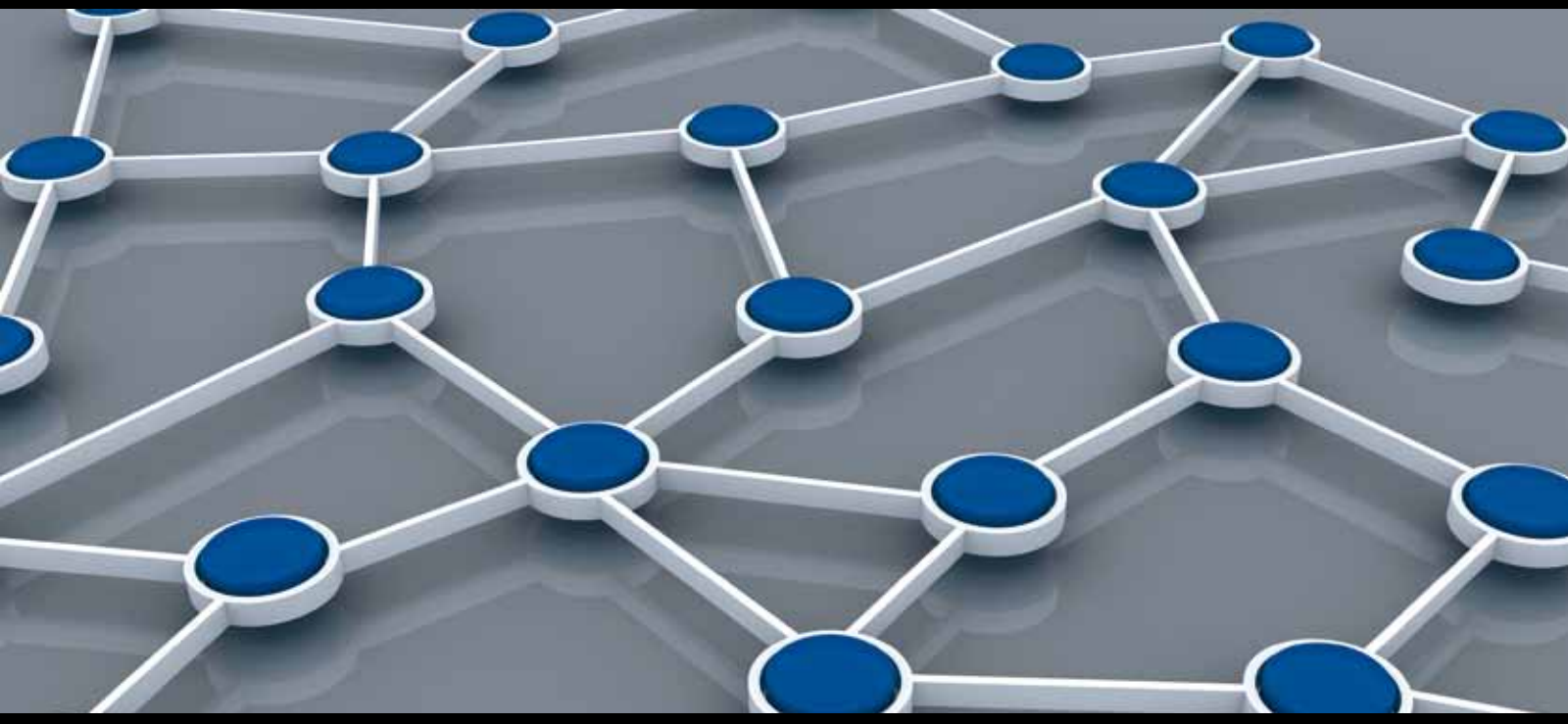
International Journal of Distributed Sensor Networks

Advanced Issues of Operating Systems for Reliable Distributed Sensor Networks: Aim and Scope

Guest Editors: Jiman Hong, Tei-Wei Kuo, and Junyoung Heo



Advanced Issues of Operating Systems for Reliable Distributed Sensor Networks: Aim and Scope

International Journal of Distributed Sensor Networks

Advanced Issues of Operating Systems for Reliable Distributed Sensor Networks: Aim and Scope

Guest Editors: Jiman Hong, Tei-Wei Kuo, and Junyoung Heo

Editorial Board

Contents

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 394197, 2 pages
doi:10.1155/2012/394197

Editorial

Advanced Issues of Operating Systems for Reliable Distributed Sensor Networks: Aim and Scope

Jiman Hong,[1] Tei-Wei Kuo,[2] and Junyoung Heo[3]

[1] *School of Computer Science and Engineering, Soongsil University, Seoul 156-743, Republic of Korea*
[2] *Department of Computer Science and Information Engineering, National Taiwan University, Taipei 10617, Taiwan*
[3] *Department of Computer Engineering, Hansung University, Seoul 136-792, Republic of Korea*

Correspondence should be addressed to Junyoung Heo, jyheo@hansung.ac.kr

Received 25 December 2011; Accepted 25 December 2011

The designs of high-reliabity operating systems for distributed sensor networks must take into account a wide variety of constraints: fault tolerance, performance, code size, presence of real-time tasks, maintainability, and possibly scalability. This special issue provides a forum for the presentation of high-quality, original research covering all aspects of operating systems to provide high reliability for distributed sensor networks; algorithm, design, analysis, implementation, evaluation, and case studies. Solutions might be proposed at different levels of abstractions, making use of an assortment of tools and methodologies.

Flash memory is a widely used device to store data in distributed sensor nodes. Sensor nodes are highly resource constrained in terms of limited processing speed, runtime memory, persistent storage, communication bandwidth, and finite energy. Therefore, for wireless sensor networks supporting sense, store, merge, and send schemes, an energy-efficient and reliable database-based query optimization technique is highly required with consideration of sensor node constraints. Databases on hard disk drives perform data storage and retrieval using index structures, which are still not practiced for sensor devices. In the first paper, "*Performance evaluation of indices-based query optimization from flash-based data centric sensor devices in wireless sensor networks,*" authors evaluate different indices like B-tree, R-tree, and MR-tree by implementing them on log structured external NAND flash-memory-based advanced file systems for supporting energy-efficient data storage and query optimization from flash-based data centric sensor devices in wireless sensor networks. Experimental results show that PIYAS file system along with B-tree indexing deployed on flash memory MLC gives the significant performance in respect of high query throughput optimization and less resources consumption for wireless sensor devices.

The IEEE 802.15.4 standard provides GTS (Guaranteed Time Slot) mechanism for reliable transmission. GTS mechanism is suitable for transmitting time-sensitive data because it allocates time slots to a specific node. However, the GTS mechanism in the standard can be used only for one-hop communication. In the second paper, "*A novel GTS mechanism for reliable multihop transmission in the IEEE 802.15.4 network,*" authors propose and implement a multihop GTS mechanism for reliable transmission in multihop networks. Simulation results using NS-2 show that low delay and high delivery ratio can be achieved using the proposed mechanism.

Mobility in mobile sensor networks causes frequent route breaks and each routing scheme reacts differently during route breaks. It results in a performance degradation of the energy consumption to reestablish the route. Since routing schemes have various operational characteristics for rerouting, the impact of mobility on routing energy consumption shows significantly different results under varying network dynamics. Therefore, we should consider the mobility impact when analyzing the routing energy consumption in mobile sensor networks. However, most analysis of the routing energy consumption concentrates on the traffic condition and often neglects the mobility impact. The third paper, "*Impact of mobility on routing energy consumption in mobile sensor networks,*" analyzes the mobility

impact on the routing energy consumption by deriving the expected energy consumption of reactive, proactive and flooding scheme as a function of both the packet arrival rate and topology change rate. Routing energy consumption for mobile sensor networks is analytically shown to have a strong relationship with sensor mobility and traffic conditions. Authors demonstrate the accuracy of our analysis through simulations. The analysis can be used to decide a routing scheme that will operate most energy-efficiently for a sensor application, taking into account the mobility as well as traffic condition.

In urban vehicular sensor networks, vehicles equipped with onboard sensors monitor some area and the result can be shared to neighbor vehicles to correct their own sensing data. However, due to the frequent change of vehicle topology compared to the wireless sensor network, it is required for a vehicle to discover neighboring vehicles. Therefore, efficient neighbor discovery algorithm should be designed for vehicular sensor networks. In the fourth paper, "*Tree-based neighbor discovery in urban vehicular sensor networks,*" authors propose two efficient tree-based neighbor discovery algorithms in vehicular sensor networks and analyze them. The expected value of neighbor discovery delay has different characteristics depending on neighbor discovery algorithms. An interesting observation of the result is M-binary tree-based neighbor discovery shows better performance than M-ary tree-based neighbor discovery in the parking lot scenario, which is a counterintuitive result.

The fifth paper, "*A fast and reliable hybrid data delivery protocol for large-scale heterogeneous sensor networks,*" proposes a hybrid data delivery method for the large-scale heterogeneous sensor networks, which is a fast and reliable delivery protocol for the aggregated data from the sinks to the GW. This paper develops a new multicriteria-ranking algorithm, which determines multiple forwarders for each hop by ranking neighbor nodes. To rank the nodes, they compute the fitness value using features for each node such as the received signal strength, nodal delay, and hop distance. This paper determines the time of sending among forwarders using the waiting time assignment algorithm. In the experimental section, the proposed method outperforms conventional data delivery protocols in terms of data delivery ratio and end-to-end delay.

Acknowledgments

The guest editors would like to acknowledge the contributions of the many experts who submitted their work, participated in the review process, and provided constructive and helpful comments to the authors to improve the technical content and presentation quality of the papers. They would also like to thank Professor Sundaraja Sitharama Iyengar, the Editor-in-Chief of the Hindawi International Journal of Distributed Sensor Networks, for their support and help in bringing this special issue to press.

Jiman Hong
Tei-Wei Kuo
Junyoung Heo

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 430439, 12 pages
doi:10.1155/2012/430439

Research Article

Impact of Mobility on Routing Energy Consumption in Mobile Sensor Networks

Jinman Jung,[1] Yookun Cho,[1] and Jiman Hong[2]

[1] *School of Computer Science and Engineering, Seoul National University, Seoul 151-744, Republic of Korea*
[2] *School of Computing, Soongsil University, Seoul 156-743, Republic of Korea*

Correspondence should be addressed to Jiman Hong, jiman@ssu.ac.kr

Received 1 September 2011; Accepted 10 November 2011

Academic Editor: Junyoung Heo

Mobility in mobile sensor networks causes frequent route breaks, and each routing scheme reacts differently during route breaks. It results in a performance degradation of the energy consumption to reestablish the route. Since routing schemes have various operational characteristics for rerouting, the impact of mobility on routing energy consumption shows significantly different results under varying network dynamics. Therefore, we should consider the mobility impact when analyzing the routing energy consumption in mobile sensor networks. However, most analysis of the routing energy consumption concentrates on the traffic condition and often neglects the mobility impact. We analyze the mobility impact on the routing energy consumption by deriving the expected energy consumption of reactive, proactive, and flooding scheme as a function of both the packet arrival rate and topology change rate. Routing energy consumption for mobile sensor networks is analytically shown to have a strong relationship with sensor mobility and traffic conditions. We then demonstrate the accuracy of our analysis through simulations. Our analysis can be used to decide a routing scheme that will operate most energy efficiently for a sensor application, taking into account the mobility as well as traffic condition.

1. Introduction

Mobile sensor networks are dynamic networks formed by a set of mobile sensor nodes and sink node connected through wireless links. These sensor nodes sense a data in application domains (ranging from wildlife monitoring to vehicle tracking) and then transmit the sensed data to the sink node through wireless multihop routing. Sensor nodes have processing and communication capacities, whose main tasks include controlling sensors, processing sensed data, and transmitting the collected data to a sink node. A typical sensor node has a low-power CPU, tiny memory (RAM/ROM), R/F module, many kinds of sensing units, and constrained battery power. For example, Berkeley's MICA motes only have an 8-bit CPU, 4 KB RAM, and only two AA alkaline batteries. The most energy-consuming component is the R/F module, which provides wireless communications. The energy consumption when transmitting 1 bit of data on a wireless channel is similar to thousands of cycles of CPU instructions [1]. Thus, the energy efficiency of the routing schemes for wireless sensor networks largely affects the energy consumption and network lifetime of wireless sensor networks [2].

In recent years, many routing schemes have been proposed. Typically, there are two main categories of routing schemes, proactive schemes (e.g., DSDV [3], SPIN [4]), and reactive schemes (e.g., AODV [5], DSR [6], and MOR [7]). In the proactive schemes, each node periodically sends control packets to the network in order to maintain a routing table. When the network topology changes, the nodes propagate control messages throughout the network in order to update fresh entries in the routing table regardless of the data packets' arrivals. In the reactive schemes, each node sends control packets for route discovery to find the path to the destination only if they are needed, on demand. The reactive scheme can use energy more efficiently than the proactive scheme does for higher mobility. This is because only if a source wants to send to a destination, it invokes the route discovery mechanisms. Meanwhile, the energy consumption of the reactive scheme can increase dramatically under heavy

traffic load. If the routing information become frequently inaccurate or stale during the packet transmission, the flooding scheme [8, 9] as a data transfer method can be used.

These various behaviors of the routing schemes according to the mobility and traffic load cause a different pattern in the energy consumption. Hence, knowledge of the characteristics of the network environment such as the mobility and traffic load is necessary when selecting a suitable routing scheme for a specific sensor network application. Fortunately, most sensor networks have some homogeneous characteristics. In a homogeneous network, each node periodically sends its readings to a sink node with the same traffic load in terms of the packet arrival rate, which is the number of sent packets to a sink node per unit time. Additionally, mobility can result from the same network environmental influences (e.g., wind, water, etc.) or the same mobile object (e.g., human, vehicles, etc.) by which the sensors may be carried. Through this assumption, we can approximately estimate the expected mobility variables or know the traffic rate of the sensor networks.

In this paper, we investigate the energy consumption of proactive, reactive, and flooding schemes for various node mobility and traffic loads in terms of the topology change rate and packet arrival rate, respectively. Through analytical evaluation of the energy consumption, we propose a decision mechanism of the routing scheme that will operate most energy efficiently for a sensor application. Our approach is to minimize the energy consumption by modeling the expected energy demands of proactive and reactive schemes as well as flooding scheme. We present the analytical tool needed to compare the energy consumption among the routing schemes.

The remainder of this paper is organized as follows. In Section 2, we present recent works in the modeling of routing schemes for the mobile sensor networks. In Section 3, the system model is described. In Section 4, we derive the expected energy consumption of the routing schemes (i.e., proactive, reactive, and flooding schemes). In Section 5, we analyze the mobility impact on routing energy consumption by comparing proactive, reactive, and flooding schemes. In Section 6, we evaluate the performance of our energy consumption model using a simulation. Finally, additional conclusions are drawn, and potential directions for future work are given in Section 7.

2. Related Works

Several studies on analytical approaches have been proposed for routing schemes [10–15].

Gao [10] presented an analytical approach that showed the characterization of the energy consumption for a sensor network application. However, this work does not cover different characteristics on the performance of the routing protocol.

Yang et al. [11] proposed an analytical model that would ensure the optimal periodic route maintenance for proactive schemes. The authors categorized the proactive protocols based on the periodic route and link maintenance operations done. They focused on the proactive scheme without a comparison of the proactive and reactive schemes, and their model is different from our study in focus of the analysis.

Zhou et al. [12] proposed a mathematical and simulative framework for quantifying an overhead of reactive scheme. They presented a simplified model of OLSR and AODV protocols and studied the scalability of the reactive scheme. However, this work was specific for reactive scheme operating under certain conditions.

Lebedev [13] proposed an analytical tool for a comparison of both the proactive and reactive schemes in the presence of faulty links. In [13], the authors used a different model of energy consumption. Furthermore, the mobility impact was not taken into consideration.

Zhao and Tong [14] proposed an analytical model focusing on the energy consumption in proactive and reactive schemes, delving further into the asymptotic behavior of the routing schemes. One of the goals is to investigate the relationship between the mobility and the energy consumption of the routing schemes, whereas the authors in [14] concentrated on the impact of the traffic conditions.

Xu et al. [15] presented a unified framework for the evaluation of the performance of the proactive and reactive routing schemes. In [15], network configurations varied for the traffic, mobility, and network density for the performance of the reactive or proactive routing schemes. The analysis in [15], however, differs from ours in both the modeling of the energy consumption and the focus of the analysis. For example, the expected energy consumption (J/bit) as shown in our study was not included in the analysis of [15].

3. System Model

3.1. Network Model. We take into consideration a network with N mobile sensor nodes distributed randomly in a square network of size $L \times L$ (see Figure 1). The average node degree, denoted as d, represents the average number of 1-hop neighbors. We assume that all the nodes have the same wireless transmission range r. Two nodes are considered neighbors if they are within the transmission range of one another. We also assume that the channels are error-free and no collisions.

Mobile nodes move according to the random direction mobility model (RDMM) [16]. In this model, mobile nodes choose a random direction and velocity at every epoch. We take into consideration that the number of packets generated at each node is distributed uniformly (λ_p packets per unit time). Each packet contains L_m message bits.

3.2. Energy Model. The energy consumption in the mobile sensor network is categorized as four operating modes: sleep, listening, reception, and transmission. Each node goes to sleep for some time and then wakes up and listens to see if any other node wants to talk to it. The energy consumed by the sleep and listening mode is E_{sleep} (Jules per unit time) and E_{listen} (Jules per bit interval), respectively. When the node detects a transmission from other nodes, it consumes

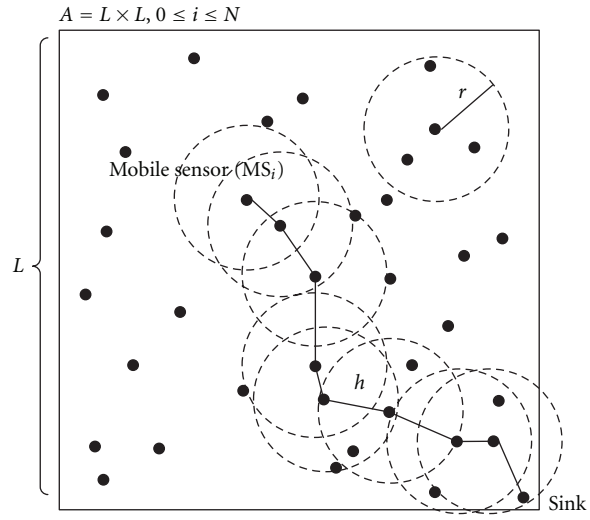


Figure 1: The network model.

Table 1: Energy consumed by reception and transmission. (CC2420, 250 kbps).

Energy	Signal strength (dBm)	Jules per bit (uJ/bit)
E_{tx}	0	0.122
	−1	0.113
	−3	0.105
	−5	0.088
	−10	0.078
	−15	0.069
	−25	0.060
E_{rx}	—	0.140

receiving energy E_{rx} (Joules/bit). The energy consumed by the transmission that covers the neighborhood of a given radius (r) is $E_{tx}(r)$ (Joules/bit). Table 1 shows the energy consumed by the reception and transmission in the case of a CC2420 radio transceiver [17].

3.3. Link Availability and Path Availability. In proactive and reactive schemes, link breaks caused by node mobility lead to the degradation of routing performance by reconstruction and rediscovery in order to update fresh entries in the routing table. As node mobility becomes higher, the likelihood that the link between two nodes will be valid decreases.

The link availability can be defined as the probability that any link between two mobile nodes will be valid from time t_0 to time $t_0 + T$ ($T > 0$) given that they can communicate directly in time t_0. This is given by

$$P\left(T, \varphi_\nu\right) = 1 - \phi\left(\frac{1}{2}, 2, \frac{-4r^2}{\alpha}\right) \quad \text{where } \alpha = \frac{4T}{\lambda}\left(\sigma_{\nu^2} + \mu_{\nu^2}\right)\varphi_\nu = \langle \mu_\nu, \sigma_\nu \rangle, \tag{1}$$

where $1/\lambda$ is the mean epoch length for node, and $\phi(x, y, z)$ is the Kummer confluent hypergeometric function [18].

For a path with h hops, path availability is the product of the individual link availabilities of the h hops [19]. Therefore, the path availability in terms of the probability that the path with h hops will be valid during time T is given by

$$P_\nu\left(T, \varphi_\nu\right) = \prod_1^h P\left(T, \varphi_\nu\right). \tag{2}$$

Assuming that a significant number of links are involved in a path, the path availability for an h hop path can be simplified to

$$P_h(T, \lambda_c) = \prod_1^h e^{-\lambda_c T} = e^{-\lambda_c h T}, \tag{3}$$

where λ_c is the topology change rate (related to the node mobility [15]). Equation (3) is verified in [19] showing that the path duration distribution can be approximated by an exponential distribution, as the number of hops along a path increases.

By incorporating the path availability in the modeling of the routing schemes, we can study the relationship between node mobility and the routing energy consumption as a probabilistic model. Table 2 shows the notations and functions in this paper.

4. Analysis of the Energy Consumption of the Various Routing Schemes

In this section, we study the expected energy consumed by the proactive, reactive, and flooding schemes while taking into specific consideration the node mobility. To analyze a comparative performance of the routing schemes, we consider a variant routing scheme similar to DSDV [3], AODV [5], and pure flooding [8] for the proactive, reactive, and flooding schemes, respectively.

4.1. Energy Consumption of the Proactive Scheme. We evaluate the expected energy consumption per unit time of the proactive scheme. The expected energy consumption of proactive routing can be expressed by

$$\varepsilon^P\left(\lambda_p, \lambda_c\right) = \varepsilon_{\text{overhead}}\left(\lambda_p, \lambda_c\right) + \varepsilon_{\text{data}}\left(\lambda_p, \lambda_c\right), \tag{4}$$

where λ_p is the packet arrival rate, and λ_c is the topology change rate. In a proactive scheme, the route information is maintained regardless of the packets arrivals. We consider that each node periodically broadcasts its HELLO message which contains their own ID in order to maintain an active neighbor set. The energy consumed by the periodic control traffic (i.e., the HELLO message) can be represented by

$$\varepsilon_{\text{hello}} = L_{\text{id}} \times \left(E_{\text{tx}}(r) + \frac{\pi r^2}{A}(N-1)E_{\text{rx}}\right), \tag{5}$$

where L_{id} is the number of bits for the ID. All neighbor nodes decode the message for each HELLO message transmission. We assume that the average number of neighbors

Table 2: Notations used in this paper.

Notation	Description
N	The total number of nodes.
A	The square area of topology.
d	The average degree of nodes.
r	The effective communication radius.
$E_{\text{tx}}(r)$, E_{rx}, E_{listen}	The transmission, receiving, and listening energy consumption, respectively.
λ_p	The packet arrival rate at each node which contains L_h header and L_m data bits.
λ_c	The topology change rate experienced by each node in one unit time. The average number of changes in the neighbor set experienced by a node in one unit time.
λ_h	The rate of HELLO message.
$L_{\text{id}}, L_p, L_q, L_e, L_{\text{ack}}$	The bit length of address, RREP, RREQ, RERR, and ACK message, respectively.
h	The number of hops between the source and sink node.
φ_v	The mobility profile which is a set of the mean and standard deviation of node's speed, that is, $\langle \mu_v, \sigma_v \rangle$.
$\varepsilon^P(\lambda_p, \lambda_c)$	The expected energy consumption per unit time of proactive scheme.
$\varepsilon^R(\lambda_p, \lambda_c)$	The expected energy consumption per unit time of reactive scheme.
$\varepsilon^F(\lambda_p, \lambda_c)$	The expected energy consumption per unit time of flooding scheme.
$L(\lambda_p, \lambda_c)$	The likelihood function for the decision routing scheme.
T_m	The packet arrival interval for the sink node at each node, that is, $1/\lambda_{\text{p}}$.
$T_d(h)$	The end-to-end delay for a particular h hop active path.

is $\pi r^2/A(N-1)$. And then, if a node detects a change in its neighbor set, the node broadcasts the new neighbor set over the network. When links change, every node exchanges the routing information by means of a DUMP mechanism, as mentioned in [3],

$$\varepsilon_{\text{dump}} = N \times L_{\text{id}} \frac{\pi r^2}{A} N \times \left(E_{\text{tx}}(r) + \frac{\pi r^2}{A}(N-1)E_{\text{rx}} \right), \quad (6)$$

where $L_{\text{id}}(\pi r^2/A)N$ is the number of bits for the routing table of N nodes. We assume that a broadcast DUMP message initiated by a sensor node will reach all the nodes in the networks. Assuming that the rate of the periodic HELLO message is λ_h and the topology change rate is λ_c, we can obtain the expected routing overhead for the proactive scheme by combining (5) and (6). Thus, we have

$$\begin{aligned} &\varepsilon^P_{\text{overhead}}\left(\lambda_p, \lambda_c\right) \\ &\quad = N\lambda_h \varepsilon_{\text{hello}} + N\lambda_c \varepsilon_{\text{dump}} \\ &\quad = N\lambda_h \left\{ L_{\text{id}} \left(E_{\text{tx}}(r) + \frac{\pi r^2}{A}(N-1)E_{\text{rx}} \right) \right\} \\ &\qquad + N\lambda_c \left\{ N L_{\text{id}} \frac{\pi r^2}{A} N \left(E_{\text{tx}}(r) + \frac{\pi r^2}{A}(N-1)E_{\text{rx}} \right) \right\}. \end{aligned} \quad (7)$$

Taking into consideration the number of expected topology changes, the rate of the HELLO messages (λ_h) generated by each node should be more than the topology change rate (λ_c) of the network.

