (b).

(b) If $n[N(H_2) \cap N(\widetilde{H}_s)] = 1$, then $N(H_2) \cap N(\widetilde{H}_s) = \{H_{k_1}\}$ and H_{k_1} becomes $\widetilde{H}_1$ for the sink. If $n[N(H_2) \cap N(\widetilde{H}_s)] = m$, with $m \geq 2$, then $N(H_2) \cap N(\widetilde{H}_s) = \{H_{k_1}, H_{k_2}, ..., H_{k_m}\}$. In the case, the nodes receiving *Get_Near_Node* message send a response messages containing the information on their own remaining energy to the sink. Then, among H_{k_i}'s, the one that has the largest energy becomes $\widetilde{H}_1$. If $N(H_2) \bigcap N(\widetilde{H}_s) = \phi$, go to (c).

(c) In the same manner as (b), the sink selects $\widetilde{H}_1$ among $N(H_1) \bigcap N(\widetilde{H}_s)$. And if $N(H_1) \bigcap N(\widetilde{H}_s) = \phi$, go to (d).

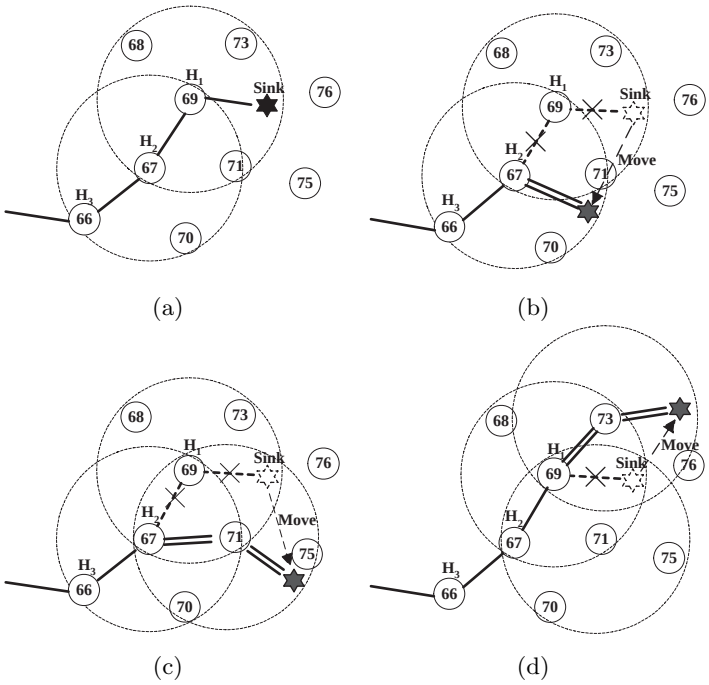


Fig. 2. Types of route correction. In these figures, solid, dot, and double solid lines represent the existing, broken, and newly created paths, respectively. (a) Initial microsensor network. (b) Type 1. (c) Type 2. (d) Type 3.

(d) In this case, there is no candidate for $\widetilde{H}_1$, i.e., $R \cap N(\widetilde{H}_s) = N(H_2) \cap N(\widetilde{H}_s) = N(H_1) \bigcap N(\widetilde{H}_s) = \phi$. Thus, the sink must set up a new route toward the source. The rebuilding method is the same algorithm that the original routing protocol adopted in the wireless sensor network follows. This case happens when the sink enters the empty regions, moves too fast, or a large number of nodes run out of their own batteries. But, in general, this case seldom happens and can be reduced or eliminated by expanding the range of *neighbor table* from one hop to two hops or more.

Route Correction The last phase of the SMS corrects the broken route caused by the sink movement. There are 3 types of route correction.

Type 1: If $\widetilde{H}_1 \in R$, the sink transmits the *Route_Update* message to $\widetilde{H}_1$. Then, $\widetilde{H}_1$ updates its data forwarding path from its downstream node to the sink (see Fig. 2(b)).

Type 2: If $\widetilde{H}_1 \in N(H_2)$, the sink must transmit to $\widetilde{H}_1$ the *Route_Update* message including the address of $\widetilde{H}_2$ since $\widetilde{H}_1$ does not know the address of its upstream node $\widetilde{H}_2$ from which $\widetilde{H}_1$ will receive data packets. $\widetilde{H}_1$ relays the *Route_Update* message to $\widetilde{H}_2$. Then $\widetilde{H}_1$ and $\widetilde{H}_2$, respectively, correct their own data forwarding paths to the sink and $\widetilde{H}_1$ (see Fig. 2(c)).