We assume that each sensor node sends λ_p data packets per unit time to a sink node. If a route is found, it immediately sends the data packet. The receiver node sends an acknowledge packet (ACK) as the confirmation of their reception of the packet. Thus, the energy consumed by successful delivery of a data packet is given by

$$\varepsilon_s = (L_h + L_m + L_{\text{ack}}) \times \left(E_{\text{tx}}(r) + E_{\text{rx}} + \frac{\pi r^2}{A}(N-2)E_{\text{listen}} \right) \times h. \quad (8)$$

We consider that a node attempts to retransmit until some maximum number of retransmission (n) in order that conformation is received. The time taken for a message to be transmitted across a path with h hops is assumed to be randomly distributed with a mean value of $T_d(h)$ seconds. The expected energy consumption per unit time of n data retransmission mechanism can be calculated by

$$\begin{aligned} &\varepsilon_{n-\text{data}}\left(\lambda_p, \lambda_c\right) \\ &\quad = N\lambda_p \left\{ \sum_{k=1}^{n} \Big((1 - P_h(T_d(h), \lambda_c))^{k-1} \right. \\ &\qquad \times P_h(T_d(h), \lambda_c) \Big((k-1)\varepsilon_f + \varepsilon_s \Big) \Big) \\ &\qquad \left. + (1 - P_h(T_d(h), \lambda_c))^n n \varepsilon_f \right\} \\ &\quad = N\lambda_p \left\{ \sum_{k=1}^{n} \left\{ (1 - P_h(T_d(h), \lambda_c))^{k-1} \right. \right. \\ &\qquad \left. \times P_h(T_d(h), \lambda_c) \left(\frac{\varepsilon_s}{2} k + \frac{\varepsilon_s}{2} \right) \right\} \\ &\qquad \left. + (1 - P_h(T_d(h), \lambda_c))^n n \frac{\varepsilon_s}{2} \right\} \end{aligned}$$

$$= N\lambda_p \frac{\varepsilon_s}{2} \left\{ \sum_{k=1}^{n} \left\{ (1 - P_h(T_d(h), \lambda_c))^{k-1} P_h \times (T_d(h), \lambda_c)(k+1) \right\} + (1 - P_h(T_d(h), \lambda_c))^n n \right\}$$
$$= N\lambda_p \frac{\varepsilon_s}{2} \left\{ \frac{1 - (1 - P_h(T_d(h), \lambda_c))^n}{P_h(T_d(h), \lambda_c)} + 1 - (1 - P_h(T_d(h), \lambda_c))^n \right\}, \quad (9)$$

where $N\lambda_p$ is the average number of packets generated per unit time and $\varepsilon_f = \varepsilon_s/2$. As the maximum number of retransmission goes to infinity ($n \to \infty$), the energy consumed by data retransmission can be obtained as follows:

$$\lim_{n \to \infty} \varepsilon_{\text{data}}(\lambda_p, \lambda_c) = N\lambda_p \frac{\varepsilon_s}{2} \left(\frac{1}{P_h(T_d(h), \lambda_c)} + 1 \right). \quad (10)$$

A plot for (4) is shown in Figure 2. This plot shows the expected energy consumption of the proactive scheme according to the topology change rate λ_c and the packet arrival rate λ_p for $n = 5$ and $n = \infty$ where $N = 120$, $L = 500$, and $r = 80$. In this figure, we can see that the energy consumed by the proactive scheme increases exponentially as the level of mobility increases due to the data retransmission resulting from unsuccessful delivery. Meanwhile, as the traffic load becomes heavier, the energy consumption increases slightly in a linear fashion. This is because the route information is maintained regardless of the packet's arrivals. Furthermore, the increase by traffic load is significant when mobility is high, due to the retransmission overhead.

4.2. Energy Consumption of the Reactive Scheme. The expected energy consumption per unit time of the reactive scheme is the sum of the amount of energy required by the route discovery process and data transmission per unit time

$$\varepsilon^R(\lambda_p, \lambda_c) = \varepsilon^R_{\text{overhead}}(\lambda_p, \lambda_c) + \varepsilon_{\text{data}}(\lambda_p, \lambda_c). \quad (11)$$

When the route path is active in a cached routing table, the node can transmit without the routing overhead. If there is no cached entry in the routing table, the source node initiates a route discovery process by broadcasting a route request (RREQ) packet, which is received and rebroadcasted by other nodes until it reaches its destination. We assume that an RREQ packet will reach almost every node N in the network. Thus, the energy consumed by an RREQ broadcast can be computed as

$$\varepsilon_{\text{rreq}} = N \times L_q \times \left(E_{\text{tx}}(r) + \frac{\pi r^2}{A}(N-1)E_{\text{rx}} \right), \quad (12)$$

where L_q is the number of bits for an RREQ packet. Then, the destination node responds to the RREP back to the source node with a unicast reply mechanism, and as a result, the energy consumed by the RREP is

$$\varepsilon_{\text{rrep}} = L_p \times \left(E_{\text{tx}}(r) + E_{\text{rx}} + \frac{\pi r^2}{A}(N-2)E_{\text{listen}} \right) \times h. \quad (13)$$

However, when a route breakage occurs during the route discovery process, the intermediate node which detects the route breakage returns a route error message (RERR) towards the source node

$$\varepsilon_{\text{rerr}} = L_e \times \left(E_{\text{tx}}(r) + E_{\text{rx}} + \frac{\pi r^2}{A}(N-2)E_{\text{listen}} \right) \times h. \quad (14)$$

To describe the traffic condition in the mobile sensor network, the interarrival intervals of the data packets in the sensor nodes are assumed to be fixed as T_m (i.e., $1/\lambda_p$). If a path for the sink node is valid during T_m, the sensor node immediately begins to forward a data packet to the sink node without a route discovery process. Unless the path is valid, the node reinitiates a route discovery process by first sending an RREQ. Since $N\lambda_p$ route request produced every unit time can cause consecutive discovery process trials, the amount of energy required by the overhead for the reactive scheme is given by

$$\varepsilon^R_{\text{overhead}}(\lambda_p, \lambda_c) = N\lambda_p \left\{ (1 - P_h(T_m, \lambda_c)) * \varepsilon_{m-\text{discovery}}(m, T_d(h), \lambda_c) \right\}. \quad (15)$$

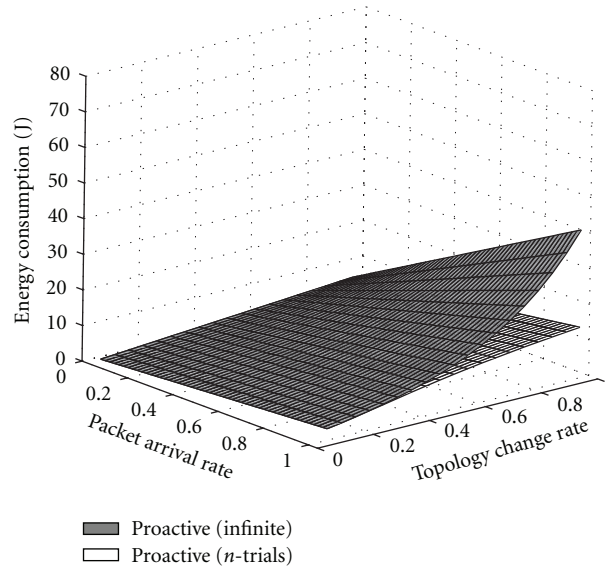


Figure 2: The energy consumption of the proactive scheme.

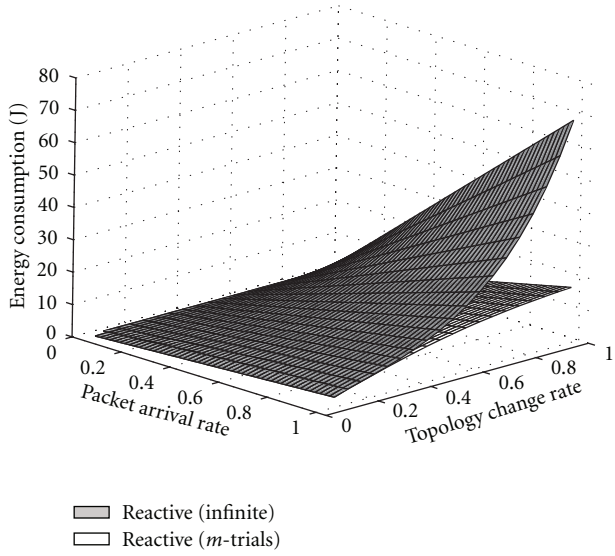


Figure 3: The energy consumption of the reactive scheme.

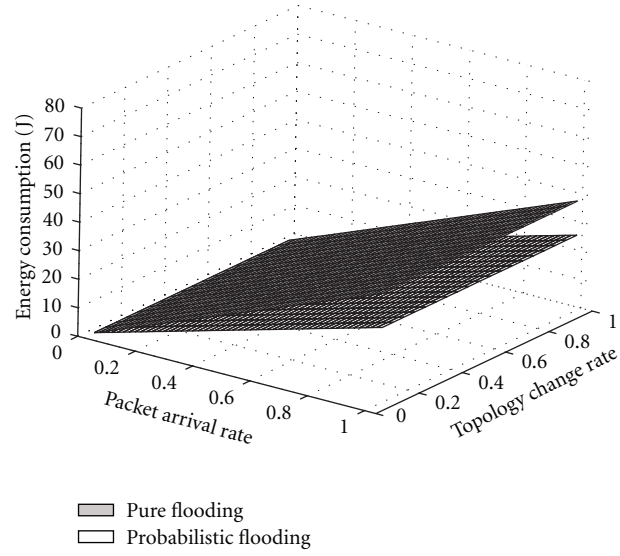


Figure 4: The energy consumption of the flooding scheme.

If an RREP is not received within a certain period, the source node rebroadcasts the RREQ with some maximum number of retransmission attempts (m)

$$\begin{aligned}
&\varepsilon_{m-\text{discovery}}(m, T_d(h), \lambda_c) \\
&\quad = \sum_{k=1}^{m} \Big\{ (1 - P_h(T_d(h), \lambda_c))^{k-1} P_h(T_d(h), \lambda_c) \\
&\qquad \times \Big(k\varepsilon_{\text{rreq}} + (k-1)\Big(\frac{\varepsilon_{\text{rrep}} + \varepsilon_{\text{rerr}}}{2} \Big) + \varepsilon_{\text{rrep}} \Big) \Big\} \\
&\quad = \sum_{k=1}^{m} \Big\{ (1 - P_h(T_d(h), \lambda_c))^{k-1} P_h(T_d(h), \lambda_c) \\
&\qquad \times \Big(k\varepsilon_{\text{rreq}} + k\varepsilon_{\text{rrep}} \Big) \Big\} \\
&\quad = \Big(\varepsilon_{\text{rreq}} + \varepsilon_{\text{rrep}} \Big) \sum_{k=1}^{m} k (1 - P_h(T_d(h), \lambda_c))^{k-1} \\
&\qquad \times P_h(T_d(h), \lambda_c) \\
&\quad = \Big(\varepsilon_{\text{rreq}} + \varepsilon_{\text{rrep}} \Big) \Big(\frac{1}{P_h(T_d(h), \lambda_c)} \\
&\qquad - \frac{(1 - P_h(T_d(h), \lambda_c))^m}{P_h(T_d(h), \lambda_c)} \\
&\qquad - m(1 - P_h(T_d(h), \lambda_c))^m \Big),
\end{aligned} \tag{16}$$

where m is the maximum number of retransmission attempts.

Assuming that the route discovery process continues until the source node successfully receives the RREP reply, we can simplify the energy consumption for a discovery process to

$$\lim_{m \to \infty} \varepsilon_{m-\text{discovery}}(m, T_d(h), \lambda_c) = \frac{\big(\varepsilon_{\text{rreq}} + \varepsilon_{\text{rrep}} \big)}{P_h(T_d(h), \lambda_c)}. \tag{17}$$

Thus, this retransmission procedure is expected to contribute immensely to the energy consumption. After the discovery processes, the node transmits the buffered data packets based on the routing table. We consider that the energy consumed by the data transmission is the same as (15) in the proactive scheme.

A plot for the expected energy consumed unit time by the reactive scheme can be described as a function of the topology change rate λ_c and packet arrival rate λ_p as illustrated in Figure 3. In the reactive scheme, as the traffic load becomes heavier, the increase of energy consumption is greater than that of the proactive scheme. This is largely due to the repetitive RREQ attempts in addition to the data retransmissions. RREP loss due to high mobility leads to degradation of the reactive scheme.

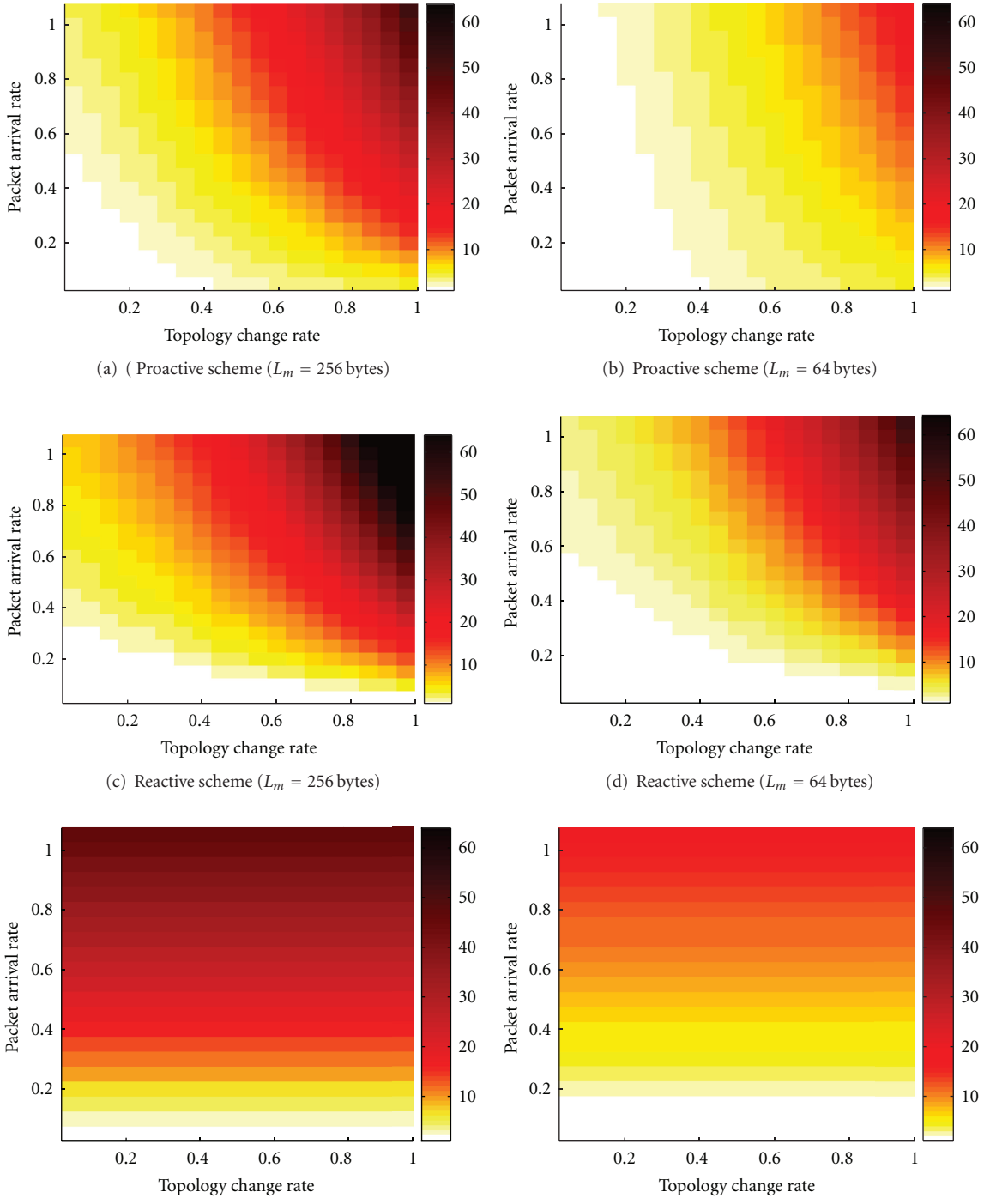
4.3. Energy Consumption of the Flooding Scheme. To analyze the energy consumption of the flooding mechanism, we start with the case of pure flooding ($\varepsilon_{\text{pure}}$) in which all the nodes rebroadcast messages when a node receives a packet for the first time [8]. We assume that a broadcast packet initiated by a source node will reach the other nodes in the network. As the average number of packets generated per unit time in the network is $N\lambda_m$, the expected energy consumed per unit time in the entire network is given by

$$\begin{aligned}
&\varepsilon^F\big(\lambda_p, \lambda_c\big) \\
&\quad = N\lambda_p \Big\{ N \times (L_h + L_m) \times \Big(E_{\text{tx}}(r) + \frac{\pi r^2}{A}(N-1) E_{\text{rx}} \Big) \Big\},
\end{aligned} \tag{18}$$

where $(L_h + L_m)$ is the size of the flooded packet. Additionally, we can consider the energy consumption of a probabilistic flooding mechanism [8] in which messages are rebroadcast with a certain fixed probability

$$\begin{aligned}
&\varepsilon^F\big(\lambda_p, \lambda_c\big) \\
&\quad = N\lambda_p \Big\{ P_f \times N \times (L_h + L_m) \times \Big(E_{\text{tx}}(r) + \frac{\pi r^2}{A}(N-1) E_{\text{rx}} \Big) \Big\},
\end{aligned} \tag{19}$$

where P_f is the fixed probability.



(a) (Proactive scheme (L_m = 256 bytes)

(b) Proactive scheme (L_m = 64 bytes)

(c) Reactive scheme (L_m = 256 bytes)

(d) Reactive scheme (L_m = 64 bytes)

(e) Flooding scheme (L_m = 256 bytes)

(f) Flooding scheme (L_m = 64 bytes)

Figure 5: The energy consumption pattern according to the topology change rate and packet arrival rate.

Figure 4 shows the plot of the expected energy consumption of the pure flooding and the probabilistic flooding schemes. Since the flooding-based routing scheme has no routing information, the energy consumption of the flooding scheme is independent of the mobility pattern.

5. The Mobility Impact on Routing Energy Consumption

In the previous section, we investigate the probabilistic energy consumption model for the proactive, reactive, and

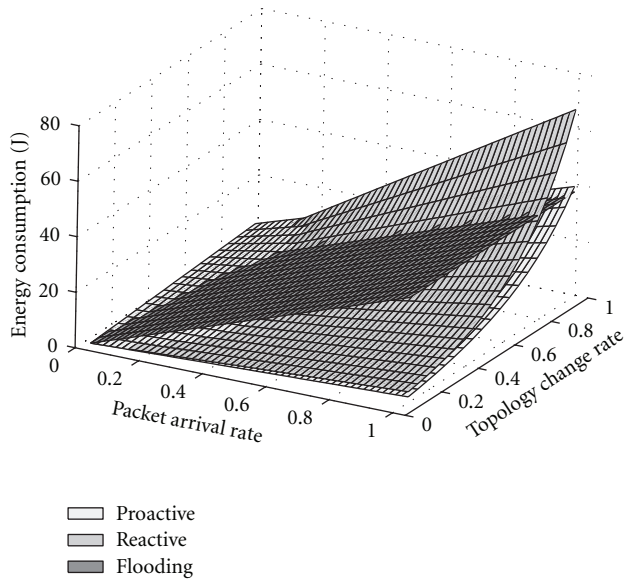


Figure 6: The comparison of the energy consumption of the routing schemes.

flooding schemes in mobile sensor networks. The results described in the previous section provide some insight into the notion that each routing scheme reacts differently during link failures. In this section, we analyze the impact of mobility on the energy consumption of the routing schemes under various networks configurations. Our routing decision approach is designed to select the most energy-effective routing schemes by finding the routing scheme i that minimizes the energy consumption $\varepsilon_i(\lambda_p, \lambda_c)$ for an energy-constrained sensor application, which is shown as follows:

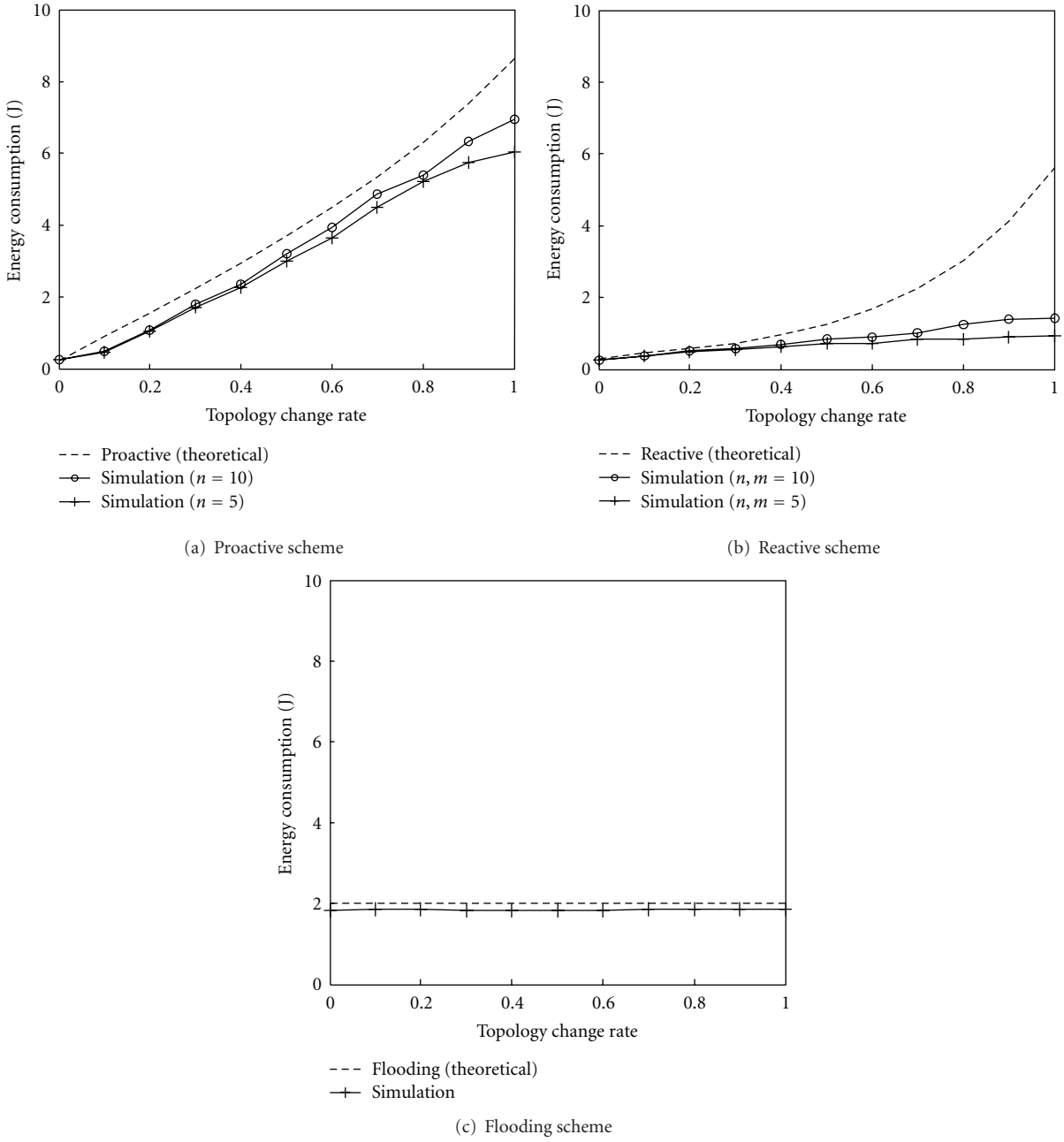
$$D\left(\lambda_p, \lambda_c\right) = \arg\min_{i \in \{F,P,R\}} \varepsilon_i\left(\lambda_p, \lambda_c\right). \tag{20}$$

Figure 5 shows a different energy consumption pattern according to the topology change rate and packet arrival rate for the proactive, reactive, and flooding schemes where $N = 120$, $L = 500$, and $r = 80$. The level of shade toward black means the higher energy consumption. For fairness of routing comparison, we assume the infinite discovery process (as given in (17)) for the reactive scheme. In addition, we consider that the length of the data packet is small (256 bytes, 64 bytes) for the mobile sensor networks [20]. This energy consumption pattern brings out some important characteristic differences between the routing schemes. Our findings show that an increase in the energy consumption is noted as the topology change rate increases in the proactive and reactive schemes. This result was highly expected, since previous studies have come to similar conclusions. However, one of our contributions is to analyze the difference in the energy consumption pattern between the proactive and reactive schemes by comparing the overhead due to the periodic control traffic with that caused by the route discovery. In a proactive scheme, whenever the current route is broken, the control overhead of the route maintenance will be flooded due to continuous route updating. In a reactive scheme, the frequent changes of the topology can lead to the establishment of broken routes which can cause the repetitive route discovery process. Meanwhile, the energy consumption of the flooding scheme is independent of mobility, as illustrated in Figures 5(e) and 5(f). When the traffic load is heavier and the node mobility is lower, the proactive scheme consumes relatively less power. In contrast, when the traffic load is lighter and the node mobility is higher, the reactive scheme is more beneficial. If mobility is much higher, both the proactive and reactive schemes can be useless. In this case, the flooding scheme can be more efficient compared to both the proactive and reactive schemes. Especially, we can see that the flooding scheme performs the best among the evaluated schemes when the length of the data is small ($L_m = 64$ bytes) and the rate of information transmission is low ($\lambda_p < 0.1$), as shown in Figures 5(b), 5(d), and 5(f). For highly dynamic networks, designing an energy-conserving routing scheme is an important issue because the flooding scheme is inherently expensive in energy cost.

We also observe that the reactive scheme is more sensitive to the topology change rate than the proactive scheme as the traffic load increases. In proactive scheme, the control messages such as DUMP or INCREMENT [3] are not retransmitted. However, in the reactive scheme, the control messages such as RREQ are retransmitted for a limited number of times if an RREP is not received within a certain interval. Due to an increase in such retransmissions, the energy consumption of reactive scheme significantly increases as both the packet arrival rate and the topology changes rate increase. This means that the increase of energy consumption caused by a repetitive route discovery process can be greater than that of the periodic control traffic when the mobility is higher. From Figures 5(c) and 5(d), the sensitivity to the mobility is less as the message length is smaller in reactive scheme. A more extensive performance comparison of the proactive, reactive, and flooding schemes can be found in Figure 6. Our analysis can be used to decide a routing scheme that will operate most energy efficiently for a sensor application, taking into account the mobility as well as traffic condition.

6. Simulation

We compare the analytical results obtained in Section 4 against simulation results. We initially consider 120 nodes initially randomly distribute in a square area with a size of 500 m $\times$ 500 m. Each node has the same transmit power of coverage of 80 m. After the initial placement, nodes keep moving continuously according to the RDMM model where every node is moving at the same constant speed and only its direction is changed. The topology change rate is estimated from the velocity by heuristic method. The traffic of the activated nodes is set to be the constant bit rate (CBR) with a packet size of 256 bytes. We consider that the energy consumption of reception and transmission for the sensor nodes is equal to the case of a CC2420 radio transceiver [17]. For each configuration, a simulation result is obtained from ten random runs. In addition, since we are interested in the



(a) Proactive scheme

(b) Reactive scheme

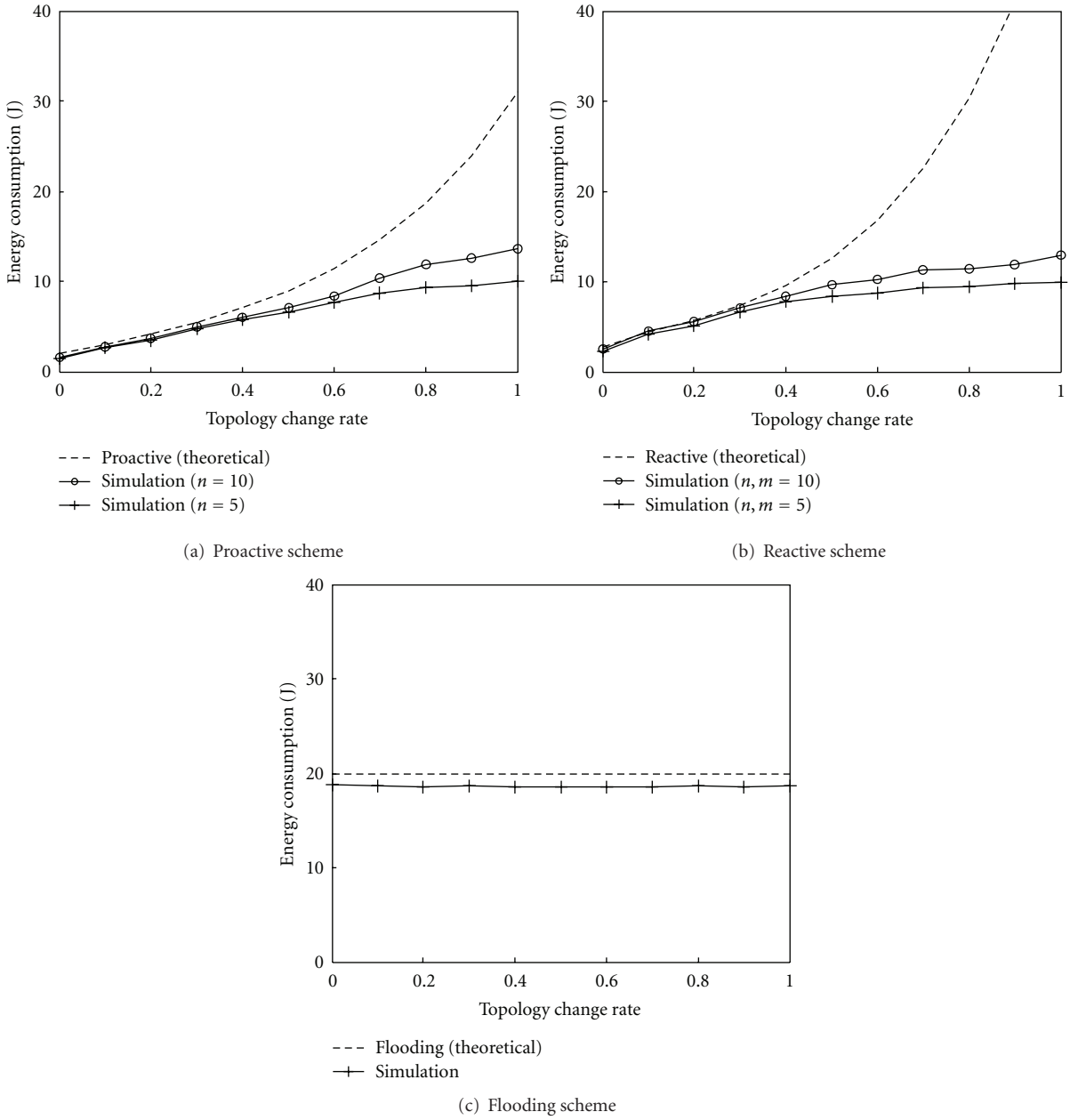
(c) Flooding scheme

Figure 7: Comparison of analytical and simulation results under light traffic load ($\lambda_p = 0.05$).

steady state, we ignore the simulation data earlier than 3 seconds from the start time.

We use the NovaSim simulator [21] and compare the proactive ($n = 5, 10$), reactive ($n = 5, 10$ and $m = 5, 10$), and flooding schemes (pure, probabilistic) for the various topology change rates and the packet arrival rates. One of our goals is to analyze the difference in the energy consumption pattern between the proactive and reactive schemes by comparing the overhead due to the periodic control traffic with that caused by the route discovery. Table 3 shows some of the important simulation parameters.

Figure 7 shows the comparison of the energy consumption under light traffic load ($\lambda_p = 0.05$). As shown in Figure 7(a), the proactive scheme is dominated by maintaining periodically the routing table no matter if the nodes need them or not. The reactive scheme, on the other hand, finds a route only when the node needs to send. It seems that the reactive scheme performs the best in all cases in terms of energy consumption. However, taking into account the packet delivery ratio, the reactive scheme becomes the worst one at higher mobility ($\lambda_c > 0.8$). This is because consecutive RREQ attempts are to fail and the



(a) Proactive scheme

(b) Reactive scheme

(c) Flooding scheme

FIGURE 8: Comparison of analytical and simulation results under heavy traffic load ($\lambda_p = 0.5$).

source node will drop the packet. A frequent change in the topology causes the RREP to possibly not arrive at the node. It can be observed in Figure 7(b) that the initial energy consumption is dependent on the packet arrival rate and becomes saturated as the topology change rate increases. In fact, the number of packet drops increases drastically at $\lambda_c > 0.5$. The discrepancies between the analytical results and the corresponding simulation results in shown Figures 7(a), 7(b), 8(a), and 8(b) are mainly due to the fact that we use a limited number of retransmissions for the RREQ and data packets. Thus, the larger n or m is, the less the discrepancy is.

Figure 8 depicts the comparison of energy consumption at the heavy traffic load ($\lambda_p = 0.5$). When the traffic load is heavy, the proactive scheme outperforms the reactive scheme for $\lambda_c < 0.4$ as shown in Figures 8(a) and 8(b). This is because the reactive scheme is more sensitive to the topology change rate than the proactive scheme as the traffic load increases, as mentioned in Section 5. We can predict that the packet delivery rate of the proactive scheme also increases drastically at $\lambda_c > 0.4$ from the discrepancy between the analytical and simulation results in Figure 8(a). As a result, the proactive scheme performs well under a heavy traffic load for $\lambda_c < 0.4$

Table 3: Parameters used in the simulation.

Parameter	Configuration
Size of field (A)	$500 \times 500\,\text{m}^2$
Distribution of nodes	Random distribution
Number of nodes (N)	120
Transmission range (r)	80 m
Data packet size	256 bytes
Mobility model	RDMM
Traffic load	Constant bit rate
Propagation model	Free space model
Energy consumption	CC2420

and $\lambda_p = 0.5$. However, when $\lambda_c > 0.4$, both the proactive and reactive schemes can be useless due to the packet drops. The energy consumed by the flooding scheme is independent of the topology change rate, as shown in Figures 7(c) and 8(c) for all λ_c.

7. Conclusion and Future Works

We analyzed the energy consumption of the proactive, reactive, and flooding schemes. Through the analysis, it was that the performance of the routing schemes in terms of energy consumption had a strong correlation between mobility and traffic conditions. We also presented a comparative performance analysis of the routing scheme in terms of energy efficiency. A routing scheme can be determined by a range of network parameters, such as the packet arrival rate and topology change rate (related to node mobility). For the sake of validity, we demonstrated the accuracy of our approach through simulations. Our proposed approach presents an energy consumption framework that helps to strengthen and deepen our understanding of the effect of mobility and traffic load on routing schemes.

In our future work, we will focus on analyses that are more sophisticated by considering the devices characteristics for the sensing, processing, and communication units. In addition, we will study an energy-efficient algorithm to dynamically switch the routing protocols between proactive, reactive, and flooding schemes according to the mobility as well as traffic conditions.

Acknowledgments

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency), NIPA-2012-C1090-1221-0010.

References

[1] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, no. 5, pp. 51–58, 2000.

[2] J. Lian, K. Naik, and G. B. Agnew, "Data capacity improvement of wireless sensor networks using non-uniform sensor distribution," *International Journal of Distributed Sensor Networks*, vol. 2, no. 2, pp. 121–145, 2006.

[3] C. E. Perkins and P. Bhagwat, "Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers," in *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM '94)*, pp. 234–244, October 1994.

[4] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks," *Wireless Networks*, vol. 8, no. 2-3, pp. 169–185, 2002.

[5] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector(AODV) routing," Tech. Rep. RFC 3561, IETF MANET Working Group, 2003.

[6] D. Johnson, D. Maltz, and Y. Chun Hu, "The dynamic source routing protocol for mobile Ad Hoc networks (DSR)," Tech. Rep., IETF MANET Working Group, 2004.

[7] E. S. Biagioni and K. W. Bridges, "The application of remote sensor technology to assist the recovery of rare and endangered species," *International Journal of High Performance Computing Applications*, vol. 16, no. 3, pp. 315–324, 2002.

[8] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile Ad Hoc network," in *Proceedings of the Annual International Conference on Mobile Computing and Networking (MOBICOM '99)*, pp. 151–162, 1999.

[9] N. Mitton, A. Busson, and E. Fleury, "Efficient broadcasting in self-organizing sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2, no. 2, pp. 161–187, 2006.

[10] J. Gao, "Analysis of energy consumption for ad hoc wireless sensor networks using a bit-meter-per-Joule metric," *IPN Progress Report*, pp. 42–150, 2002.

[11] P. L. Yang, C. Tian, and Y. Yu, "Analysis on optimizing model for proactive ad hoc routing protocol," in *Proceedings of the IEEE Military Communications Conference (MILCOM '05)*, vol. 5, pp. 2960–2966, October 2005.

[12] N. Zhou, H. Wu, and A. A. Abouzeid, "Reactive routing overhead in networks with unreliable nodes," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom '03)*, pp. 147–160, September 2003.

[13] D. Lebedev, "A framework for the comparison of AODV and OLSR protocols," *Advanced Heterogeneous Network*, vol. 3837, pp. 98–111, 2005.

[14] Q. Zhao and L. Tong, "Energy efficiency of large-scale wireless networks: proactive versus reactive networking," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 5, pp. 1100–1112, 2005.

[15] H. Xu, X. Wu, H. R. Sadjadpour, and J. J. Garcia-Luna-Aceves, "A unified analysis of routing protocols in MANETs," *IEEE Transactions on Communications*, vol. 58, no. 3, Article ID 5426524, pp. 911–922, 2010.

[16] P. M. Melliar-Smith, E. M. Royer, and L. E. Moser, "An analysis of the optimum node density for ad hoc mobile networks," in *Proceedings of the International Conference on Communications (ICC '01)*, pp. 857–861, June 2000.

[17] A. Barberis, L. Barboni, and M. Valle, "Evaluating energy consumption in wireless sensor networks applications," in *Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD '07)*, pp. 455–462, August 2007.

[18] A. Bruce McDonald and T. Znati, "A path availability model for wireless AdHoc networks," in *Proceedings of the*

IEEE Wireless Communications and Networking Conference (WCNC '99), pp. 21–24, New Orleans, La, USA, 1999.

[19] Y. Han, R. J. La, and A. M. Makowski, "Distribution of path durations in mobile Ad-Hoc networks Palm's theorem at work," in *Proceedings of the 16th ITC Specialist Seminar on Performance Evaluation of Wireless and Mobile Systems*, August 2004.

[20] Z. Marco and K. Bhaskar, "Integrating future large-scale wireless sensor networks with the internet," USC Computer Science Technical Report CS 03-792, 2003.

[21] J. Jung, Y. Cho, Y. Kim, Y. Chung, B. Gim, and J. Kwak, "Virtual protocol stack interface for multiple wireless sensor network simulators," in *Proceedings of the 25th Annual ACM Symposium on Applied Computing (SAC '10)*, pp. 240–241, March 2010.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 258080, 10 pages
doi:10.1155/2012/258080

Research Article

Performance Evaluation of Indices-Based Query Optimization from Flash-Based Data Centric Sensor Devices in Wireless Sensor Networks

Sanam Shahla Rizvi[1] and Tae Sun Chung[2]

[1] *Department of Computer Sciences, Preston University, No. 85 Street 3, Sector H-8/1, Islamabad 44000, Pakistan*
[2] *School of Information and Computer Engineering, Ajou University, San 5 Woncheon-dong Yeongtong-gu, Suwon 443-749, Republic of Korea*

Correspondence should be addressed to Sanam Shahla Rizvi, s.s.rizvi@preston.edu.pk

Received 30 July 2011; Revised 11 October 2011; Accepted 16 October 2011

Academic Editor: Junyoung Heo

Flash memory has become a more widespread storage medium for modern wireless devices because of its effective characteristics like nonvolatility, small size, light weight, fast access speed, shock resistance, high reliability, and low power consumption. Sensor nodes are highly resource constrained in terms of limited processing speed, runtime memory, persistent storage, communication bandwidth, and finite energy. Therefore, for wireless sensor networks supporting sense, store, merge, and send schemes, an energy efficient and reliable database-based query optimization technique is highly required with consideration of sensor node constraints. Databases on hard disk drives perform data storage and retrieval using index structures which are still not practiced for sensor devices. In this paper, we evaluate different indices like B-tree, R-tree, and MR-tree by implementing them on log structured external NAND flash memory-based advanced file systems for supporting energy efficient data storage and query optimization from flash based data centric sensor devices in wireless sensor networks. Experimental results show that PIYAS (Rizvi and Chung, 2010) file system along with B-tree indexing deployed on flash memory MLC gives the significant performance in respect of high query throughput optimization and less resources consumption for wireless sensor devices.

1. Introduction

The continuous improvement in hardware design and advances in wireless communication have enabled the deployment of various wireless applications. Wireless sensor network (WSN) applications become essential tools for monitoring the activity and evolution of our surrounding environment. The examples of WSN applications include environmental and habitat monitoring, seismic and structural monitoring, surveillance, target tracking, ecological observation, and a large number of other applications.

In WSNs, monitoring can be deployed by the following three techniques. First, each sensor node transmits its generated data to a sink node immediately [1]. This approach is referred as *Sense and Send*. Second, every sensor node aggregates its own generated data and data coming from its children nodes and then sends them to its parent node [2]. This scheme is called *Sense, Merge, and Send*. Third, each sensor node stores its own generated data in its local memory. The data are aggregated and are sent to the sink node when they are queried [3]. This approach is called *Sense, Store, Merge, and Send*.

Currently, the advanced applications follow the third approach mentioned above. They store the sensor data in local on-chip and/or off-chip flash memory and perform in-network computation when required [4–6]. Such in-network storage approach significantly diminishes the energy and communication costs and prolongs the lifetime of sensor networks. As a result, many techniques in the areas of data centric storage, in-network aggregation, and query processing in WSNs have been proposed.

We compare previous schemes as shown in Table 1. Matchbox [7], the first file system for sensor nodes, provides only the append operation and does not allow the random

Table 1: Comparison of PIYAS with previous schemes.

Technique	Storage device	Storage location	Data life efficient	Device life efficient	Energy efficient	Storage efficient	Indexing
Matchbox	NOR	Internal	No	No	No	No	No
ELF	NOR	Internal	No	Yes	No	No	No
Capsule	NOR/NAND	Internal	No	Yes	Yes	No	No
MicroHash	NAND	External	No	Yes	Yes	No	Yes
PIYA	NAND	External	Yes	Yes	Yes	Yes	Yes
PIYAS	NAND	External	Yes	Yes	Yes	Yes	Yes

access of data for modification. It does not offer built-in features for device life efficiency in terms of wear-leveling. It has small size code and occupies reduced main memory footprints that rely on the number of open files. ELF [8] claims to outperform the Matchbox by higher read throughput and random access of data by timestamps. Like Matchbox and ELF, Capsule [9] is also a limited internal memory technique. It claims to outperform ELF in terms of energy efficiency. MicroHash [10] is an external large memory centric approach.

It appends the data in time series and uses the hash index structure for answering queries. It suffers from the need for extra I/O operations to maintain the huge metadata. However, none of the four previously discussed approaches consider the data life efficiency in terms of in-network data persistence as they simply erase the data to provide space for new data when the memory is exhausted. Storage efficiency in terms of optimal memory bandwidth utilization is also not guaranteed in the previous schemes as a small amount of data consume a complete memory page where remaining bytes remain unused. However, PIYA [11] and PIYAS [4] schemes provide long-term in-network data availability by retaining data in form of raw and aggregate data and provide optimal utilization of memory space by gathering data in main memory buffers. The data flush in the flash memory when the data of one complete page become available, except in exceptional cases where the sensor stops sensing and switches to its sleep mode. Plus, they offer high throughput with various natures of queries. Furthermore, PIYAS prolongs device life in terms of wear-leveling and offers higher energy efficiency.

Even though a recent study [12] shows that flash storage is two orders of magnitude cheaper than communication and comparable in cost to computation plus the fact that flash memory offers many other advantages in terms of large size and reliable storage, its special hardware read, write, and erase characteristics impose design challenges on storage systems [4, 13] (discussed in detail in Section 2.2). Additionally, due to the problems of flash memory, storage management techniques developed for disks may not be appropriate for flash.

Furthermore, for database systems, index structures are widely studied for hard disk as well as for flash memory [14]. However, in our knowledge, indices on flash based sensor devices in WSNs are never considered. Thus, we believe that implementation of indexing on resources-constrained sensor devices for data access comes up with significant performance. Therefore, to make flash media useful for sensor environments and to efficiently satisfy the network business goals and requirements relevant to sensor data storage and retrieval, a reliable data management scheme along with an efficient index structure is highly required.

In this paper, we experiment B-tree, R-tree, and MR-tree indices on advanced NAND flash-based memory management schemes to evaluate the performance effectiveness of query optimization for the resources-constrained sensor devices. Subject index structures are selected due to their sensor environment-oriented particular features, as B-tree is used for sequential data access, and it can be advantageous for networks accumulating data for environmental and habitat monitoring. R-tree and MR-tree structures are used for indexing multidimensional information like geographic information system. They can be significant to access the spatial objects such as restaurant locations and typical maps made for streets, buildings, outlines of lakes, and coastlines. Therefore, the increase in performance totally depends on selection of right type of index structure according to sensor environment.

The remainder of this paper is organized as follows. We review the background of system architecture of sensor node plus flash memory, and different indices characteristics are explained in Section 2. Comprehensive experiments are performed and discussed in Section 3. Finally, Section 4 presents the conclusions.

2. Background

2.1. System Architecture of Sensor Node. The architecture of the wireless sensor node consists of a microcontroller unit (MCU) that interconnects a data transceiver, sensors along with analog-to-digital converters (ADCs), an energy source, and an external flash memory, see Figure 1.

The MCU includes a processor, a static RAM (SRAM), and on-chip flash memory. The processor increases efficiency by reducing power consumption. It runs at low frequency (~4–58 MHz) and further saves energy while the node is in standby or sleep mode. The low-power microcontrollers have limited storage, typically less than 10 KB of SRAM, mainly for code execution. However, in latest generation of sensors [5], it also uses for in-memory buffering. The limited amount of on-chip flash memory provides a small nonvolatile storage area (~32–512 KB). It is used for storing

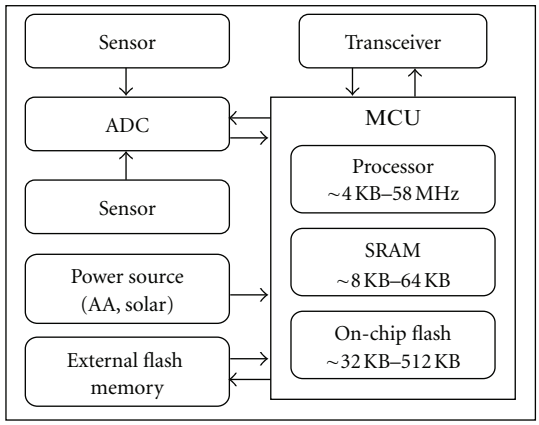


Figure 1: Sensor node architecture.

executable codes and accumulated values for a small period of time. However, it consumes most of the chip area and much of the power budget. Therefore, a larger amount of extra flash memory, perhaps more than a megabyte, is used on a separate chip to support the enhanced network functionality. The required amount of power can be obtained from many sources. Most sensors deploy a set of AA batteries and/or solar panels [15]. However, in most cases, the choice of correct energy source is application specific.

Flash memory is a nonvolatile solid state memory which has many attractive features such as small size, light weight, fast access speed, shock resistance, high reliability, and low-power consumption. Because of these attractive features, decreasing price, and increasing capacity, flash memory is becoming ideal storage media for mobile and wireless devices [16].

2.2. Overview of Flash Memory. Flash memory array is partitioned into equal size erase units called blocks, and each block is composed of a fixed number of read/write units called pages, see Figure 2. Every page has two sections: data area and spare area. Spare area stores metadata like logical block number (LBN), logical page number (LPN), erase count number (ECN), error correction code (ECC), cleaning flag for indicating garbage collection process in block, used/free flag to show that page is used or still free, and information of being valid/obsolete about data in data area. The sizes of pages and blocks differ by product.

Flash memory has three kinds of operations: page read, page write, and block erase. The performance of three kinds of operations is summarized based on memory access time and required energy at maximum values as shown in Table 2 [17].

There are two types of flash memory. Single-level cell (SLC) flash stores one bit of data (0,1) in single memory cell. Multilevel cell (MLC) flash is capable to store more than one bit of data in single cell. However, four states per cell that yield two-bit information (00, 01, 10, and 11) reduce the amount of margin separating the states and result in the possibility of more errors. Currently, the MLC flash memory is becoming popular for large size applications due to its continuously increasing capacity, decreasing price, and high throughput. However, compared to MLC, the traditional SLC flash memory still outperforms with its outstanding features like more data reliability, neglectable bit error ratio, and increased endurance cycles. Table 3 compares the specifications of modern flash SLC [18] and MLC [19] devices.

Table 2: Performance of NAND flash memory.

Operation	Time (μsec)	Energy (mA) (Current 3.3 V)
Page read (512 + 16) B	15	20
Page write (512 + 16) B	500	25
Block erase (16 K + 512) B	3,000	25

Table 3: Specification comparison: SLC and MLC memory.

	Flash SLC	Flash MLC
Capacity	64 GB	256 GB
Reads	100 MB/s (max.)	220 MB/s (max.)
Writes	80 MB/s (max.)	200 MB/s (max.)
Endurance	100,000 cycles	10,000 cycles
MTBF	2,000,000 hours	1,000,000 hours

Even though flash memory has many attractive features; its special hardware characteristics impose design challenges on storage systems. It has two main drawbacks.

First Drawback. An inefficiency of in-place-update operation. When we modify data, we cannot update data directly at the same address due to the physical erase-before-write characteristics of flash memory. Therefore, updating even one byte data in any page requires an expensive erase operation on the corresponding block before the new data can be rewritten. To address this problem, the system software called flash translation layer (FTL) was introduced, as in [20–22]. FTL uses a non-in-place-update mechanism to avoid having to erase on every data update by using logical-to-physical address mapping table maintained in main memory. Under this mechanism, the FTL remaps each update request to different empty location and then the mapping table updates due to newly changed logical-to-physical addresses. This protects one block from being erased per overwrite. The obsolete data flagged as garbage which a software cleaning process later reclaims. This process is called garbage collection, as in [23–25].

Second Drawback. The number of erase operations allowed to each block is limited like 10,000 to 1,000,000 times, and the single worn-out block affects the usefulness of the entire flash memory device. Therefore, data must be written evenly to all blocks. This operation is named as wear-leveling, as in [26, 27]. These drawbacks represent hurdles for developing a reliable flash memory-based sensor storage systems.

2.3. Index Structures. B-tree [28–30] is an optimized data structure that most representatively uses in hard disk drive-based database management systems and file systems.

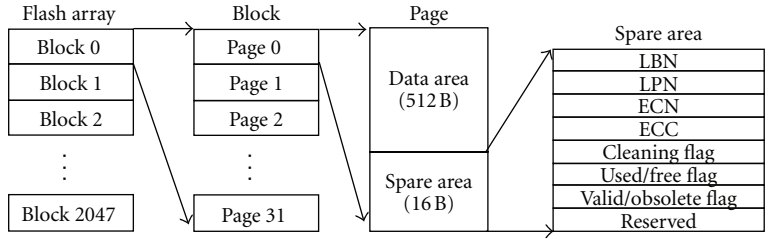


Figure 2: Flash memory (32 MB) architecture.

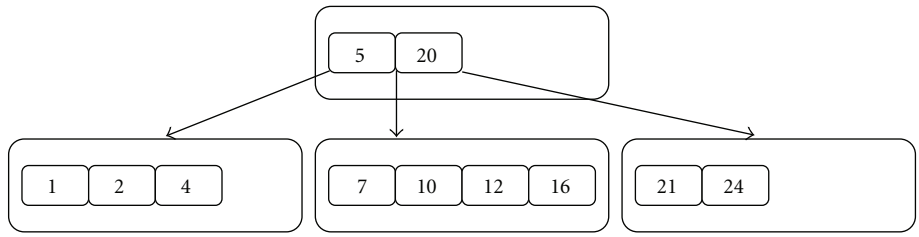


Figure 3: Structure of B-tree.