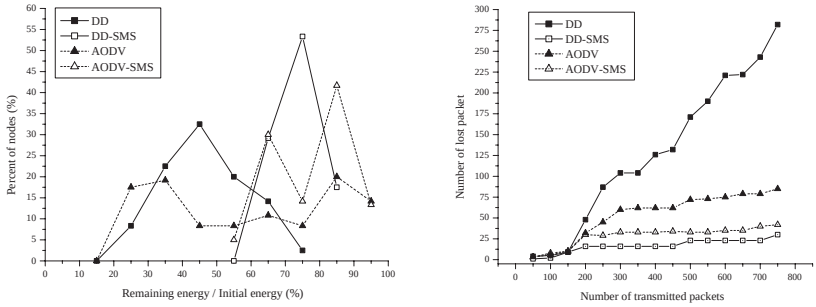
Type 3: If $\widetilde{H}_1 \in N(H_1)$, $\widetilde{H}_1$ relays the *Route_Update* message from the sink to $\widetilde{H}_2$ in the same manner as Type 2 (see Fig. 2(d)).

In route correction phase, the sink sends the *Route_Update* message to the $\widetilde{H}_1$ in order to correct the old route. If $\widetilde{H}_1$ and $\widetilde{H}_2$ send to the sink the response messages including their own neighbor tables, the future preliminary investigation phase for the next movement of the sink can be conducted in current route correction phase, simultaneously.

3 Experimental Result

The performance of the SMS is evaluated using the NS-2 simulator [7]. Our simulation uses the power consumption model that requires 0.660W for transmitting and 0.395W for receiving and 0.035W for idle. The target wireless microsensor network consists of 120 sensor nodes in a 1000m $\times$ 1000m field. The transceiver has a 150m radio range and the energy consumption is measured in terms of Joules/node. In our experiment, a single mobile sink is moving at 10 m/sec, i.e., the fastest human speed and the simulation time is 1000 sec. Since the repaired route may not be a globally optimized one, the sink does not perform route correction but rebuilds the entire route whenever the sink has moved thirty times.

In our experiment, we combined the most famous routing protocols, Ad Hoc On Demand Distance Vector (AODV) and Direct Diffusion (DD), with the proposed SMS. These two upgraded versions are named DD-SMS and AODV-SMS. Fig. 3(a) is a graph showing the distribution of the remaining energy for each protocol. In DD, the period for interest packets is set to 5 sec. Since the DD performs flooding to make a new routing table, the remaining energy of all nodes is small and its variance is relatively even over the whole network. As shown in Fig. 3(a), the remaining energy of DD is distributed within a band between 15% $\sim$ 75%. In case of AODV, the sink broadcasts query packets throughout the network to rebuild an entire route. In addition, unlike DD, AODV uses *Hello packet* to search neighboring nodes, which causes additional energy consumption. Thus, AODV is less efficient than DD in respect of energy consumption. In Fig. 3(a), the measured remaining energy of AODV is between 15% $\sim$ 95%. Since DD-SMS and AODV-SMS do not use the flooding method and route rebuilding is performed only inside the limited local region, they are more energy efficient than the original DD and AODV protocols. In our experiment, the measured remaining energy of DD-SMS is distributed within a narrow band between 55% $\sim$ 85%, and that of AODV-SMS is distributed between 55% $\sim$ 95%. The average remaining energy of DD-SMS and AODV-SMS is improved about 40% as compared with that of DD and AODV.



(a) (b)

Fig. 3. (a) Remaining energy. (b) Packet delivery.

We also compare the number of lost packets of DD and AODV with that of DD-SMS and AODV-SMS. Fig. 3(b) shows the number of lost packets versus the number of transmitted packets. DD performs worst among the above mentioned four routing protocols in all cases. This is because a new route is made after the sink floods interest packets at the fixed time, i.e., the disconnection time of DD is longer than that of any other protocol. Fig. 3(b) shows that the combination of DD with SMS (DD-SMS) can significantly improve the performance of the DD. By adopting proposed SMS, the packet loss ratio of the AODV and DD is decreased by about 47% and 85%, respectively.

References

1. H. S. Kim, T. Abdelzaher, and W. H. Kwon,: Minimum-Energy Asynchronous Dissemination to Mobile Sinks in Wireless Sensor Networks, ACM Conference on Embedded Networked Sensor Systems (2003) 193–204
2. Haiyun Luo, Fan Ye, Jerry Cheng, Songwu Lu, and Lixia Zhang,: TTDD: Two-tier Data Dissemination in Large-scale Wireless Sensor Networks, ACM/Kluwer Mobile Networks and Applications (MONET), Special Issue on ACM MOBICOM (2003) 161–175
3. L. Song and D. Hatzinakos,: Dense Wireless Sensor Networks with Mobile Sinks, Acoustics, Speech, and Signal Processing, Proceeding (2005) vol. 3, 677–680
4. Wang, Z.M. Basagni, S. Melachrinoudis, E. Petrioli, C.,: Exploiting Sink Mobility for Maximizing Sensor Networks Lifetime, 38th Annual Hawaii International Conference on (2005) 287a
5. C. E. Perkins, E. M. Belding-Royer and S. Das,: Ad Hoc On Demand Distance Vector (AODV) routing, IETF Internet draft, (2002) 38
6. C. Intanagonwiwat, R. Govindan and D. Estrin,: Directed diffusion: A scalable and robust communication paradigm for sensor networks, Proc. ACM MOBICOM Conf. Boston, Massachusetts (2000) 56–67
7. http://www.isi.edu/nsnam.