It is an effective method to keep data sorted and perform data insert, delete, search, and sequential access efficiently in logarithmic amortized time. In B-tree, a node can have variable number of child nodes within some predefined range, as shown in Figure 3. In order to maintain the range, nodes may join or split. In B-tree, every parent/internal node keeps the keys/physical addresses of records rather than leaf nodes. It keeps the records in sorted order to sequentially traverse and uses a hierarchical index to minimize the number of memory units read to access a data record.

R-tree [31, 32] data structure is similar to B-tree, but it uses for spatial access methods like multimedia data, see Figure 4. Every node in R-tree can keep a predefined number of entries where each entry of leaf node consists of information about pointer to actual data element and depth of node. R-tree is height balancing structure that often increases the height of tree to balance the leaf nodes.

MR-tree [33, 34] data structure is similar to R-tree but the height between subtrees can be unbalanced with difference by maximum 1, as shown in Figure 5. When height crosses its threshold, it runs a height balance algorithm that rearranges the entries and splits the parent nodes. MR-tree nodes are classified in parent nodes, leaf nodes, and half leaf nodes where half leaf nodes refer to the node with 1 entry. By insertion of new objects in half leaf nodes, they become leaf nodes that have height of minimum 2.

3. Performance Evaluation

3.1. Simulation Methodology. To demonstrate the performance effectiveness of index structures in sensor environment, we performed a trace-based simulation. We applied different indices on PIYAS [4], PIYA [11], and MicroHash [10] schemes and analytically compared them on different parameters. Evaluation focuses on four parameters.

(i) *Space Management.* This shows the flash memory allocation against the thousands of continuous sensor readings and main memory consumption for maintaining the data buffers and metadata.

(ii) *Search Performance.* This shows the number of pages required to be read for responding to a query.

(iii) *Throughput Performance.* This shows the response of number of queries in a unit of time.

(iv) *Energy Consumption.* This shows the energy consumption while data writes to and data reads from sensor local flash memory.

We have built a simulator with 32 MB of flash space that is divided into erase blocks of equal size. Each block size is 16 KB, and every block is composed of 32 pages as read/write units. Every page size is 512 B with 16 B spare area. We extracted the trace file from COAGMET [35]. The two years raw data were extracted on an hourly basis from January 01, 2009 to December 31, 2010 from the Willington climate station. The trace file contains a total of 279,968 sensor readings, and it is a combination of all known data formats like negative, positive, and decimal values.

Every network has business rules to achieve some business goals. To achieve services in sensor networks, business rules are an effective method for programming a file system for sensor nodes. Rules are logically linked as chain where the structure of rules represents the simple business logic in a compact and efficient way. For example, the business goal says to collect the temperature readings in discrete range from 1F to 80F. In that case, we can split the range in set of rules like (A: [1–20]), (B: [21–40]), (C: [41–60]), and (D: [61–80]). The formulation of set of rules highly depends on the probability of type of data accumulation from environment and location for implementation of sensors. Since the sensor nodes assist the real life processes, the variation in set of rules is expected to address the monitoring of service parameters. Therefore, we assume that the set of rules is available to sensor nodes from network applications.

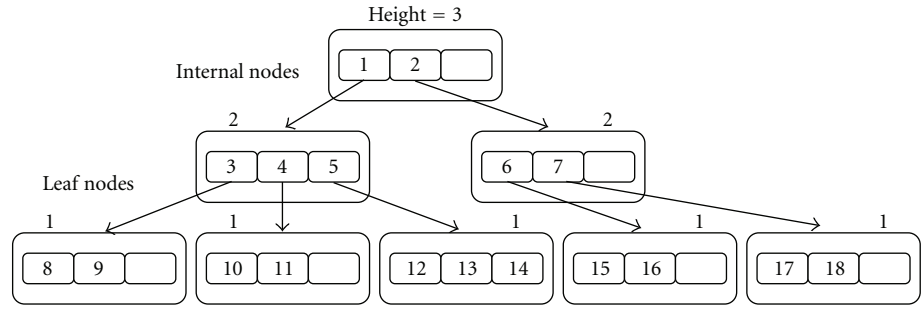


Figure 4: Structure of R-tree.

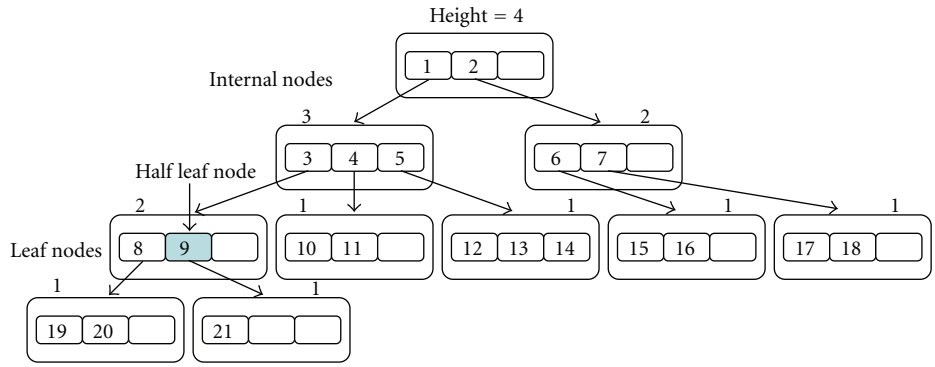


Figure 5: Structure of MR-tree.

To prove the enhancement of our idea for large size of sensor data centric applications, we experimented with a broad range of rule values. Rules are adopted as directory buckets in case of MicroHash. The rules are given in Table 4.

The total elapsed time is calculated by (1) for effective comparison between schemes. Time required for read in unit of page from flash memory to data register is calculated by (2). Time required to read a byte unit from data register to main memory is calculated by (3). Time required for computation in main memory for building mapping structure and query processing framework is calculated by (4). Time required to write data from main memory to flash media is calculated by (5). For better understanding of experimental results in terms of time and energy, we refer to Table 2,

$$\text{Total time} = \text{TR}_{\text{FR}} + \text{TR}_{\text{RR}} + \text{T}\alpha + \text{TW}_{\text{RF}}, \quad (1)$$

$$\text{TR}_{\text{FR}} = (\text{read count (page)} \times \text{read time}), \quad (2)$$

$$\text{TR}_{\text{RR}} = (\text{read count (byte)} \times \text{read time}), \quad (3)$$

$$\text{T}\alpha = \text{Time for Computation in RAM}, \quad (4)$$

$$\text{TW}_{\text{RF}} = (\text{write count (page)} \times \text{write time}). \quad (5)$$

3.2. Experimental Results. Figure 6 shows the consumption of flash memory in number of erase blocks for number of sensor readings attempted by every rule. Trigger with every individual rule buffer (TgRule) is used in SRAM for sampling sensor readings. We show the fine granularity of data arrival in buffer of every rule by taking a small value of threshold as

Table 4: Rules description for simulation.

Rule symbol	Rule range (temperature)
A	−99–0
B	1–100
C	101–200
D	201–300
E	301–400
F	401–500
G	501–600

TgRule = 3 for PIYA and PIYAS schemes, and as MicroHash does not sample data, so we show the consumption of media for MicroHash by keeping trigger unset as TgRule = 0. In figure, flash blocks are individually allocated as chains to every rule for saving the sensor data corresponding to a trigger threshold where thousands of readings are stored in a very small flash memory space by both PIYA and PIYAS schemes. MicroHash stores data in linear sequential order. Therefore, we calculated blocks consumed by MicroHash by counting the number of pages allotted to every bucket. In this result, we only show the space consumed by data pages, and space assigned to metadata is not added. However, results clearly show the effectiveness of our memory management schemes. Our proposed schemes outperform the MicroHash for efficient media utilization.

Figure 7 shows the consumption of SRAM space in KB units while sensor filters and buffers the accumulated readings. SRAM provides opportunity to reserve data buffers

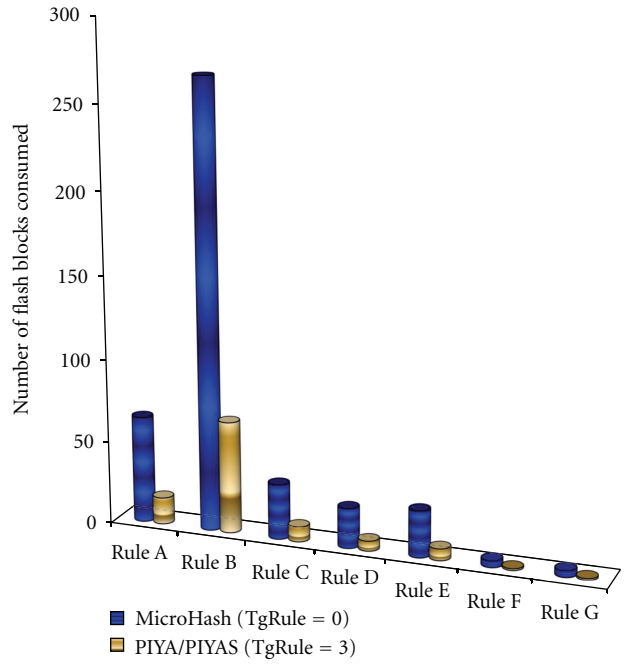


Figure 6: Flash memory consumption in number of erase units.

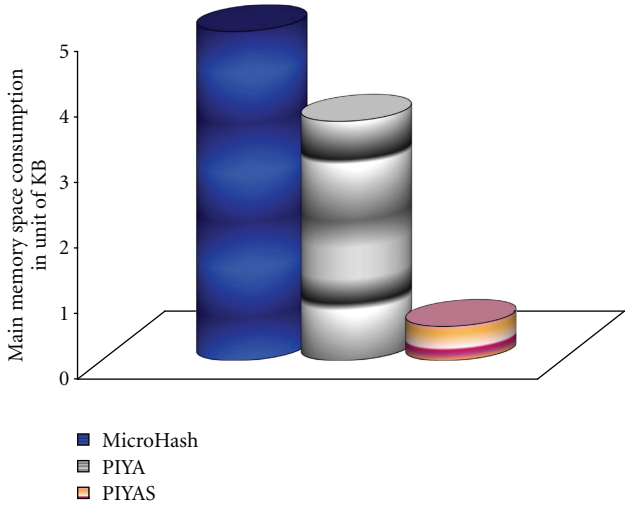


Figure 7: SRAM consumption in KBs for buffering sensor readings.

to put together the currently accumulated sensor readings from environment and then data store in a sensor's local memory. Data buffering saves the flash space and reduces the write overhead. We reserve data buffers by the number of business rules where every buffer size is of one read/write unit of flash memory. When data arrives in the range of any rule, main memory space is assigned dynamically in chunks of bytes as buffer. Data of a complete buffer flush in flash memory when it becomes full. Results show that PIYAS scheme clearly outperforms both PIYA and MicroHash schemes. This is because unlike PIYA and MicroHash, PIYAS does not allocate static buffers, but buffers are allotted dynamically in chunks of bytes whenever some sensor reading arrives in the data buffer of some rule. Therefore, even though in a very write intensive scenario, PIYAS optimizes main memory space accumulation by 71.4% and 79.2% more than PIYA and MicroHash, respectively.

Flash memory mapping information stores in flash media in dedicated map blocks for fast initialization of system. At the time of system startup, mapping information fetches in SRAM. Limited SRAM and lengthy initialization time are challenging constraints of sensor resources. Therefore, to achieve instant mounting using very small size of SRAM footprints, data is saved sequentially on first available page of latest allocated block according to some rule. Where, every rule keeps only first available physical page number (PPN) in SRAM where single page mapping reserves only 2 B in main memory for 32 MB of flash memory which has 2^{16} total number of pages. Therefore, we need only the limited number of pages mapped by the number of rules.

At system initialization time, for building the mapping table, we extract mapping information from map blocks to the main memory. We obtain a fast mounting in 136.75 μs; it consumes 0.396 J and 154 B in SRAM. Both PIYA and PIYAS schemes use same time and number of bytes while mounting the mapping structure in main memory and for saving the mapping information back to the map blocks.

System fetches the metadata from map blocks and builds the query processing framework in main memory for entertaining the read intensive scenarios efficiently. PIYA scheme extracts timestamps by reading spare area of first page of every block and sets the time between two consecutive data blocks of same rule chain. Then, the table arranges in main memory for fast access of data. When some query comes in the range of some rule, system forwards that in corresponding block according to the desired time range of query. System evaluates the timestamp written in spare area from latest written page. If page supports queried value, then system checks the data items inside data section of page, otherwise it moves one page up.

In case of a large size of space being occupied, scanning by PIYA of spare area of first page of every block to build the mapping table and then finding the exact pages by reading spare areas of every page in the corresponding block consumes a long time and high energy. Therefore, PIYAS scheme implements a more energy efficient data access and provides a high throughput for responses to user queries. It maintains the data storage log in form of metadata in dedicated map blocks separated from the file system mapping information. It stores the metadata regarding memory assigned to every rule in a particular time interval.

Though, in a read intensive scenario, PIYAS scheme preserves the average of 24 times more resources while building B-tree, R-tree, and MR-tree structures; it reserves average of 7.56 and 3.57 times more resources in terms of space, time, and energy while building the query processing framework in SRAM, compared to MicroHash and PIYA, respectively.

Figure 8 shows the query throughput by the average number of queries responded to per minute time unit. Evaluation is performed without applying any index structure on any scheme using their original mapping structures. Results

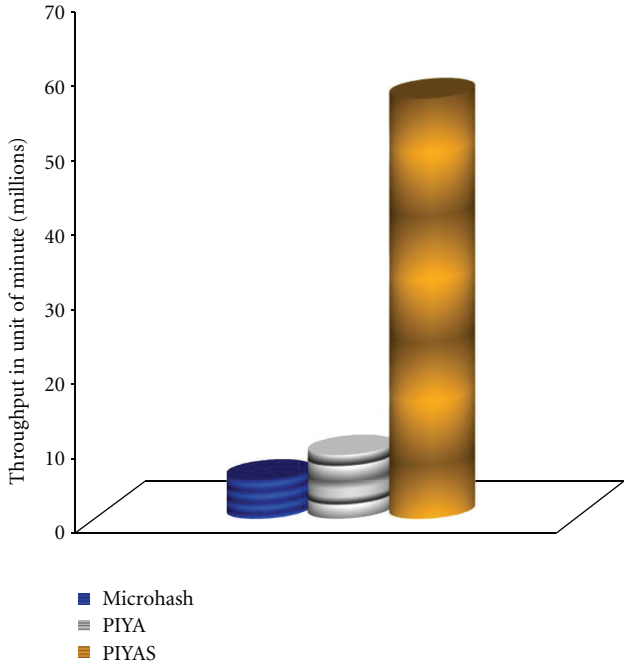


Figure 8: Throughput in unit of minute without indexing.

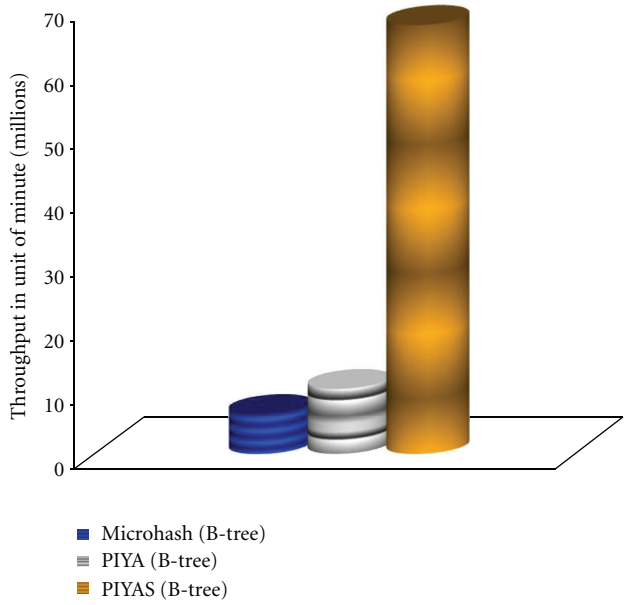


Figure 9: Throughput in unit of minute with B-tree indexing.

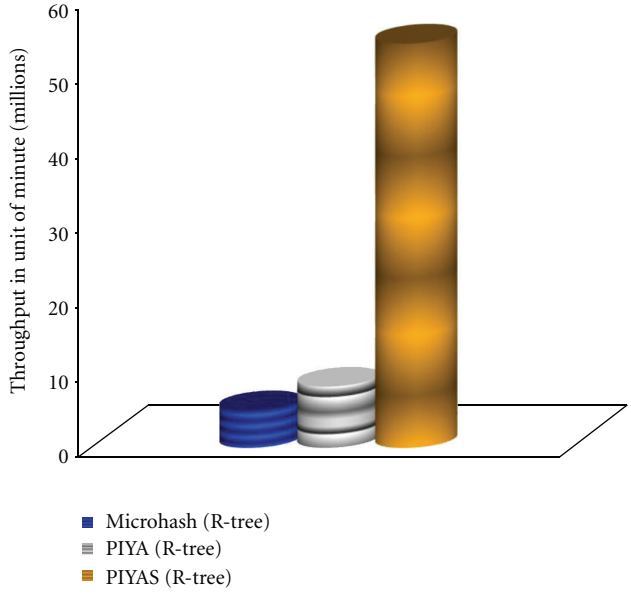


Figure 10: Throughput in unit of minute with R-tree indexing.

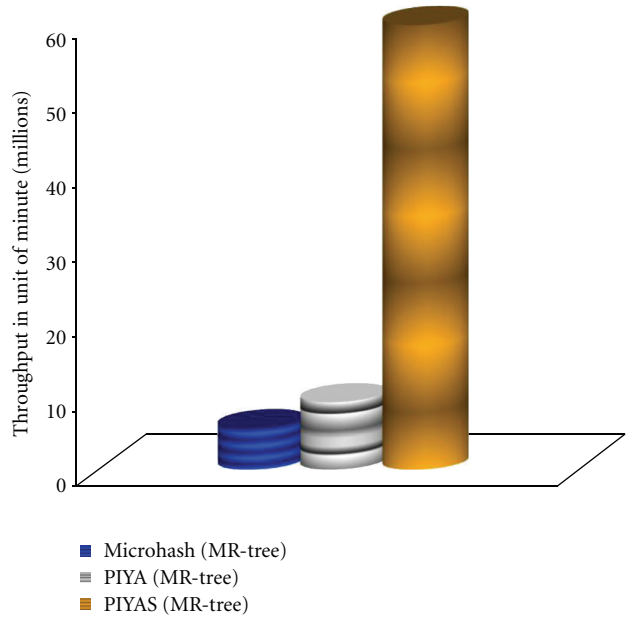


Figure 11: Throughput in unit of minute with MR-tree indexing.

show that PIYAS greatly outperforms the previous PIYA and MicroHash schemes in time required for query responses with 90% and 84% queries per minute, respectively.

Figures 9, 10, and 11 present the results of query throughput by the average number of queries responded to per minute time unit which are obtained by applying B-tree, R-tree, and MR-tree indexes, respectively, on all three memory management schemes. Comparison between indices proves the dominance of B-tree with 18.2% better performance compared to R-tree which in turn improves 9% more throughput compared to MR-tree. However, the significance of index structures totally depends on selection of type of index according to sensor environment. Since, we experiment the traces obtained from environment and habitat monitoring sensors where the queries usually perform to store and access the data sequentially. Therefore, B-tree prominent here though R-tree and MR-tree has worth in spatial sensor environment.

Alternatively, results show that PIYAS scheme significantly outperforms PIYA with 6.57 times more throughput

Table 5: Resources (time and energy) accumulation.

	Time (μs)	Energy (J)
MicroHash	2884.2	3.135
PIYA	2702.04	2.937
PIYAS	151.8	0.165
PIYAS (B-tree)	121.44	0.132
PIYAS (R-tree)	698.28	0.759
PIYAS (MR-tree)	455.4	0.495

which in turn has advantage over MicroHash by 1.64 times more throughput. Lower performance of MicroHash is observed which can be because when flash space becomes exhausted and there is no space remaining for further data storage, scheme selects the victim block for garbage collection, and data in the victim block are simply erased and then the future queries cannot access such data. This generates a data failure for user applications.

However, PIYA and PIYAS congregate the values of victim block from flash erase unit to a read/write unit where every erase unit is composed of multiple read/write units. It means that the number of pages of victim block aggregate based on user-defined parameters like MIN, MAX, AVERAGE, COUNT, and so forth, on single page size. Therefore, every page on the aggregate data block represents the major information of data of one complete previously erased raw data block. This way, they preserve in-network data sustainability and availability by allocating aggregate data blocks to rules.

Further more, PIYAS enhances the scheme, conserves the energy, and takes a reduced search time for answering any query by allocating separate aggregate data blocks to individual rules to seek the exact data corresponding to particular rule values by avoiding the unnecessary read operation as PIYA does.

Table 5 shows the resources accumulation by the schemes addressed here. This information is calculated by obtaining the results of average number of pages that system reads on every request from network applications while searching the queried data in a very read intensive environment. Experimental results show that PIYAS optimizes 94.7%, 94.4%, 78.3%, and 66.7% resources in terms of time and energy compared to MicroHash, PIYA, R-tree, and MR-tree, respectively. However, it takes 20% more resources compared to B-tree.

Until now, almost all results show the performance dominance of PIYAS over other two previous schemes. It is also observed that PIYAS scheme along with B-tree index structure performs notably well. Therefore, to show the effectiveness of flash SLC [18] and MLC [19] devices in sensor environment, an experiment is performed for average read speed and read burst speed where burst throughput is the speed that data can be accessed from drive's read-ahead memory register. This measures the speed of drive and controller interface. Such evaluation is performed using PIYAS scheme along with B-tree index structure.

Figure 12 demonstrates the results achieved per second in unit of time where MLC flash delivers by far the highest sustained more volume 48.59% for average sequential read and 57.3% for read burst throughput compared to SLC flash. Both flash SLC and MLC devices optimize random access data throughput with 0.1 ms.

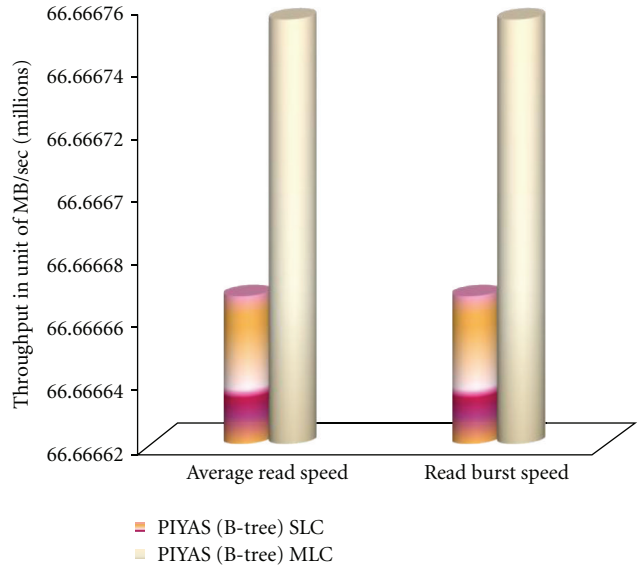


Figure 12: Average read and read burst throughput in unit of MB/s.

4. Conclusion

This research evaluates the performance effectiveness of index structures to acquire the queried data from wireless sensor networks. Different indices like B-tree, R-tree, and MR-tree are compared by implementing them on advanced log-structured external NAND flash memory-based data management schemes called PIYAS, PIYA, and MicroHash. We performed trace-driven simulations to explore in detail also the effectiveness of SLC and MLC flash devices in sensor environment. Our comprehensive experimental results with real traces from environmental and habitat monitoring show that the B-tree index structure along with PIYAS memory management scheme comes up with significant performance in terms of time, energy, and space preservation.

Plus, we achieved instant mounting and reduced SRAM footprints by keeping a very low-mapping information size. The main memory required for accumulation of sensor readings is minimized. Storage utilization is optimized by effective data buffering in main memory before writing data to flash media. Data failure is mitigated by long-term in-network data availability. Fast access of memory to write data, computation in situ, high query throughput, more energy efficiency, and minimized reads, writes, and erases are effectively achieved.

Acknowledgments

The authors wish to thank Mr. M. A. S. Rizvi, Professor and Dean, Greenwich University, Karachi, Pakistan, and Mr. W. A. S. Rizvi, Plan Controller, Huawei Technologies Pakistan (Pvt) Ltd. for their valuable time for reviewing the whole manuscript and responding with their helpful comments.

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science, and Technology (2010-0013487).

References

[1] J. Considine, F. Li, G. Kollios, and J. Byers, "Approximate aggregation techniques for sensor databases," in *Proceedings of the 20th International Conference on Data Engineering (ICDE '04)*, pp. 449–460, Boston, Mass, USA, April 2004.

[2] S. Madden, M. J. Franklin, J. Hellerstein, and W. Hong, "TAG: a tiny aggregation service for Ad-Hoc sensor networks," in *Proceedings of the Symposium on Operating Systems Design and implementation*, pp. 131–146, Boston, Mass, USA, 2002.

[3] D. Zeinalipour-Yazti, S. Neema, V. Kalogeraki, D. Gunopulos, and W. Najjar, "Data acquisition in sensor networks with large memories," in *Proceedings of the International Workshop on Biomedical Data Engineering (BMDE '05)*, pp. 1188–1192, Tokyo, Japan, 2005.

[4] S. S. Rizvi and T. S. Chung, "PIYAS-Proceeding to intelligent service oriented memory allocation for flash based data centric sensor devices in wireless sensor networks," *Sensors*, vol. 10, no. 1, pp. 292–312, 2010.

[5] D. Lymberopoulos and A. Savvides, "XYZ: a motion-enabled, power aware sensor node platform for distributed sensor network applications," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, IPSN 2005*, pp. 449–454, Los Angeles, Calif, USA, April 2005.

[6] A. Banerjee, A. Mitra, W. Najjar, D. Zeinalipour-Yazti, V. Kalogerakil, and D. Gunopulos, "RISE—Co-S: high performance sensor storage and co-processing architecture," in *Proceedings of the 2nd Annual IEEE Communications Society Conference on Sensor and AdHoc Communications and Networks (SECON '05)*, vol. 2005, pp. 1–12, Santa Clara, Calif, USA, 2005.

[7] D. Gay, "Design of matchbox: the simple filing system for motes," TinyOS 1.x Distribution, 2003, http://www.tinyos.net.

[8] H. Dai, M. Neufeld, and R. Han, "ELF: an efficient log-structured flash file system for micro sensor nodes," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pp. 176–187, Baltimore, Md, USA, November 2004.

[9] G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy, "Capsule: an energy-optimized object storage system for memory-constrained sensor devices," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pp. 195–208, Boulder, Colo, USA, November 2006.

[10] D. Zeinalipour-Yazti, S. Lin, V. Kalogeraki, D. Gunopulos, and W. A. Najjar, "An efficient index structure for flash-based sensor devices," in *Proceedings of the USENIX Conference on File and Storage Technology*, pp. 31–44, San Francisco, Calif, USA, 2005.

[11] S. S. Rizvi and T. S. Chung, "PIYA—Proceeding to Intelligent service oriented memory Allocation for flash based sensor devices in wireless sensor networks," in *Proceedings of the 3rd International Conference on Convergence and Hybrid Information Technology (ICCIT '08)*, pp. 625–630, Busan, Korea, November 2008.

[12] G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy, "Ultra-low power data storage for sensor networks," in *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN '06)*, pp. 374–381, Nashville, Tenn, USA, April 2006.

[13] E. Gal and S. Toledo, "Algorithms and data structures for flash memories," *ACM Computing Surveys*, vol. 37, no. 2, pp. 138–163, 2005.

[14] H. S. Lee, H. Y. Song, and K. C. Kim, "Performance of Index trees on flash memory," in *Proceedings of the International Conference in Principles of Information Technology and Applications*, pp. 725–734, 2007.

[15] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, "Hardware design experiences in ZebraNet," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pp. 227–238, Baltimore, Md, USA, November 2004.

[16] B. Dipert and M. Levy, *Designing with Flash Memory*, Annabooks Publisher, Poway, Calif, USA, 1993.

[17] Samsung Electronics NAND Flash Memory, *K9F5608U0D data book*, Samsung Electronics, Jung-gu Seoul, South Korea, 2011.

[18] Samsung Electronics NAND Flash Memory, *MCCOE64G5-MPP-0VA Data Book*, Samsung Electronics, Jung-gu Seoul, South Korea, 2011.

[19] Samsung Electronics NAND Flash Memory, *MMD0E56G5-MXP-0VB Data Book*, Samsung Electronics, Jung-gu Seoul, South Korea, 2011.

[20] T. S. Chung and H. S. Park, "STAFF: a flash driver algorithm minimizing block erasures," *Journal of Systems Architecture*, vol. 53, no. 12, pp. 889–901, 2007.

[21] T. S. Chung, D. J. Park, S. Park, D. H. Lee, S. W. Lee, and H. J. Song, "A survey of Flash Translation Layer," *Journal of Systems Architecture*, vol. 55, no. 5-6, pp. 332–343, 2009.

[22] S. J. Kwon and T. S. Chung, "An efficient and advanced space-management technique for flash memory using reallocation blocks," *IEEE Transactions on Consumer Electronics*, vol. 54, no. 2, pp. 631–638, 2008.

[23] T. S. Chung, M. Lee, Y. Ryu, and K. Lee, "PORCE: an efficient power off recovery scheme for flash memory," *Journal of Systems Architecture*, vol. 54, no. 10, pp. 935–943, 2008.

[24] L.-Z. Han, Y. Ryu, T.-S. Chung, M. Lee, and S. Hong, "An intelligent garbage collection algorithm for flash memory storages," in *Proceedings of the Computer Science and Its Applications, Lecture Notes in Computer Science*, pp. 1019–1027, Glasgow, UK, 2006.

[25] L. Han, Y. Ryu, and K. Yim, "CATA: a garbage collection scheme for flash memory file systems," in *Proceedings of the Ubiquitous Intelligence and Computing, Lecture Notes in Computer Science*, pp. 103–112, Wuhan, China, 2006.

[26] L. P. Chang, "On efficient wear leveling for large-scale flash-memory storage systems," in *Proceedings of the ACM Symposium on Applied Computing*, pp. 1126–1130, Seoul, Korea, March 2007.

[27] Y. H. Chang, J. W. Hsieh, and T. W. Kuo, "Endurance enhancement of flash-memory storage systems: an efficient static wear leveling design," in *Proceedings of the 44th ACM/IEEE Design Automation Conference (DAC '07)*, pp. 212–217, San Diego, Calif, USA, June 2007.

[28] B. C. Ooi and K. L. Tan, "B-trees: bearing fruits of all kinds," in *Proceedings of the 13th Australasian Database Conference*, Melbourne, Australia, 2002.

[29] J. H. Nam and D. Park, "The efficient design and implementation of the B-tree on flash memory," in *Proceedings of the 23rd Korea Information Science Society*, pp. 5–7, Seoul, Korea, 2005.

[30] C. H. Wu, L. P. Chang, and T. W. Kuo, "An efficient B-tree layer for flash memory storage systems," in *Proceedings of the*

9th International conference on Real-Time and Embedded Computing systems and Applications, pp. 406–430, 2003.

[31] A. Guttman, "R-trees: a dynamic index structure for spatial searching," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 47–57, 1984.

[32] C. H. Wu, L. P. Chang, and T. W. Kuo, "An efficient R-tree implementation over flash-memory storage systems," in *Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems*, pp. 17–24, New Orleans, La, USA, November 2003.

[33] K.-C. Kim and S.-W. Yun, "MR-tree: a cache-conscious main memory spatial index structure for mobile GIS," in *Proceedings of the Web and wireless geographic information systems*, pp. 167–180, LNCS, 2005.

[34] H. S. Lee, H. Y. Song, and K. C. Kim, "Performance improvement of MR-Tree operations on NAND Flash memory," *International Journal of Computer Science and Information Systems*, vol. 3, no. 1, pp. 59–70, 2007.

[35] "COAGMET," 2011, http://ccc.atmos.colostate.edu/~coagmet/index.php.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 156590, 7 pages
doi:10.1155/2012/156590

Research Article

Tree-Based Neighbor Discovery in Urban Vehicular Sensor Networks

Heejun Roh and Wonjun Lee

Department of Computer Science and Engineering, Korea University, Seoul 136-701, Republic of Korea

Correspondence should be addressed to Wonjun Lee, wlee@korea.ac.kr

Received 2 September 2011; Accepted 24 October 2011

Academic Editor: Junyoung Heo

In urban vehicular sensor networks, vehicles equipped with onboard sensors monitor some area, and the result can be shared to neighbor vehicles to correct their own sensing data. However, due to the frequent change of vehicle topology compared to the wireless sensor network, it is required for a vehicle to discover neighboring vehicles. Therefore, efficient neighbor discovery algorithm should be designed for vehicular sensor networks. In this paper, two efficient tree-based neighbor discovery algorithms in vehicular sensor networks are proposed and analyzed. After suggesting detailed scenario and its system model, we show that the expected value of neighbor discovery delay has different characteristics depending on neighbor discovery algorithms. An interesting observation of our result is that M-binary tree-based neighbor discovery shows better performance than M-ary tree-based neighbor discovery in the parking lot scenario, which is a counterintuitive result. We analyze why such result appears extensively.

1. Introduction

In recent years, wireless sensor networks (WSNs) [1], especially urban vehicular sensor networks [2], have inspired research and industry interests in various applications such as traffic engineering, environment monitoring, and civic and homeland security. A typical sensor network includes hundreds to thousands of sensor nodes, each of which is equipped with various kinds of sensors. In general, sensor nodes have stronger resource constraints than nodes in typical wireless ad hoc network, for example, battery power, memory, computation capability, and communication technology. However, vehicular sensor networks have different characteristics to typical sensor network. First, there is no battery constraint. Therefore, in vehicular sensor networks, communication between vehicles with less delay is one of the most important issues, compared to typical wireless sensor networks.

Especially, since vehicular sensor networks may have no infrastructure controlling the network environment, it is required for vehicles to autoconfigure the environment for establishing a communication network. For example, if sensor-occupied vehicles arrive at the same area, each vehicle would not know which other vehicles are in its transmission range, implying that the vehicle needs to discover its neighboring vehicles. Since information obtained during the discovery process is required by many network protocols, neighbor discovery of each vehicle in the vehicular sensor network is an important configuration process.

Currently, explicit neighbor discovery may not be performed in typical ad hoc networks [3], a superset of vehicular sensor network. That is, it is assumed that all transmissions are listened to by neighboring nodes in the medium access control layer level. This assumption is not suitable for practical vehicular sensor network implementation, because when many vehicles are in the one-hop transmission range, the delay of neighbor discovery increases dramatically. In addition, the delayed result for neighbor discovery process can be outdated since the neighboring vehicles move too fast to depart the transmission range. Therefore, an efficient neighbor discovery scheme for vehicular sensor networks is required, implying that average delay minimization of neighbor discovery schemes in vehicular sensor networks is an important problem.

Recently, numerous studies on neighbor discovery for wireless networks have been proposed in the literature [4–6]. These studies mainly focus on applications of mobile robot networks, and the IEEE 802.11 network provides robot intercommunication, which is very similar to vehicle-to-vehicle communications in vehicular sensor networks. However, these approaches cannot be applied into the vehicular sensor networks without modification, because if there are many groups of vehicles, the performance of the network may be reduced significantly due to interference and collisions. Furthermore, the network load cannot be controlled by a group of nodes in general [5].

In Santos et al. work [4], an adaptive TDMA protocol for mobile autonomous nodes is proposed. This protocol is an adaptive TDMA protocol with new self-configuration capabilities according to the current number of active group members. This protocol operates over IEEE 802.11 networks in infrastructure and adhoc mode. However, it does not support a dynamic number of nodes. The improved protocol proposed in [5] has dynamic reconfiguration of the TDMA round, which supports changes of the number of nodes. But it is not suitable to assumption, because each node in a group maintains a state machine and the whole member information of the group.

Arai et al. [6] proposed an adaptive reservation-TDMA (AR-TRMA) MAC protocol, which focuses on real-time communication among nodes in a heterogeneous environment by using a reservation mechanism. This protocol supports dynamic time slot allocation schemes during node intercommunication. Furthermore, this protocol considers the packet collision avoidance method in the joining procedure, using the battery voltage level and ID of each node. However, when a packet collision occurs, the protocol uses fixed-size minislots to resolve the collision. This scheme suffers from lack of scalability, because it may not support dynamic situations of vehicular sensor network which include a situation that a large number of nodes and groups move around frequently.

With this motivation, we have studied tree-based neighbor discovery schemes in vehicular sensor networks, which guarantee the discovery of whole neighbors in each leader vehicle. In this study, our objective is to reduce the average delay of tree-based neighbor discovery. According to our study, the delay depends on the tree construction scheme. We focus on M-ary and M-binary tree-based neighbor discovery, because these algorithms have simplicity and optimality, respectively.

In summary, our contributions are twofold. First, we apply tree algorithm variations to neighbor discovery in vehicular sensor networks. To the best of our knowledge, our study is the first work applying tree algorithms to neighbor discovery in vehicular sensor networks. Second, we show that the M-Binary tree-based scheme with optimal M and N is suitable for the neighbor discovery process.

The reminder of this paper proceeds as follows. Our model for sensor-equipped vehicles is described in Section 2. The proposed join schemes are described in Section 3. In Section 4, determining the number of groups in each neighbor discovery process is discussed, and the performance results are also shown. Our conclusion is given in Section 5.

2. System Model

First of all, we explain our system model assumptions. We assume that numerous vehicles are uniformly distributed in a working field (e.g., road), and each vehicle may move freely. Though this assumption is not suitable for some vehicle sensor network applications, many mobile sensor network applications in dense-area monitoring scenario allow this assumption. It is also assumed that the carrier sensing range is twice the transmission range.

In this model, vehicles are classified into leaders and followers. The leader vehicle (LV) is the vehicle managing and controlling a group of vehicles. The other vehicles in the group are called follower vehicles (FVs) or members of the group. A vehicle which is not a follower vehicle and can communicate with a leader vehicle in onehop is called a neighboring vehicle. Note that follower vehicles do not need to identify other vehicles except for the leader vehicle, and a leader vehicle has the number of required vehicles for performing a cooperative sensing task.

A group following a leader vehicle uses TDMA with a channel distinct from the default channel when communication among nongroup members or between a member and nonmember is required. Because the TDMA frame structure guarantees a delay bound and transmission opportunities, each member of the group has a chance to transmit its packet fairly. Figure 1 shows an example of a TDMA frame.

Each of the TDMA frames has several periods: beacon period, joining period, request/leave period, control period, and data period. LV notifies the beginning of a frame by beacon message broadcasting in a beacon period. Acknowledgment of the previous frame is also contained in the beacon message. In a joining period, LV requests its neighbor vehicles to join the group under the number of required vehicles via the proposed schemes. Note that neighboring vehicles which are already joined as follower vehicles do not participate in the joining period. In Figure 1, vehicle A is the follower vehicle which participated in the joining period before, and vehicles B and C are neighboring vehicles which are identified as follower vehicles after the joining period. After joining period, each follower vehicle can request a chance of data transmission or notify the leave itself in request/leave period. LV broadcasts a control message containing the follower vehicles data period schedules during the control period, and each of the follower vehicles can send its own data according to the schedule. This TDMA frame structure can be modified or redesigned given the application scenario and the system requirements.

In this system model, neighbor discovery is performed in the joining period. Depending on the neighbor discovery scheme, the time interval of the joining period may be variable, which causes a significant delay. Therefore, a joining scheme to minimize the total expected delay in the period is required. In this paper, it is called the delay-minimized

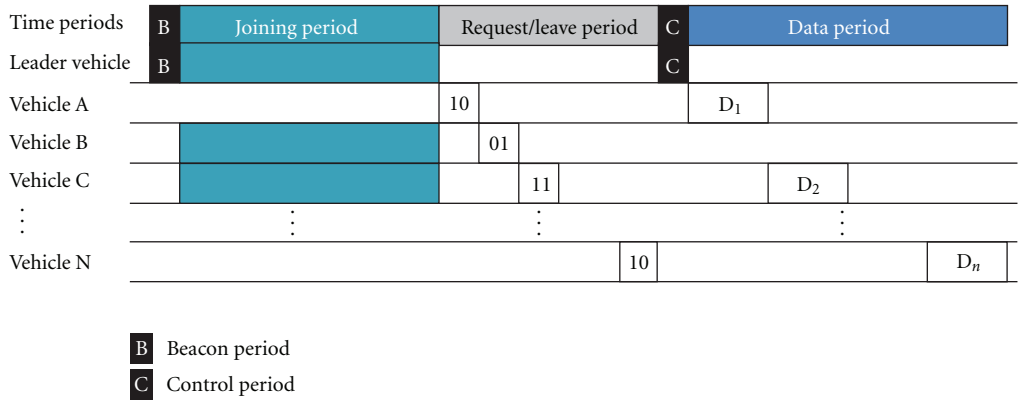


Figure 1: An example of TDMA frame structure.

neighbor discovery (DMND), and in the next section, two tree-based DMNDs are discussed.

Generally, the number of neighboring vehicles of LV affects the performance of DMND in our model. Therefore, LV may estimate and use this information to improve the performance. When LV has no follower vehicles, one of the easiest estimation techniques is to use the field size, the transmission range, and the total number of vehicles in the uniformly distributed working field

$$N_{\text{Neighbor}} = \left\lfloor \frac{N_{\text{Tot}} \cdot (\pi \rho_{\text{TX}}^2)}{S_{\text{Field}}} \right\rfloor, \quad (1)$$

where ρ_{TX} is the transmission range of LV, N_{Tot} is the total number of vehicles in the working field, and S_{Field} is the area of the field.

The other method that can be applied to estimate N_{Neighbor} is found in [7], and this can be utilized via the default channel. But further discussion on the estimation method is not considered in this paper.

3. Tree-Based DMNDs

We mainly focus on minimization of the expected delay in order to identify a required number of vehicles from a greater number of neighbor vehicles. The two kinds of tree-based DMNDs proposed are based on various splitting methods. The first one applies M-ary tree splitting for grouping collided vehicles, while the second one combines M-ary and binary tree splitting methods to improve the associated delay performance. The latter requires an algorithm to decide the value of M affecting the performance of the method, and we will derive the optimal $M(M^*)$ which minimizes the average of total delay. Some notations for the description of the proposed DMNDs are as follows. M is used to denote the number of children nodes (branches) in a tree diagram, and N and k are the numbers of neighbor and required vehicles, respectively.

3.1. M-Ary Tree-Based DMND (MT-DMND). In M-ary tree-based DMND, after LV sends a query, which includes the state information of the previous slot, each neighboring vehicle determines its joining operation using this information. If the state of the previous slot is "C (collision slot)," each of the vehicles which sent an ACK in the slot randomly chooses an integer in the interval $[0, M-1]$ and concatenates the number onto the end of its ID. Then, if its new ID satisfies the requirements of LV, which is included in the query, the vehicle sends an ACK with its new ID. In case that the state is "S (success slot)," the vehicle which sent an ACK with its ID in the previous slot recognizes itself as a follower vehicle of LV. Each of the other neighboring vehicles that did not send an ACK do send one if its ID satisfies the requirements of LV. If the state is "I (Idle slot)" and the ID of a neighboring vehicle satisfies the requirements of LV, the vehicle sends an ACK with its ID.

On the other hand, in M-ary tree-based DMND, the operation of LV is as follows. First, at the beginning slot of a joining period, LV sends a query requesting an anonymous ID. Then LV waits for a time slot to receive an ACK and recognize the state of the time slot. Based on the state, LV sends another query and waits for a time slot. This procedure repeats until LV identifies k follower vehicles. The policy that LV queries the IDs of neighboring vehicles in the procedure is conceptually similar to "depth-first-search (DFS)" algorithm. When LV recognizes that the state of the previous slot is "C," it concatenates 0 to $M-1$ to the respective queried ID and stacks M IDs into the candidate list. Then it pops an element of the list and sends a query with the element. When the state of the previous slot is "S," it identifies the vehicle which sent the ACK in the previous slot as a follower vehicle. Then it pops an element of the list, and sends a query. When the state of the previous slot is "I," it only pops an element of the list and sends a query. Therefore, LV can construct an M-ary tree by the result of M-ary tree-based DMND, as shown in Figure 2.

3.2. M-Binary Tree-Based DMND (M-BT-DMND). M-binary tree-based DMND is a modification of M-ary tree-based DMND designed to reduce excessive collisions and idle slots. It is based on M-binary tree construction, just as 2-ary tree-based DMND is based on M-ary tree construction. An M-binary tree is a tree that each node has at most 2 nodes, except that the root node has at most M nodes.

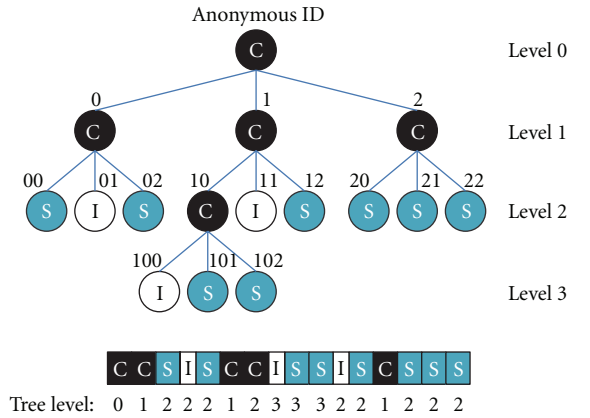


Figure 2: The M-ary tree-based DMND (when $M = 3$).

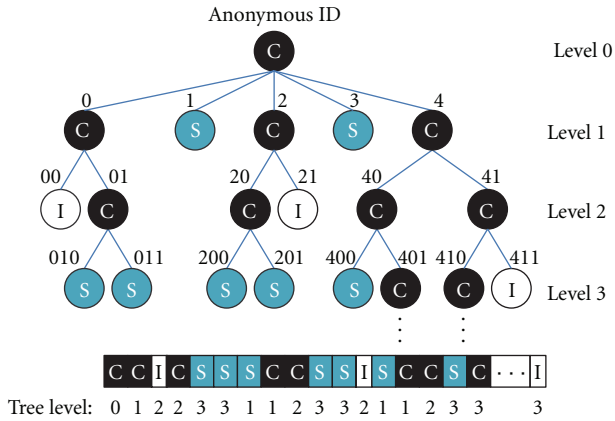


Figure 3: The M-binary tree-based DMND (when $M = 5$).

Therefore, LV and its neighboring vehicles choose either 0 or 1 for concatenation, except when LV sends a query with an anonymous ID. After LV sends a query with an anonymous ID and a collision slot has occurred, LV and its neighboring vehicles choose an integer in the interval $[0, M-1]$. Figure 3 shows a typical example of M-binary tree-based DMND. Note that there is no node with only one child, because the "S" and "I" nodes have no children and the "C" node always has 2 or more children. Though further improvement of this scheme can be achieved based on this observation, we do not consider it further due to the complexity of the analysis.

4. Simulation Results

4.1. Simulation Environments. Some simulation experiments were performed to compare DMNDs. To validate each simulation result, we use the values of the IEEE 802.11 FHSS system parameters listed in Table 1 as in [8]. Note that Short Interframe Space (SIFS) is a fixed-size time period between frame transmission used for IEEE 802.11 nodes to detect the correct channel idle. As comparison targets, p-persistent DMNDs [9] with or without query are also simulated. Note that p is optimally chosen by the simulation with the precision of 0.001. We performed 1000 trials for each scheme, and the delay results are averaged over each scheme for the given (fixed) parameters. In addition, when we use query in a DMND, since success/collision slot needs one query (with 2-bit payloads for 3 slot types), 2 SIFSs, and 1 ACK message (with 32-bit ID), it takes about 532 μs, while the idle slot that includes one query and one *aSlotTime* takes 388 μs. It implies that the ratio of the collision slot time to idle slot time equals to 1.371. We denote this ratio to α. α is one of the most important factors in this simulation result, because α determines the overhead ratio in DMNDs. On the other hand, when we do not use query in a DMND, we use *aSlotTime* for the idle slot and 495 μs for the success/collision solt, according to [9].

Table 1: FHSS System Attributes (Parameters) and Additional Parameters Used to Obtain Numerical Results.

Parameters	Default value
aSlotTime	50 μs
DIFS (DCF Interframe Space)	128 μs
SIFS (Short Interframe Space)	28 μs
MAC Header	272 bits
PHY Header	128 bits
Management frame (w/o payload)	28 bytes
ACK frame	14 bytes
MAC Header	272 bits
MAC Header	272 bits

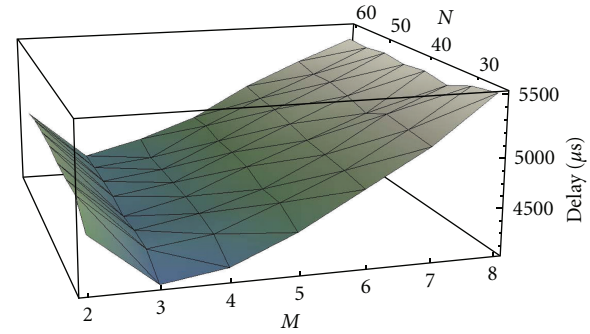


Figure 4: Total average delay of M-ary tree-based DMND (MT-DMND) with fixed k ($k = 25$).

4.2. Optimal Value of M. As mentioned, each of our DMNDs has triple parameters (M, N, k). However, since N is given within the joining period of a frame and k is determined by the task given to LV and the number of follower vehicles identified by LV, M is a tunable parameter in many cases. Therefore, consideration should be given to finding the optimal value of M to minimize the total average delay. Figures 4–7 show the average total delay in case of M-ary and M-binary tree-based DMND with fixed N and k, respectively.

Figure 4 shows the delay of MT-DMND for M and N, where the number of required vehicles for a task, k, is fixed to 25. For a given task and value of M, as shown in the figure,

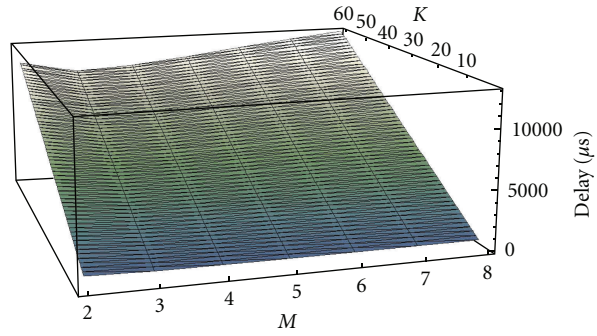


Figure 5: Total average delay of M-ary tree-based DMND (MT-DMND) with fixed N ($N = 60$).

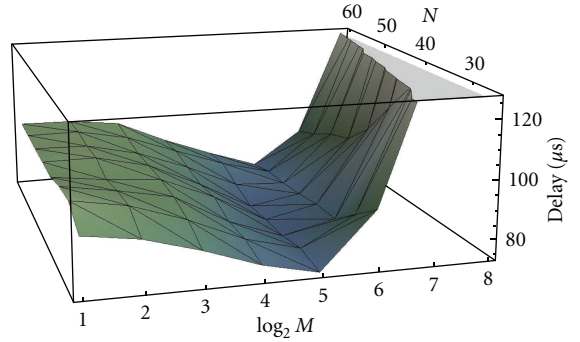


Figure 6: Total average delay of M-binary tree-based DMND (M-BT-DMND) with fixed k ($k = 25$).

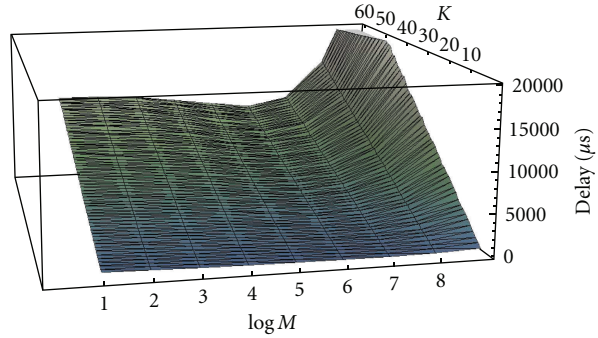


Figure 7: Total average delay of M-Binary tree-based DMND (M-BT-DMND) with fixed N ($N = 60$).

the number of neighboring vehicles does not affect the delay. This result is quite different to the depth-first-search (DFS) of M-ary tree. In DFS problem, each node has a distinct key and when a key is given, the objective of this problem is to search the node with the same key. It is popular for DFS to have a time complexity of $O(|N| + |E|)$ where $|N|$ is the number of nodes in the tree and $|E|$ is the number of edges. Therefore, the total delay of DFS can be considered as a linear function of $|N| + |E|$.

However, in case of MT-DMND, the total delay can be calculated approximately. It is not a linear function of $|N| + |E|$ though the procedure is similar to that of DFS. Given N neighboring vehicles, suppose that LV should identify k follower vehicles with MT-DMND. The total delay of the procedure is denoted by $d_M(N, k)$. We cover only the case that $N \gg k > 1$. In this case, root node is a collision node when this fact is recognized, each vehicle concatenates a random integer $[0, M-1]$ uniformly to its ID. Assuming a uniform random distribution of integers, M subtrees of the root node have almost the same number of nodes. Then, the following equation is approximately satisfied:

$$d_M(N,k) \approx d_{\text{Coll}} + d\left(\frac{N}{M}, k\right), \tag{2}$$

where d_{Coll} is the collision delay in the root node.

Because (2) is a recursion formula, we find a number r such that $N/M^r = k$ ($r \geq 1$),

$$\begin{aligned} d_M(N,k) &\approx r \times d_{\text{Coll}} + d_M(k,k) \\ &= \log_M\left(\frac{N}{k}\right) \times d_{\text{Coll}} + d_M(k,k), \end{aligned} \tag{3}$$

where $d_M(k, k)$ is the total delay of the procedure given k neighboring vehicles and k required follower vehicles.

In this context, for a considerably large k, $\log_M(N/k)$ can be ignored. Therefore, in this case, N has little effect on the total delay of MT-DMND. This is unique characteristic of MT-DMND, which identifies only a fraction of all neighboring vehicles. We can conclude that in MT-DMND, when we choose the value of M, the characteristic of the task is more important than the number of neighboring vehicles.

On the other hand, for a given task (i.e., fixed k), the value of M has a considerable effect on the total delay, as shown in Figure 4. The explanation for this has already been described in the previous section: the number of idle and collision slots is important. If M increases, the number of collision slots decreases. But, if M is very large, the number of idle slots is so large that the delay increases. Therefore, we can determine the optimal value of M by considering the trade-off between the number of collision and idle slots. In the figure, the optimal value of M is 3, since we assume that the value of α in our protocol is 1.371. Note that a different α may lead to a different optimal value of M. Furthermore, it is observed in the figure that when M increases, the effect of the number of idle slots on the total delay is approximately linear.

Figure 5 shows the total average delay of MT-DMND for M and k when the number of neighboring vehicles N is fixed to 60. In this figure, the total delay increases linearly when the number of required vehicles k increases. It shows that if k increases, the respective number of collision and idle slots increases linearly. Especially, the slope of the delay for k is the minimum value in case that M is equal to 3, implying that the optimal value of M is not changed, even though the value of k is changed. Therefore, in M-ary tree-based DMND, the calculation of optimal value of M to minimize delay does not depend on N or k, but depends on the given MAC protocol parameters such as α.

Figures 6 and 7 show the total average delay of M-BT-DMND when k is fixed to 25 and N is fixed to 60, respectively. A comparison of Figures 4 and 6 shows that the

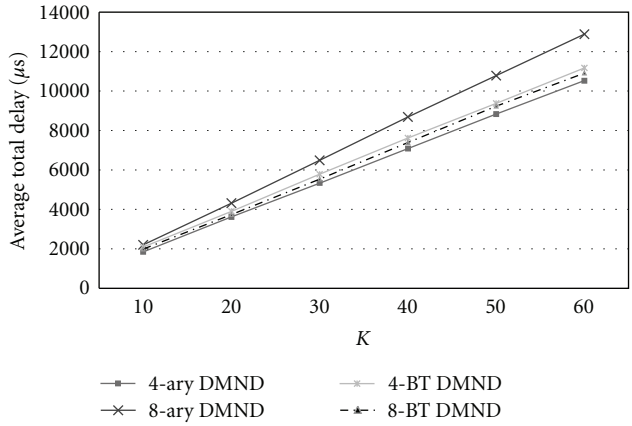


Figure 8: Performance comparison of MT-DMND and M-BT-DMND with $M = 4, 8$.

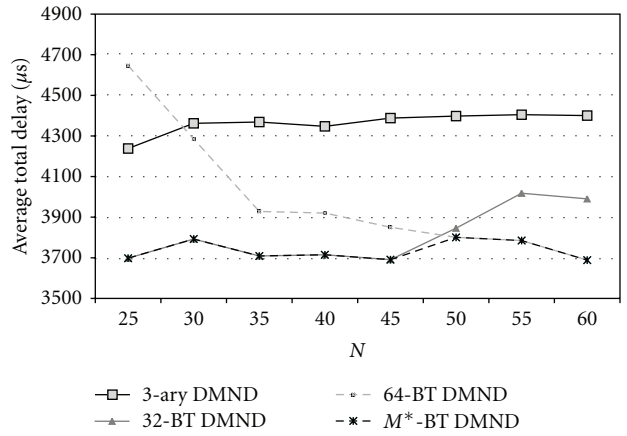


Figure 9: Performance comparison of DMNDs with $k = 25$.

number of neighboring vehicles N has a considerable effect on the optimal value of M in M-BT-DMND, compared to the case of MT-DMND. Furthermore, in many cases, M-BT-DMND has a smaller delay than MT-DMND. Therefore, this characteristic is a motivation of M^*-binary tree-based DMND, which is a dynamic version of M-binary tree-based DMND. In the next section, the details of the scheme are discussed.

4.3. Performance Comparison. In this section, we compare DMNDs. First, Figure 8 shows (N is fixed to 60) the average total delay for k, where $M = 4$ and $M = 8$. In this figure, when M is 8, M-BT-DMND has a smaller delay than MT-DMND, but when M is 4, MT-DMND has a smaller delay than M-BT-DMND. This shows that choosing a suitable value of M has a considerable effect on the performance of each scheme. Especially, it justifies the calculation of optimal value of M.

Figure 9 shows the average total delay for N with fixed k ($k = 25$) when each of 3T-DMND, 32-BT-DMND, 64-BT-DMND, M^*-BT-DMND, p-persistent DMND with query, and p-persistent DMND without query is performed, respectively. M^*-BT-DMND is a dynamic scheme for N that selects the optimal value of M. However, in this simulation, the range of M is given to $\{2, 2^2, 2^3, \ldots, 2^{12}\}$ to reduce its complexity. As shown in this figure, the delay of M^*T-DMND (i.e., 3T-DMND) is generally longer than that of M^*-BT-DMND, and the performance of p-persistent DMND with or without query is worse than tree-based DMNDs. That is, though p is optimally selected, tree-based schemes can show better performance, depending on the system parameters. Furthermore, as we discussed, the optimal value of M in M-BT-DMND depends on N. Therefore, we can conclude that the performance of MT-DMND is stable, while that of M-BT-DMND is dynamic with respect to N. That is if the estimation of N_{Neighbor} is correct, M^*-BT-DMND is more effective than other DMNDs, because it has a smaller delay. But, if the estimation error of N_{Neighbor} is large, MT-DMND may have better performance.

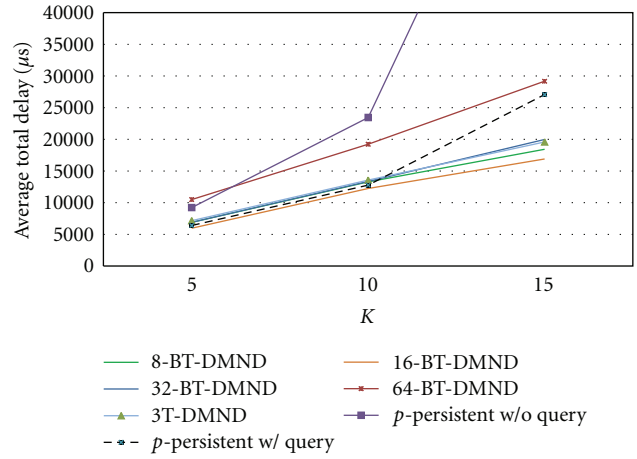


Figure 10: Performance comparison of DMNDs with $N = 15$.

Figure 10 shows the average total delay for k with fixed N ($N = 15$) when each of 3T-DMND, 16-BT-DMND, 32-BT-DMND, 64-BT-DMND, p-persistent DMND with query, and p-persistent DMND without query is performed, respectively. As shown in this figure, when M is similar to N, M-BT-DMND has better performance. However, p-persistent DMND with query may have better performance, when N and k are very small (e.g., $k = 5$). It is similar to the result in [9]. Note that p-persistent DMND without query has the worst performance generally, in both Figure 9 and Figure 10. It is because p-persistent DMND without query experiences too many collision, though p is optimally selected.

5. Conclusion

We have discussed the joining scheme of the TDMA MAC protocol for vehicular sensor networks. This paper proposed two joining schemes for minimizing the total delay of the joining period: MT-DMND and M-BT-DMND. Based on analyses of the proposed DMNDs and p-persistent DMNDs, we found that MT-DMND has better performance when M

is fixed to 3 among MT-DMNDs, while in M-BT-DMND, the optimal value of M depends on the number of neighboring vehicles, N. Furthermore, when we choose M optimally, M-BT-DMND has the best performance than MT-DMND and p-persistent DMNDs. Therefore, when it is hard to estimate the number of neighbors and stability is important, MT-DMND is suitable, while for a small estimation error when quick joining is required, M-BT-DMND with dynamic M calculation is suitable. As a future work, we will study and analyze a scheme to solve the problem that arises from the estimation error.

Acknowledgments

This research was jointly sponsored by MEST, Korea, under WCU (R33-2008-000-10044-0), MEST, Korea under Basic Science Research Program (2011-0012216), MKE/KEIT, Korea under the IT R&D program [KI001810041244, SmartTV 2.0 Software Platform], NRF of Korea under the project of Global Ph.D. Fellowship, and MKE, Korea under ITRC NIPA-2011-(C1090-1121-0008).

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–105, 2002.

[2] U. Lee, B. Zhou, M. Gerla, E. Magistretti, P. Bellavista, and A. Corradi, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *IEEE Wireless Communications*, vol. 13, no. 5, pp. 52–57, 2006.

[3] M. J. McGlynn and S. A. Borbash, "Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks," in *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '01)*, pp. 137–145, Long Beach, Calif, USA, October 2001.

[4] F. Santos, L. Almeida, P. Pedreiras, and T. Facchinetti, "An adaptive TDMA protocol for soft real-time wireless communication among mobile autonomous agents," in *Proceedings of the Workshop on Architectures for Cooperative Embedded Real-Time Systems (WACERTS '04) in Conjunction with the 25th IEEE International Real-Time Systems Symposium (IEEE RTSS '04)*, Lisbon, Portugal, December 2004.

[5] F. Santos, L. Almeida, and L. S. Lopes, "Self-configuration of an adaptive TDMA wireless communication protocol for teams of mobile robots," in *Proceedings of the 13th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA '08)*, pp. 1197–1204, Hamburg, Germany, September 2008.

[6] J. Arai, A. Koyama, and L. Barolli, "Performance analysis of an adaptive medium access control protocol for robot communication," in *Proceedings of the 11th International Conference on Parallel and Distributed Systems Workshops (ICPADS '05)*, pp. 210–216, Fukuoka, Japan, July 2005.

[7] G. Bianchi and I. Tinnirello, "Kalman filter estimation of the number of competing terminals in an IEEE 802.11 network," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '03)*, vol. 2, pp. 844–852, San Francisco, Calif, USA, March 2003.

[8] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[9] K. Kim, H. Roh, and W. Lee, "Minimizing the joining delay for cooperation in mobile robot networks," in *Proceedings of the 2nd IEEE International Conference on Ubiquitous and Future Networks (ICUFN '10)*, pp. 139–144, Jeju Island, Korea, June 2010.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 796426, 10 pages
doi:10.1155/2012/796426

Research Article

A Novel GTS Mechanism for Reliable Multihop Transmission in the IEEE 802.15.4 Network

WoongChul Choi and SeokMin Lee

Department of Computer Science, Kwangwoon University, Seoul 139-701, Republic of Korea

Correspondence should be addressed to WoongChul Choi, wchoi@kw.ac.kr

Received 31 August 2011; Accepted 14 October 2011

Academic Editor: Junyoung Heo

The IEEE 802.15.4 standard provides Guaranteed Time Slot (GTS) mechanism for reliable transmission. GTS mechanism is suitable for transmitting time-sensitive data because it allocates time slots to a specific node. However, the GTS mechanism in the standard can be used only for one-hop communication. This paper proposes and implements a multihop GTS mechanism for reliable transmission in multihop networks. Simulation results using NS-2 show that low delay and high delivery ratio can be achieved using the proposed mechanism.

1. Introduction

Reliable communication in data networking is normally defined to ensure that there is no loss of data, that the packets are in the right order, that the delay is to an acceptable level, and that there is no duplication of packets. All this is to ensure that the data received is consistent, in order.

Many protocols used in wireless communication have mechanism in themselves for reliable communication. Retransmission and sequence number are generally used to guarantee the features of the reliable transmission of no loss of packets and of the packets in the right order, respectively. One example of providing acceptable delay is to allow a certain node to use media exclusively during a certain amount of time. In fact, the IEEE 802.15.4 guarantees acceptable delay using the Guaranteed Time Slot (GTS) mechanism.

The IEEE 802.15.4 standard, widely used in sensor networks, is a standard for low-speed, low-power, low-cost Wireless Personal Area Networks (WPANs). The IEEE 802.15.4 defines the physical layer (PHY) and medium access control (MAC) sublayer specifications for low-rate wireless personal area networks (LR-WPANs), which support simple devices that consume minimal power and typically operate in the personal operating space (POS) of 10 m or less. The nodes of the IEEE 802.15.4 standards basically contend with each other to gain a channel by using the carrier sense multiple access with collision avoidance (CSMA/CA) mechanism. The GTS mechanism is also supported in the beacon-enabled mode in order for a specific node to be able to use a channel exclusively during a certain time interval. Since a node can use a channel without contention by using the GTS, it can be applied to reliable transmission of data with a high priority in a stable way [1]. However, the GTS mechanism guarantees only one-hop communication with a PAN coordinator. Based on an application, a certain technology is required that should be able to transmit important and time-sensitive data reliably in the multihop network environment.

Transmitting a fire alarm signal can be a good example for that. The fire alarm signal can be delivered to a rescuer via a prereserved sensor network or via an alternative IEEE 802.15.4 network. If the paths from the node of sending the fire alarm signal to the sink node are composed of multiple hops, then the transmission of the fire alarm signal cannot be guaranteed using the currently defined GTS mechanism.

This paper proposes and evaluates a novel GTS allocation mechanism for the multihop path from a node to a sink node in the beacon-enabled network of the IEEE 802.15.4 standards.

The rest of the paper is structured as follows. In Section 2, the IEEE 802.15.4 GTS Mechanism standard is studied. Next,

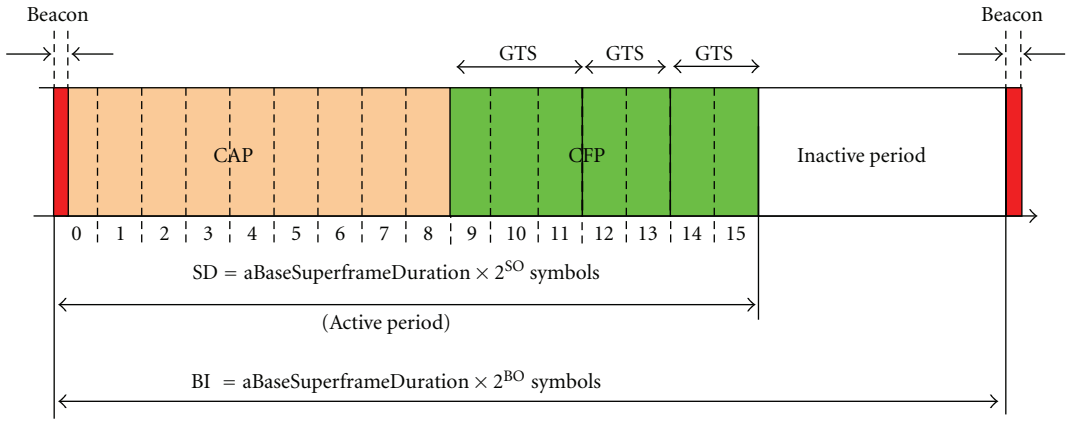


Figure 1: An example of the superframe structure.

in Section 3, the related works and multihop communication using GTS are studied. Then, in Section 4, the proposed multihop GTS mechanism is provided. In Section 5, the simulations are performed using the NS-2 for evaluation. Finally, Section 6 concludes the paper with the closing remarks.

2. Overview of the IEEE 802.15.4

In this section, the necessary parts of the IEEE standards required for our work are overviewed.

2.1. Superframe Structure. The IEEE 802.15.4 standard specifies the physical layer and the MAC sublayer for low-rate wireless personal area network (LR-WPANs) [1]. The MAC protocol in IEEE 802.15.4 can operate both in a beacon-enabled and in a non-beacon-enabled mode. The operation mode is determined by a single central controller termed the PAN coordinator. Basically, media access is contention based (slotted or unslotted CSMA/CA); however, in the beacon-enabled mode, Guaranteed Time Slots can be allocated by the PAN coordinator exclusively to devices willing to transmit time-sensitive data or data requiring specific bandwidth reservation. The allocated device transmits a data frame without checking the channel if it is idle or not; that is, fast transmission is possible because there is no processing delay for the CSMA/CA mechanism. In beacon-enabled mode, beacon frames are periodically sent by the PAN coordinator every Beacon Interval (BI) to identify its PAN, to synchronize associated devices, and to describe the superframe structure (Figure 1), comprising an active period and, optionally, an inactive period. The active period, corresponding to the Superframe Duration (SD), is divided into 16 equally sized time slots, during which data transmission is allowed. SD cannot exceed the BI. Each active period can be further divided into a Contention Access Period (CAP) and an optional Contention Free Period (CFP), composed of GTSs. Slotted CSMA/CA mechanism is used within the CAP. The slotted CSMA/CA mechanism is implemented using unit of time, called backoff period, whose unit length is defined as the *aUnitBackoffPeriod*.

The structure of the superframe is described by the values of the *macBeaconOrder* (BO) and the *macSuperframeOrder* (SO). The value of BO describes the BI at which the coordinator shall transmit its beacon frames. The value of SO describes the length of the active period of the superframe which is referred to as SD. Two values are in $0 \le \mathrm{SO} \le \mathrm{BO} \le 14$. If it is operated as non-beacon-enabled mode and it has not superframe structure. BI and SD can be expressed as follows:

$$\begin{aligned} \mathrm{BI} &= aBaseSuperframeDuration \times 2^{\mathrm{BO}} \text{ symbols}, \\ \mathrm{SD} &= aBaseSuperframeDuration \times 2^{\mathrm{SO}} \text{ symbols}. \end{aligned} \tag{1}$$

The *aBaseSuperframeDuration* constant denotes the minimum length of the superframe when BO equals 0. The standard fixes this duration to 960 symbols (one symbol corresponds to 4 bits, assuming the 2.4 GHz frequency band and 250 kbps of bit rate).

2.2. GTS Mechanism. A GTS allows a device to operate on the channel within a portion of the superframe that is dedicated (on the PAN) exclusively to that device. The concept of a GTS allocation is similar to a Time Division Multiple Access (TDMA). A GTS is allocated only by the PAN coordinator, and it is used only for communications between the PAN coordinator and a device. A single GTS may extend over one or more superframe slots. The PAN coordinator may allocate up to seven GTSs simultaneously, provided there is sufficient capacity in the superframe.

GTS management is undertaken by the PAN coordinator. The PAN coordinator needs to be able to store all the information necessary to manage seven GTSs to facilitate GTS management. For each GTS, the PAN coordinator stores its starting slot, length, direction, and associated device address. For each allocated GTS, the device stores its starting slot, length, and direction. Like the transmission during a CAP, an acknowledgement is occasionally required optionally for a data frame transmission during a GTS.

The device that wants a GTS allocation sends a GTS allocation request command frame to the PAN coordinator in a CAP. Upon receiving a GTS allocation request, the PAN

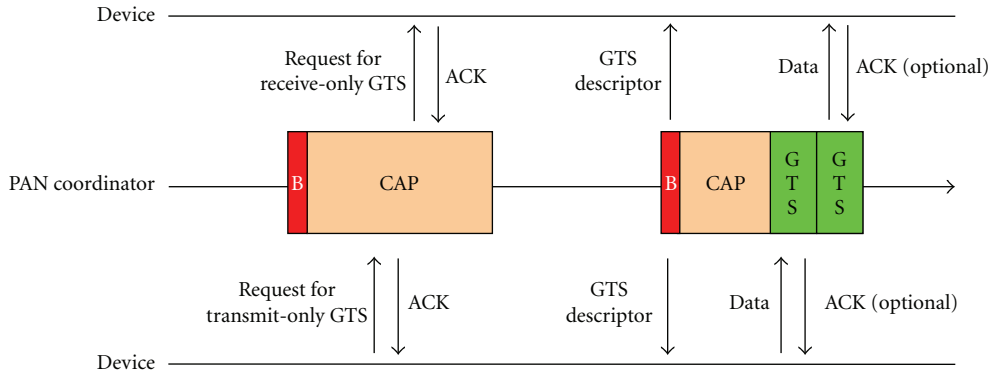


Figure 2: GTS mechanism in the IEEE 802.15.4.

coordinator checks whether there are sufficient resources and, if possible, allocates the requested GTS. When the PAN coordinator determines whether capacity is available for the requested GTS, it generates a GTS descriptor with the requested specifications and the short address of the requesting device. The allocation of the GTS cannot reduce the length of the CAP to less than *aMinCAPLength* (440 symbols). If the GTS was allocated successfully, the PAN coordinator sets the start slot in the GTS descriptor to the superframe slot at which the GTS begins and the length in the GTS descriptor to the length of the GTS [1].

If there was not sufficient capacity to allocate the requested GTS, the start slot is set to 0 and the length to the largest GTS length that can currently be supported. The PAN coordinator then includes this GTS descriptor in its beacon and updates the GTS specification field of the beacon frame accordingly. When a device receives a beacon frame including the GTS descriptor which corresponds to the *macShortAddress* of the device, it judges the success or failure of the GTS allocation by checking whether the GTS starting slot is 0. Note that a device to which a GTS has been allocated can also transmit during the CAP. Figure 2 shows a simple example of the GTS mechanism in the IEEE 802.15.4.

3. Related Works

3.1. GTS Mechanism Improvements. Even though the GTS mechanism of the IEEE 802.15.4 standards provides reliable communication, it hardly meets the requirements of various real-time applications. There have been several research works on its improvement.

Reference [2] proposes a D-GTS (Dynamic GTS) allocation algorithm for both the efficient use of the GTS and the periodic message transmission. D-GTSs are allocated at regular intervals in the CAP. Its length is determined not by an alternative superframe slot unit but by its backoff period unit. References [3, 4] reduce the waste of the channel bandwidth by dividing the length of a GTS further to smaller than a superframe slot. With smaller slots, GTS utilization in the new scheme is better than that in the standard scheme. Reference [5] proposes a method that allocates the GTS with higher priority first after classifying the priorities of the requests of GTSs. This allows GTSs to be allocated first for real-time applications by letting them to have higher priorities. Reference [6] proposes an implicit GTS allocation mechanism where a GTS is shared by several nodes in round-robin way. It overcomes the number of the concurrently allocatable GTSs and the underutilization of GTS bandwidth.

Most of the previous works focus on the reduction of the waste and the efficient use of the GTSs. Those works guarantee the reliability only for one-hop communication as in the GTS mechanism of the IEEE 802.15.4 standard. On the contrary, our work is on guaranteeing the reliability of not only single-hop but also multihop communication.

3.2. Multihop Communication. On a beacon-enabled PAN, a coordinator that is not the PAN coordinator maintains the timing of both the superframe in which its coordinator transmits a beacon (the incoming superframe) and the superframe in which it transmits its own beacon (the outgoing superframe). The relative timing of these superframes is defined by the StartTime parameter. The relationship between incoming and outgoing superframes is illustrated in Figure 3. The BO and SO is equal for all superframes on a PAN. All devices interact with the PAN only during the active period of a superframe [1].

Communication among several PAN coordinators should be possible in the multihop beacon-enabled network. However, there is no specific procedure on how to communicate among coordinators in the IEEE 802.15.4 standard. Reference [7] presents and analyzes nonacknowledged GTS option for interconnection of IEEE 802.15.4 beacon-enabled network cluster using ordinary network nodes as bridge nodes. A bridge node periodically visits source and sink cluster and exchanges data using GTS access as shown in Figure 4. While the communication among the coordinators using different channels is studied in [7], the direct communication using the same channel without the help of a bridge node is studied in our work. The communication among the coordinators using a same channel is shown in Figure 5.

In Figure 5(a), arrow represents association direction. The coordinator at the beginning part of the arrow is child while the coordinator at the ending part is parent. In Figure 5(b), dotted rectangles represent the incoming

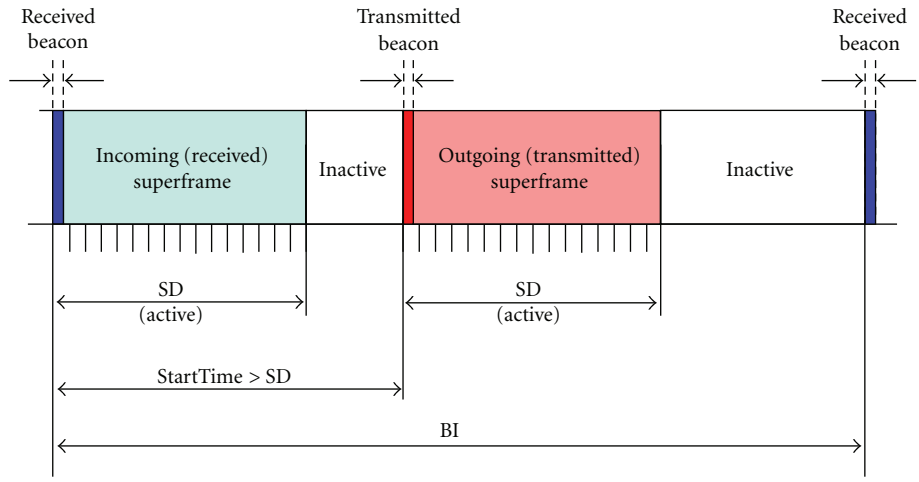


Figure 3: The relationship between incoming and outgoing beacons [1].

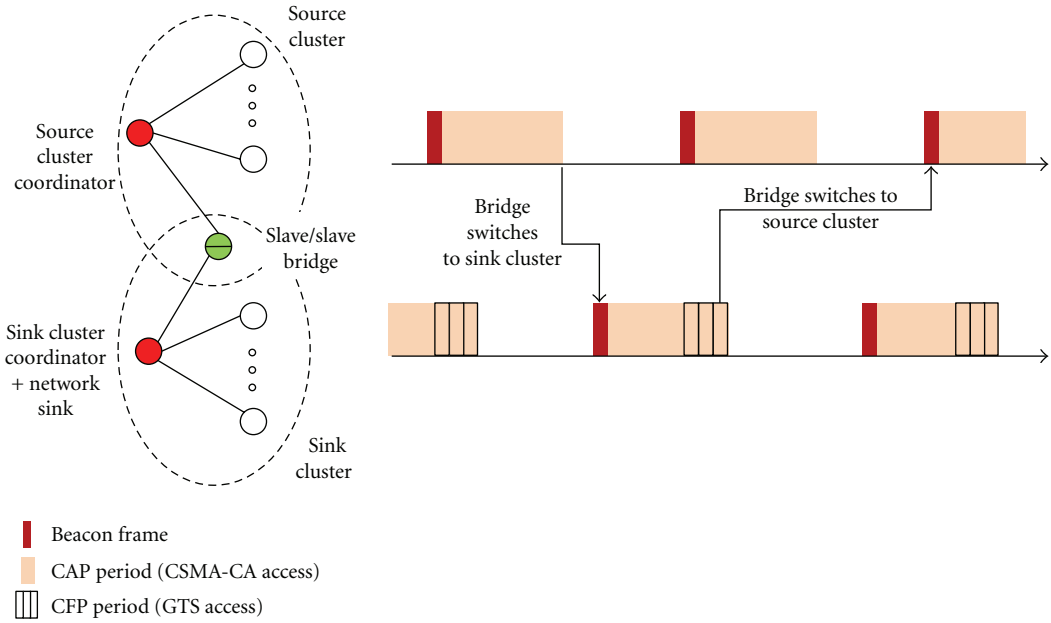


Figure 4: Intercluster communication using a bridge [7].

superframes and other rectangles are the outgoing superframes. The child coordinator transmits data frame in the active period of the parent coordinator. If the parent coordinator has data to transmit to the child coordinator, it indicates in the network beacon that the data are pending. When the child coordinator receives beacon and confirms that there exist data that are pending, it transmits a MAC command, requesting data in the CAP. After the parent coordinator transmits the ACK, it transmits the pending data. This procedure is the same as the one for communication between coordinator and device in the IEEE 802.15.4 standards. In this study, the PAN coordinator is regarded as parent coordinator while device is considered to be child coordinator. Then, as the case in the standards, it is assumed that allocation is made for transmitting the GTS and receiving the GTS, and the CFP performs communication.

Reference [8] analyzed feasibility of IEEE 802.15.4 multihop beacon-enabled network and demonstrated that the 802.15.4 multihop beacon-enabled network was feasible when the BO was larger than 1 and when distribution of coordinators was not too dense under low traffic load. In particular, if the BO was 4 or higher, occurrence of beacon collision did not increase even though the number of coordinators increased, which resulted in low lost synchronization probability. (In the experiment, a coordinator starts the beacon transmission at random time.) Since all coordinators in the same network have the same value of the BO as that of the SO, the beacon interval is equal. However, the active period should not be overlapped in order to avoid

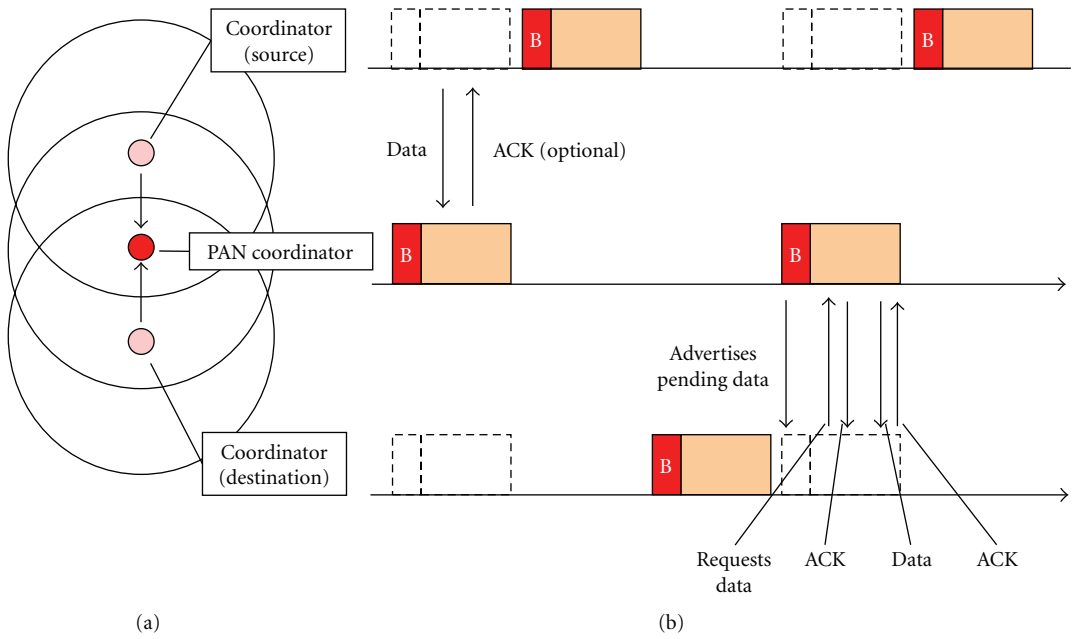


Figure 5: Communication between two coordinators using a same channel.

beacon collision between coordinators. In the IEEE 802.15.4 standards, there is no clear description on how to avoid collision of beacon frames with each other and data frames with each other. Reference [9] introduced several methods to solve the problems on beacon collision, and compared the two most popular ones. This paper does not cover resolving problems on beacon collision.

4. Multihop GTS Mechanism

4.1. BO, SO for Multihop Communication Using the GTS. For prevention of coordinators, which can receive beacons of others, from having their active periods overlapped with one another, it is required to ensure that the value of the BO that determines beacon interval should be larger enough than that of the SO. The values of the BO and the SO determine the number of coordinators that enables intercommunication in a dense state. As shown in the Section 2.1, SD is the same with the active period and both of the BI and the SD are the multiples of *aBaseSuperframeDuration*. If the BO is larger than the SO by as much as 1, the active period is half of the beacon interval. In this case, two active periods in maximum can exist during one BI.

The number N of coordinators available for communication, as coordinators can receive beacons of others and the active period is not overlapped with those of others, can be calculated in the equation following.

$$N = 2^{(\text{BO-SO})}. \quad (2)$$

Since the BO is larger than or equal to the SO, N has the value of exponential multiplication of 2 such as 1, 2, 4, 8, and 16. For example, when the BO is 3 and the SO is 1, N is 4. Therefore, four coordinators can communicate with each other without their active periods overlapped. In our work, the number of the coordinators that can receive the beacons of the others is assumed to be no more than N.

Octets: 2	1	4	1	2
Frame control	Sequence number	Addressing field	Command frame identifier (0x0a)	Frame check sequence
MAC header			MAC payload	MAC footer

Figure 6: Sink notification command frame format.

4.2. Propagation of the Sink Node Information. In the IEEE 802.15.4 standards, it is explained that only PAN coordinator allocates and manages the GTS. And the GTS is used only for communication between the PAN coordinator and a device. However, it is assumed in this study that the coordinator that is not the PAN coordinator allocates the GTS for multihop GTS allocation and that the GTS is used for communication.

A node is required to find out sink node address first if it intends to be allocated with multihop GTS for communication. The sink node address starts to be propagated as the sink node sends sink notification command frame to its parent coordinator. Figure 6 shows the sink notification command frame format. Addressing field contains destination PAN ID and source address. The 16-bit short address is used for address. In order to indicate sink notification command frame, the value of command frame ID field is set at 0x0a that is not used as the standards. When the coordinator receives such frame, it transmits ACK frame.

Afterward, the coordinator maintains the sink node information for allocation of multihop GTS. This information includes sink address, next hop address, hop count, and valid time. Next hop address means the address of the next

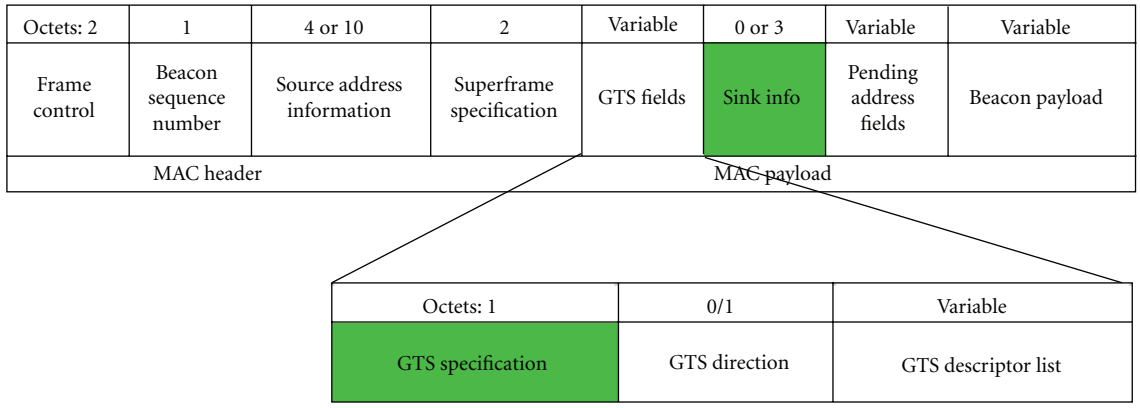


Figure 7: Modified beacon frame format.

hop for delivering data to the sink node. The coordinator that received the sink notification command frame saves the source address of the sink notification command frame to the sink address and the next hop address. Hop count means the hop count up to the sink address. In this case, 1 is saved. Valid time means the valid time of the sink node information. The value of the valid time is set at the constant value of *aMaxSinkInfoValidTime* when the sink node information is received for the first time. The default value for *aMaxSinkInfoValidTime* is set to 4, and the value will become a larger one in the environment where there is much noise or there is possibility for collision.

The coordinator that has the sink node information transmits the beacon frame that includes the sink address and the hop count. And the coordinator decreases the value of the valid time by 1 for every transmission. If the value of the valid time is 0, the coordinator deletes the sink node information and stops transmitting it. The sink node transmits a sink notification command frame at every superframe in order for the sink node information of its parent coordinator not to be deleted. The valid time has two purposes. First, the coordinator stops to transmit the gradually sink node information when the sink node disappears in order to deallocate the allocated GTSs. Second, it prevents the immediate deallocation of the allocated GTSs when a child coordinator cannot temporarily receive the beacon from the parent coordinator.

Figure 7 shows the modified beacon frame format. The sink info field is located next to the GTS fields. The sink info format shown in Figure 8 consists of the sink address of 2-octets and the hop count of 1-octets. The GTS specification field in the beacon frame was changed as shown in Figure 9. The sink info available field is used to indicate that the sink info field is included in the beacon frame. And it uses 1-bit among 4-bit reserved.

The coordinator that received the beacon frame that included the sink node information of other coordinator sets the next hop address of the sink node information at the address of the coordinator that transmitted the beacon frame before saving the setting. The hop count adds 1 to the received value before saving it. The value of valid time is reset at the constant value of *aMaxSinkInfoValidTime*

Bits: 0–15	16–23
Sink address	Hop count

Figure 8: Sink info field format.

Bits: 0–2	3	4–6	7
GTS descriptor count	Sink info available	Reserved	GTS permit

Figure 9: Modified GTS specification field format.

when the sink node information is received from the same source.

In this process, every coordinator sends the beacon with sink node information and every node knows the address of the sink node in the network. Figure 10 shows a simple example of the process.

4.3. Multihop GTS Allocation and Deallocation. After the sink node information is propagated through coordinators in the network, the node that intends to transmit data to the sink node can be allocated with multihop GTS. In order to be allocated with the multihop GTS, the node transmits multihop GTS request command frame to parent coordinator. Figure 11 shows the multihop GTS request command frame format. Addressing field contains destination PAN ID and source address. In order to indicate multihop GTS request command frame, the value of command frame ID field is set at 0x0b that is not used as the standards. GTS characteristics field is identical to the standards, and the sink address field was added. When coordinator receives such frame, it transmits ACK frame.

The coordinator that received the multihop GTS request frame checks if the sink address received from the node is identical to the saved address. And it determines whether or not it allocates the multihop GTS on the same basis as when it determines allocation of the GTS following the

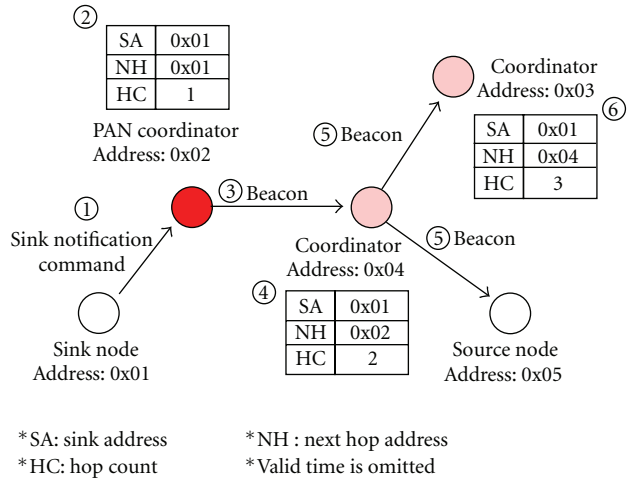


Figure 10: An example of propagation of the sink node information.

Table 1: Simulation parameters.

Item	Value
NS-2 version	2.34
Routing protocol	DumbAgent
Simulation area	40 m × 40 m
Traffic	CBR
Packet size	50 byte
Tx range	9 m
CS range	9 m
TxOptions	Acknowledged transmission, direct transmission
Beacon	Enabled
Beacon order	5
Superframe order	3
Simulation time	300 simulation seconds

standards. If the coordinator can allocate multihop GTS, it informs the node of allocation by using GTS descriptor in a beacon frame. There is no beacon indication that the allocated GTS is for multihop or not. However, the coordinator distinguishes the allocation information of the GTS from that of the multihop GTS before saving it. This is for delivering a data frame from a child node during a multihop GTS to the next hop using the multihop GTS. And then, the coordinator transmits the multihop GTS request command frame to the coordinator that indicates the next hop of the saved sink node information before getting allocated with the multihop GTS. When this procedure is repeated, the parent coordinator of the sink node receives the multihop GTS request command frame in the end. This coordinator, having the sink node information that the next hop is identical to the sink address, allocates the GTS for reception to the sink node. When all of the coordinators on the path from the source node to the sink node can allocate the GTS, allocation of the multihop GTS comes to an end successfully.

When a source node wants the deallocation of the multihop GTS, it transmits the multihop GTS request command frame to the parent coordinator. In that case, the value of the character type of the GTS characteristics field is set to 0 for the deallocation. This complies with the standard. However, the operation after the parent receives this command frame differs. The coordinator deallocates the multihop GTS that has been allocated to the source node and then transmits a multihop GTS request command frame to its parent node as well. Like the allocation, the deallocation of a multihop GTS is sequentially executed from a source node to the parent coordinator of a sink node.

4.4. Communication Using a Multihop GTS. The source node allocated with the multihop GTS transmits data to a parent coordinator in the CFP. When the coordinator receives data during the multihop GTS that has been allocated to a child node, it checks the next hop of the sink node information. Then the coordinator relays the data using the multihop GTS allocated from the parent node.

In this case, the data frame does not go up to the upper layer but directly replaces source address with its own address and destination address with the next hop. If the destination of the received data frame is not the sink node, the data frame is sent to the upper layer for routing. The parent coordinator of the sink node uses the allocated GTS for reception to transmit data to the sink node. Figure 12 shows an example of the GTS mechanism when the sink is 3 hops away from the source node.

5. Evaluation

In this section, the reliability of the multihop GTS is evaluated using NS-2.

5.1. Simulation Environment. The parameters used in the simulation are shown in Table 1. The routing protocol is not used in the simulation to prevent any effect of multihop routing paths on performance. Similarly, the ARP is not enabled. The beacon order is set to 5, and the superframe order is set to 3. When this value is substituted in the expression (2), four coordinators in maximum can communicate with each other without overlapping their active periods when receiving the beacons of each other. Every coordinator including the PAN coordinator sets to the same BO and SO value. It is also assumed that a beacon is transmitted after the active period of their parent coordinator in order for the active period not to be overlapped.

The simulation considers a situation where the time-sensitive data such as fire alarms are recurrently transmitted. Figure 13 shows the topology for the simulation. Node A in Figure 13 is the node that transmits time-sensitive data. The RFDs except Node A periodically transmit normal data frames to sink nodes. All the nodes are separated 5 meters away from each other and can sense the carrier signals of the nodes in diagonals. Every coordinator has four child nodes.

Octets: 2	1	4	1	1	2	2
Frame control	Sequence number	Addressing field	Command frame identifier (0x0b)	GTS characteristics	Sink address	Frame check sequence
MAC header			MAC payload			MAC footer

Figure 11: Multihop GTS request command frame format.

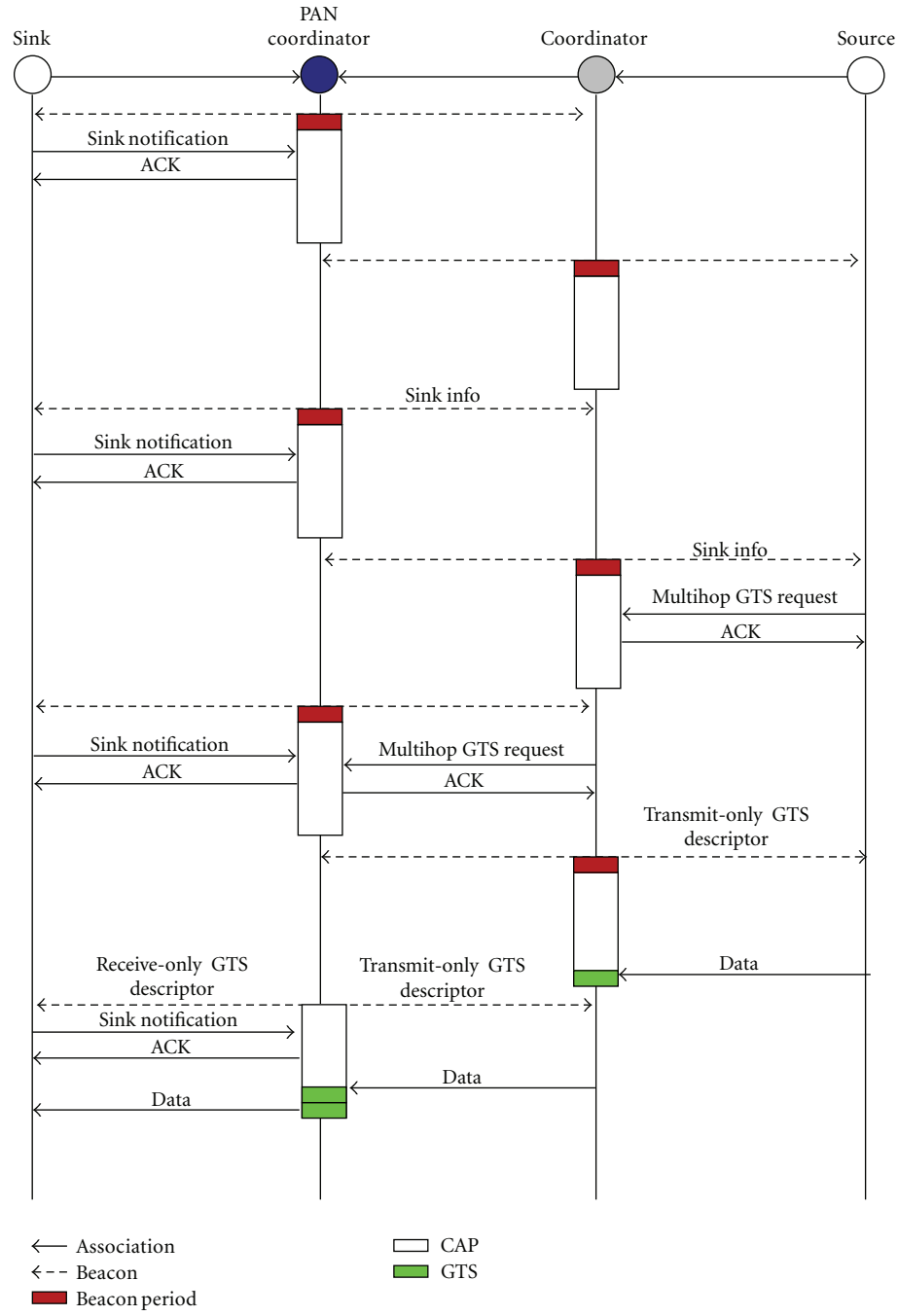


Figure 12: An example of the multihop GTS mechanism.

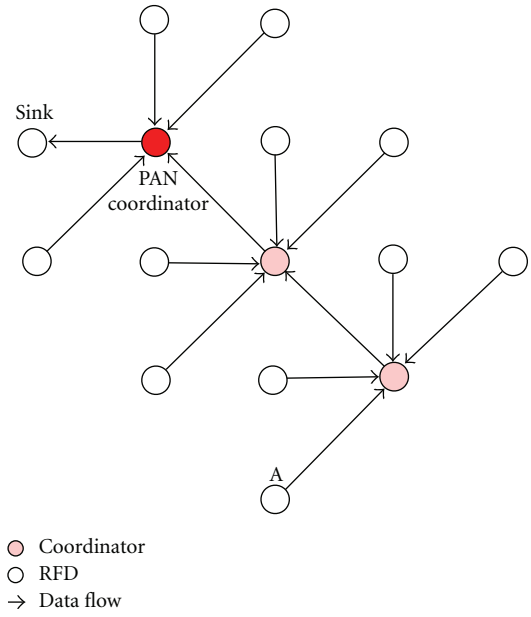


Figure 13: Experiment scenario.

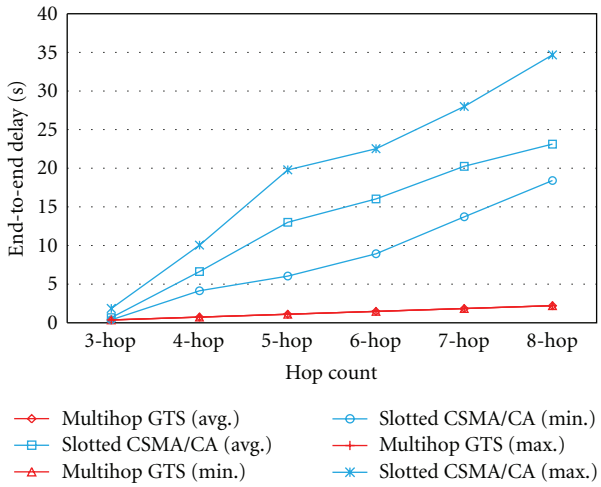


Figure 14: End-to-end delay of hop count.

The simulation compares the cases where node A uses the slotted CSMA/CA and the multihop GTS. In the first simulation, the performance of various hop counts by changing the numbers of the intermediary coordinators is evaluated. In the second simulation, the performance of various number of the contending nodes by changing the number of the child nodes of a coordinator is evaluated. The hop count is fixed as 4.

The simulation runs for 300 simulation seconds. The results after 100 seconds are used for the evaluation, considering the association time and the multihop GTS allocation.

5.2. Simulation Result. Figure 14 shows the end-to-end delay for various hop counts. The result shows that the delay of using a multihop GTS is smaller by about a half for 3-hop

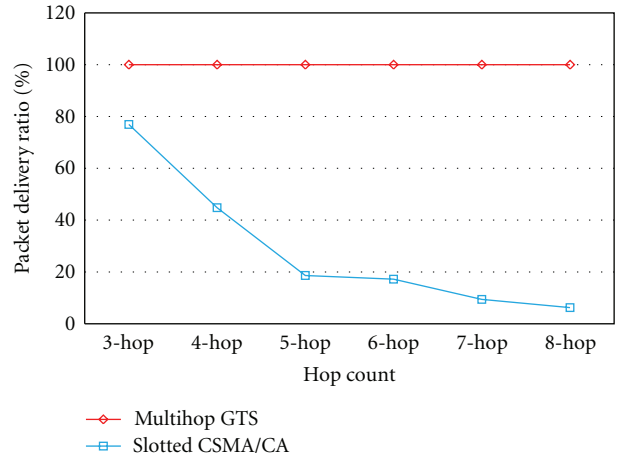


Figure 15: Packet delivery ratio.

case and by about 1/15 for 8-hop case than that of using the slotted CSMA/CA.

The amount of the delay increment corresponds to the length of the beacon interval. This is because the coordinator should wait for the active period of the parent in order to transmit the received data from a child node to the parent node. This applies to the case of using the slotted CSMA/CA in the same way. However, the delay increases much due to the fact that the use of a channel is possible only by contention among the contending nodes.

While the variance of the end-to-end delay is big when using the slotted CSMA/CA, there is not much change when using the multihop GTS. In order to guarantee the acceptable delay, the delay not only should be small, but also should be its variance. Figure 14 shows that the slotted CSMA/CA in multihop networks does not meet such requirement.

Figure 15 shows the packet delivery ratio of various hop counts. While the delivery ratio decreases as the number of the hop counts increases when using the slotted CSMA/CA, all the transmissions are successful when using the multihop GTS. This result shows that the stable transmissions of data are possible if there is no bit error. Contending situations often occur when the slotted CSMA/CA is used in multihop networks. Contentions can cause collisions, which makes reliable data transmissions difficult.

Figure 16 shows the end-to-end delay of various numbers of the child nodes of an intermediary node. When the number of the child nodes of a coordinator increases, so does the number of the contending nodes trying to use a sharing channel. However, this applies only to the slotted CSMA/CA. In case of the multihop GTS, the increment of the number of the nearby nodes does not affect the delay, because a channel can be exclusively allocated to a node for a time interval.

6. Closing Remarks

Previous research on the GTS of the IEEE 802.15.4 has been focusing on increasing utilization and reducing the waste of bandwidth. However, the GTS of the current IEEE standard has a drawback that does not guarantee the reliable

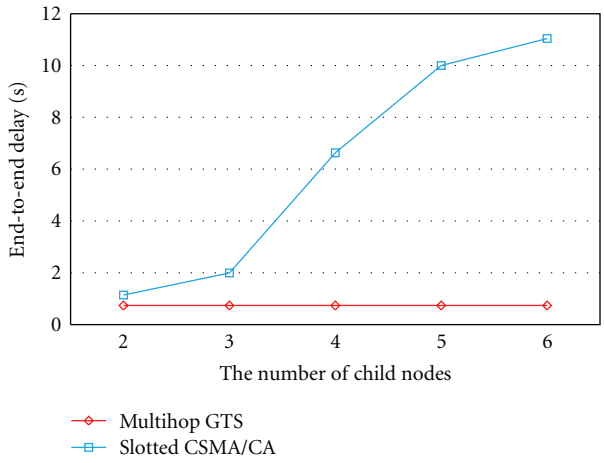


Figure 16: End-to-end delay of the number of child nodes.

transmission in multihop networks. The multihop GTS mechanism introduced in this paper provides the guarantee on the reliable transmission for multihop networks. According to the results using NS-2, low end-to-end delay and high delivery ratio can be checked. The proposed mechanism is especially suitable for delivering time-sensitive data thanks to its feature that the end-to-end delay is low and its variance is small regardless of various numbers of the hop counts.

Acknowledgment

The present research has been conducted by the Research Grant of Kwangwoon University in 2010.

References

[1] IEEE 802.15.4 Standard-2003, *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR WPANs)*, IEEE SA Standards Board, 2003.

[2] S. JunKeun, R. Jeong-Dong, K. SangCheol, K. JinWon, K. HaeYong, and M. PyeongSoo, "A dynamic GTS allocation algorithm in IEEE 802.15.4 for QoS guaranteed real-time applications," in *Proceedings of the IEEE International Symposium on Consumer Electronics (ISCE '07)*, pp. 1–6, June 2007.

[3] H. Yong-Geun, K. Hyoung-Jun, P. Hee-Dong, and K. Do-Hyeon, "Adaptive GTS allocation scheme to support QOS and multiple devices in 802.15.4," in *Proceedings of the 11th International Conference on Advanced Communication Technology (ICACT '09)*, vol. 3, pp. 1697–1702, February 2009.

[4] L. Cheng, X. Zhang, and A. G. Bourgeois, "GTS allocation scheme revisited," *Electronics Letters*, vol. 43, no. 18, pp. 1005–1006, 2007.

[5] Z. Youling, W. Yi, M. Jianhua, J. Junpin, and W. Furong, "A low-latency GTS strategy in IEEE802.15.4 for industrial applications," in *Proceedings of the 2nd International Conference on Future Generation Communication and Networking, (FGCN '08)*, vol. 1, pp. 411–414, December 2008.

[6] K. Anis, A. Mario, T. Eduardo, and C. Andre, "An implicit GTS allocation mechanism in IEEE 802.15.4 for time-sensitive wireless sensor networks: theory and practice," *Real-Time Systems*, vol. 39, no. 1–3, pp. 169–204, 2008.

[7] M. Jelena, "On slave-slave bridging with non-acknowledged GTS access in 802.15.4 beacon enabled networks," in *21st International Conference on Advanced Information Networking and Applications (AINA '07)*, vol. 2, pp. 642–647, May 2007.

[8] H. Jaeyeol, H. K. Wook, J. J. Kim, Y.H. Kim, and Y. H. Shin, "Feasibility analysis and implementation of the IEEE 802.15.4 multi-hop beacon-enabled network," in *Proceedings of the Proceedings of The 15th Joint Conference on Communications and Information (JCCI '05)*, June 2005.

[9] C. V. Berta, D. P. A. Rodolfo, R. Susan, and P. Dirk, "Experimental evaluation of beacon scheduling mechanisms for multihop IEEE 802.15.4 wireless sensor networks," in *Proceedings of the 4th International Conference on Sensor Technologies and Applications (SensorComm '10)*, pp. 226–231, July 2010.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 890238, 6 pages
doi:10.1155/2012/890238

Research Article

A Fast and Reliable Hybrid Data Delivery Protocol for Large-Scale Heterogeneous Sensor Networks

Sanggil Kang,[1] Yujin Lim,[2] Wilmarc Lopez,[1] and Hoyoung Hwang[3]

[1] *Department of Computer Science and Computer Engineering, Inha University, Incheon 420-751, Republic of Korea*
[2] *Department of Information Media, University of Suwon, Hwaseong-si 445-743, Republic of Korea*
[3] *Department of Multimedia Engineering, Hansung University, Seoul 136-792, Republic of Korea*

Correspondence should be addressed to Yujin Lim, yujin@suwon.ac.kr

Received 14 September 2011; Accepted 16 October 2011

Academic Editor: Junyoung Heo

We propose a hybrid data delivery method for the large-scale heterogeneous sensor networks, which is a fast and reliable delivery protocol for the aggregated data from the sinks to the GW. We develop a new multicriteria-ranking algorithm which determines multiple forwarders for each hop by ranking neighbor nodes. To rank the nodes, we compute the fitness value using features for each node such as the received signal strength, nodal delay, and hop distance. We determine the time of sending among forwarders using the waiting time assignment algorithm. In the experimental section, we show that our method outperforms conventional data delivery protocols in terms of data delivery ratio and end-to-end delay.

1. Introduction

Wireless sensor networks (WSNs) are multihop wireless networks which main purpose is to deliver sensed data collected from multiple sensors to one or more data-collecting devices (or sinks). Its inherent power limitation [1], wireless connectivity and vast number of applications have been attracting much attention from a good number of researchers in the recent years [2, 3]. WSNs are used in acquiring information from inaccessible or perilous areas. Sensed data collected at a sensor are delivered to a remote monitoring center through an intermediate system.

Large-scale heterogeneous sensor network, one type of WSN, has a number of heterogeneous wireless nodes deployed over a wide geographical region. The WSN is commonly made up of two subnetworks: a sensor network and a distribution network. Sensors in the sensor network collect or sense data and deliver them to the nearest sink. In general, sensors are static and densely deployed so they can be easily connected to a sink. The distribution network on the other hand is responsible for delivering data gathered from sinks to the remote monitoring center through a gateway (GW). The GW in turn is connected to an external network like the Internet where the remote monitoring center gets the data. Issues on the deployment of nodes and data collection in the sensor network have already been well addressed in a number of literatures [4–7]. In general, data delivery from sensors to the nearest sink is assumed to be loss tolerant due to the sheer amount of correlated data [8]. However, the aggregated data in the distribution network are not. Furthermore, the significant distance between the monitoring center and the GW is also a challenge. But since they are connected through a high-speed wired network, we focus on the data delivery between the sinks and the GW. In addition, it is desirable for the protocol to be relatively fast because a remote monitoring center may need to take appropriate actions based on the information provided by the distribution network.

Hence in this paper, we develop a fast and reliable delivery protocol for the aggregated data from the sinks to the GW. We obtain good data delivery ratio and end-to-end delay by solving the drawbacks of conventional protocols. We also present a new multicriteria ranking paradigm and present a waiting time assignment algorithm. In the experimental section, we show that our protocol outperforms conventional flooding and relaying techniques.

This paper is organized as follows. Section 2 presents the drawbacks of related works. In Section 3, we describe

in detail our proposed protocol followed by the analysis of experimental results in Section 4. We conclude in Section 5.

2. Related Works

In general, wireless data delivery protocols can be divided into two major categories, (1) relaying and (2) flooding approaches [9]. Relay methods send messages once to a single forwarder. Usually, these methods make use of a multi-criteria fitness function to determine the link cost of its neighbors [10, 11]. Such criteria are usually extracted through a network discovery process commonly done by sending control messages. The more the number of control messages sent, the better the decision process of choosing the next forwarder reflects the state of the network. However, too many of these messages might cause data delivery ratio to fall due to increase in the probability of collision. And since packets are sent only once to a single forwarder, the probability of packet lost is also high. Moreover, the delivery time may suffer from the overhead of the network discovery process.

Flooding techniques on the other hand, in their purest sense, broadcast messages epidemically by forwarding the data to all of its neighbors [12]. In flooding techniques, data is delivered relatively fast because there is minimal or no overhead [13]. However, flooding protocols are notorious for causing *broadcast storms* which lowers deliver ratio due to high chance of collision between messages. Usual attempts to solve the problem like RBVT [13, 14], MHVB [12, 15] and RSB [16] assign neighboring nodes a *waiting time* before sending. It has a value commonly dependent on the forward progress given by $d_{SD} - d_{N_iD}$, where d_{SD} is the distance between the sink S and the destination D and d_{N_iD} is the distance between the receiving neighbor N_i and D [12–16]. The best neighbor, through computation of its own waiting time, should in principle have the least value. By this, the best node broadcasts first. The other neighbors cancel their timers and suppress rebroadcasting or reset their timer to a random value [16], after receiving the duplicate broadcast from the winning node. Such technique can be generally referred to as the election approach [14]. The approach has also an advantage of implicitly having multiple backup forwarders.

The following section explains how to solve the drawbacks of both relay and flooding approaches for the WSN's distribution network.

3. Multicriteria Sender-Side Election Protocol

To explain our proposed protocol named multi-criteria sender-side election (MCSS) protocol, we discuss how we model the local network using neighbor information then present how it is used to rank the neighboring nodes. We present how we assign justifiable waiting time to the best k ranking sinks. Lastly, we discuss how the idea fits into the whole delivery protocol.

3.1. Neighbor Table. Instead of building a path from the sink to the GW, our protocol selects its next forwarder on each hop in order to improve data delivery ratio and end-to-end delay. For each hop, the forwarding node ranks its one-hop neighbors with respect to how much they are able to receive the message. We call it the fitness value of a neighboring node. The fitness value reflects the communication environment of the node. To do this, the Neighbor Table (NT) contains three fields: received signal strength (RSS), nodal delay of messages received from the neighbor, and hop distance from the GW.

The RSS of a message measures the signal strength between the sink and the neighbor. The higher the RSS, the better the chance that broadcasts are successful. The nodal delay describes the amount of time it takes for a message to be delivered from the sink to the neighbor. It includes the processing delay, the queuing delay, the transmission delay, and the propagation delay. The less is the nodal delay of messages received, the less is the end-to-end delay. Since packet loss probability in a wireless multi-hop communication environment increases with the number of hops [17], we choose the hop distance between the neighbor and the GW as one of the criteria. To infer the hop distance of each node, we periodically flood a PROBE message in the distribution network.

The sinks update their own NTs for every message received. The three fields of the received messages are averaged using the Exponential Moving Average (EMA). We think of the NT as an ordered set of the ith neighbor vector $\vec{v_i} = \langle c_{i,j} \rangle$, $i = 1, 2, \ldots, n$ and $j = 1, 2, 3$ with $c_{i,1}$ as the RSS, $c_{i,2}$ as the nodal delay, and $c_{i,3}$ as the hop distance of neighbor i from the GW. Each value is updated as follows:

$$\bar{c}_{i,j}(t) = \alpha \bar{c}_{i,j}(t-1) + (1-\alpha) c_{i,j}(t), \tag{1}$$

where $\bar{c}_{i,j}(t)$ and $\bar{c}_{i,j}(t-1)$ are the new and old average values at time t, respectively. $c_{i,j}(t)$ is measured value at time t and the constant α is a smoothing factor.

3.2. Selection of Forwarders. The conventional schemes selecting a single forwarder can increase the chance of data delivery failure because the forwarder can miss the message in dynamic communication environment. Thus, we choose more than one forwarder by ranking neighbor nodes. To rank the nodes, we compute the fitness value using the three fields in NT for each neighbor. Note that the number of forwarders depends on the degrees of connection. In our experiments, we set it the half of the number of sink's neighbors.

In general, the larger is the value of RSS and the smaller are the nodal delay and hop distance, the better is the performance. For the consideration, we inverse the nodal delay and hop distance using

$$c_{i,j} = \begin{cases} \dfrac{1}{\left(c_{i,j} + \lambda\right)}, & j = 2, 3 \\ c_{i,j}, & j = 1, \end{cases} \tag{2}$$

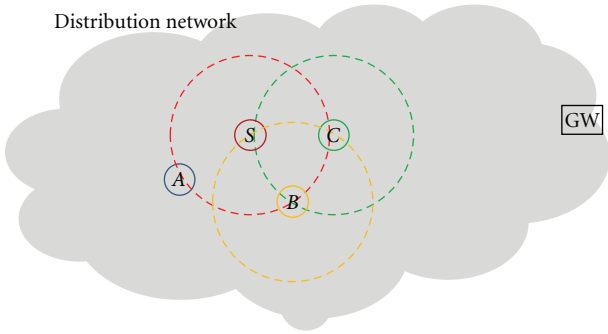


Figure 1: Sink S sends a message to the GW through one of its neighbors A, B or C.

Table 1: Neighbor table of S in Figure 1.

Neighbor	RSS (dBmV/m)	Nodal delay (microseconds)	Hop distance (hops)
A	3	4.5	2
B	2.5	4	1
C	2	4.3	1

Table 2: Inverse values for Table 1 with $\lambda = 1 \times 10^{-9}$.

Neighbor	RSS*	Nodal delay	Hop distance
A	3	0.22	0.5
B	2.5	0.25	1
C	2	0.23	1

where λ is a very small positive number used to avoid division by zero. The fitness value F_i of node i is given as

$$F_i = \sum_{j=1}^{3} \Gamma_j\left(c_{i,j}\right) \cdot w_{i,j}, \tag{3}$$

where $\Gamma_j(c_{i,j})$ is the min-max normalization function of $c_{i,j}$. The weight $w_{i,j}$ can be thought of as the importance of field j to the overall evaluation and it is calculated as

$$w_{i,j} = \frac{\max_{i \in n} c_{i,j}}{\sum_{i \in n} c_{i,j}}. \tag{4}$$

The top k ranked neighbors are chosen as forwarders. To illustrate our method, we use the node setup in Figure 1 and NT in Tables 1–3. Here, the sink S sends a data to the GW and ranks its neighbors A and B. The final ranking makes B the overall best neighbor with a fitness value equal to 0.95. If $k = 1$, then only B is considered as a forwarder.

The responsibility of computing the waiting time is moved from the receiving neighbors to the sink. After choosing the forwarders, the sink assigns each forwarder with a waiting time [18]. We assign the best ranked forwarder with zero waiting time. For the waiting time gaps between adjacently ranked forwarders, we use the nodal delays between each of them.

Table 3: Scores and fitness values of the neighbors of S based on Table 2 with $w_1 = 0.4$, $w_2 = 0.35$, and $w_3 = 0.4$.

Neighbor	RSS Score	Nodal delay Score	Hop distance Score	Fitness value
A	1	0	0	0.4
B	0.5	1	1	0.95
C	0	0.37	1	0.52

Table 4: Waiting time table with $n = 2$ ($\beta = 0$).

Neighbor	Waiting Time
B	0
C	4.15

The waiting time assigned to a forwarder with rank $r = 1, 2, \ldots, k$ is given as

$$(r-1)\frac{\sum_{i=1}^{k-1} c_{i,2}}{k-1} + \beta, \tag{5}$$

where β is a small random number. The waiting time table as shown in Table 4 is appended to and sent together with the data.

3.3. The Delivery Protocol. This section discusses the overall data delivery protocol from a sink to the GW. When a sink collects and aggregates sensed data from the sensors in its service area, it sends the aggregated data to the GW. We refer to the aggregated data as the message. We use the hybrid sender-side election approach discussed above. Additionally, each sink node keeps a record of the messages and sends and forwards in a message table (MT). An entry in the MT contains a message ID and the last known time-to-live (L_TTL). An ACK field is also included to serve as a crossing-out mechanism for successfully acknowledged one-hop broadcasts and is initially set to NO_ACK.

The sink node computes its waiting time and attaches it to the message. The message is broadcasted to its neighbors. The sink enters waiting mode for the message by starting a timer initially set to the mean of its forwarders' nodal delay. And the sink creates an entry in the MT with L_TTL equals to the maximum TTL. If a duplicate message is received during the time and the TTL is less than L_TTL, the sink stops its timer and exits waiting mode. The message is acknowledged by setting ACK to YES_ACK and ignores future duplicates. In other words, the received message is already a rebroadcast from one of its forwarders and has already progressed by one hop. We refer to the condition as the suppression condition. Otherwise, if the timer expires, the message is rebroadcasted. Figure 2 illustrates the data delivery process.

4. Experimental Results and Analysis

In this section, we present our experiments done using the NS-2 simulator [19]. We used the shadowing propagation model [20] with path loss exponent $\beta = 3.0$ and shadowing deviation $\sigma_{\text{dB}} = 5.0$ to model a noisy outdoor environment.

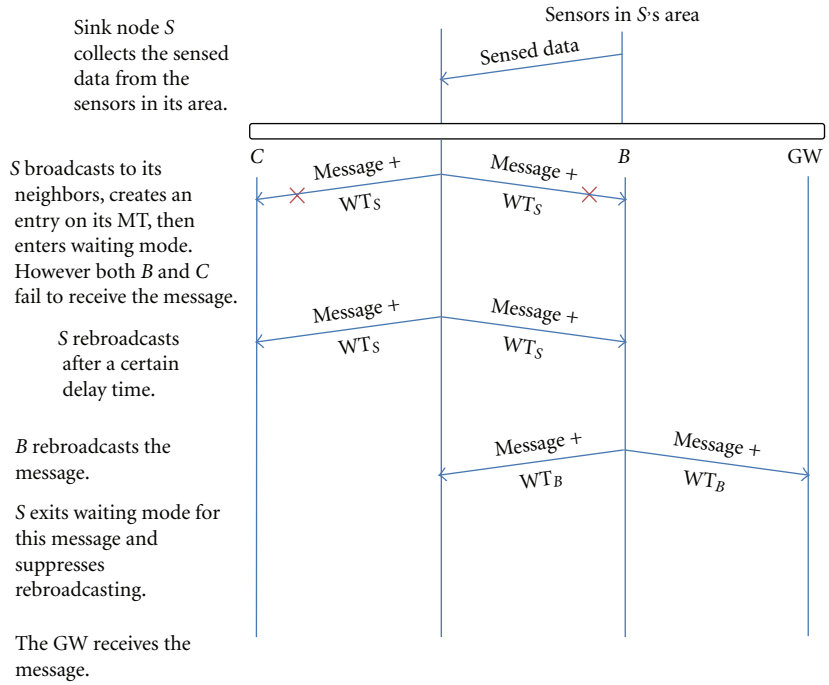


Figure 2: Sequence diagram for an example data delivery scenario for Figure 1.

Sinks are placed in a 2000 m by 2000 m area and a transmission range of each node is 250 m. A sink sends a message to the GW every second at constant bit rate (CBR) for 100 seconds. Additionally, we use IEEE 802.11 [21] as the MAC layer.

We first experimented on both conventional flooding and relay approaches to serve as basis for comparison with our protocol. For the relay approach, we investigated four combinations of criteria setups. In all setups, nodes send a control message every 100 milliseconds. For the flooding approach, we experimented on two weight combinations of the RBVT protocol [13]. This protocol uses forward progress, RSS and optimal transmission area given by

$$f_{\text{trans}}(x) = \begin{cases} x + d_{\text{trans}}, & x \le d_{\text{opt}}, \\ -x + d_{\max}, & x > d_{\text{opt}}, \end{cases} \tag{6}$$

where $d_{\text{opt}} = 100$ m and $d_{\max} = 250$ m are the optimal and the estimated maximum transmission ranges, respectively, and $d_{\text{trans}} = 150$ m is the translation distance.

We evaluated the performance of the protocols through the following performance factors.

(1) Data delivery ratio—the value computed by dividing the number of data packet successfully received at the GW by the total number of data packets sent by sinks.

(2) End-to-end delay—the time needed to forward data from a sink to the GW.

In Figure 3, we can see that the relay approach using only RSS performs best in terms of delivery ratio compared to other conventional setups. Its performance falls with the increase of neighbor density. This is due to collisions from the intense number of control messages. On the other hand, though RBVT does not send control messages, its delivery ratio might have suffered from huge waiting time values being directly equal to the sum of the three criteria.

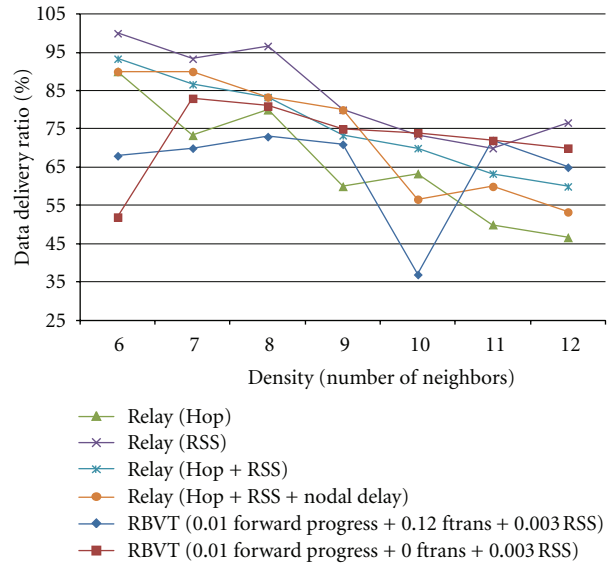


Figure 3: Data delivery ratio of conventional protocols.

In Figure 4, relay approach using only RSS is still the best performing protocol in terms of end-to-end delay. Though RBVT does not have a discovery process, the distance-dependent waiting time makes it perform worst. Additionally, it is also important to note that considering RSS

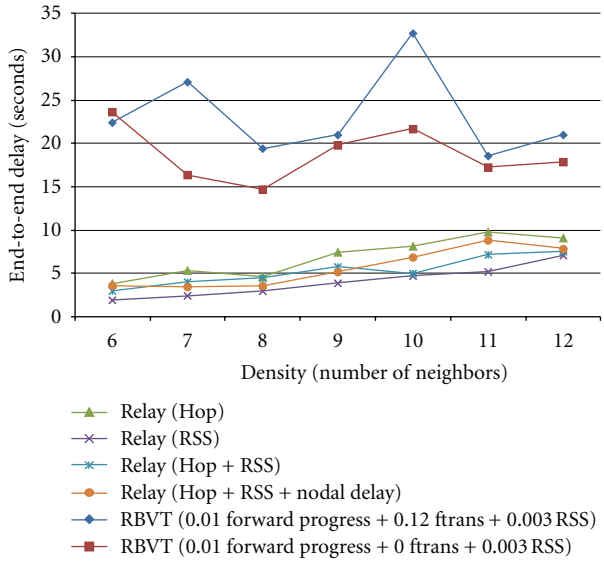


Figure 4: End-to-end delay of conventional protocols.

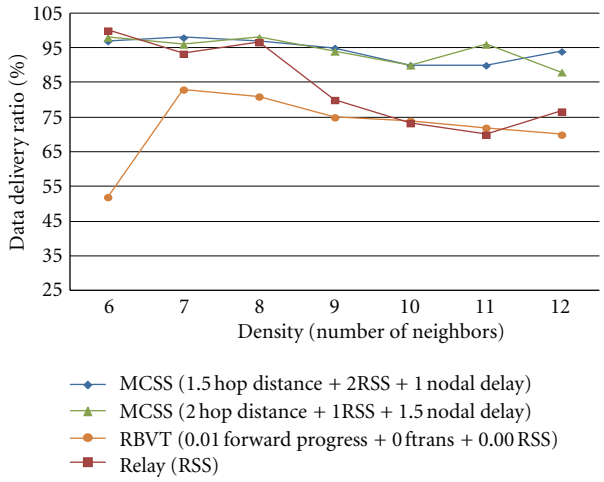


Figure 5: Data delivery ratio of MCSS and conventional protocols.

more makes the overall performance of the relay approach better and somewhat less spiky. This gives us an idea on how important the assignment of weights is.

Using the best setups for both conventional approaches, we compared the performance of our MCSS protocol using two weight combinations. In Figure 5, we can see that our protocol achieves an increase of about 20% in data delivery ratio. It is caused by the reduction of control messages. Additionally, the assignment of zero waiting time value for the best forwarders avoids unnecessary pending for message. Moreover, the improved performance can also be attributed to having the $k - 1$ ranking forwarders as backup resenders.

In Figure 6, we can see that MCSS decreases the end-to-end delay of relay approach and RBVT protocol by about 100% and 500%, respectively. It is due to the waiting time values being well gapped, that is, nodal delay dependent and sender-side assigned. Additionally, note that varying

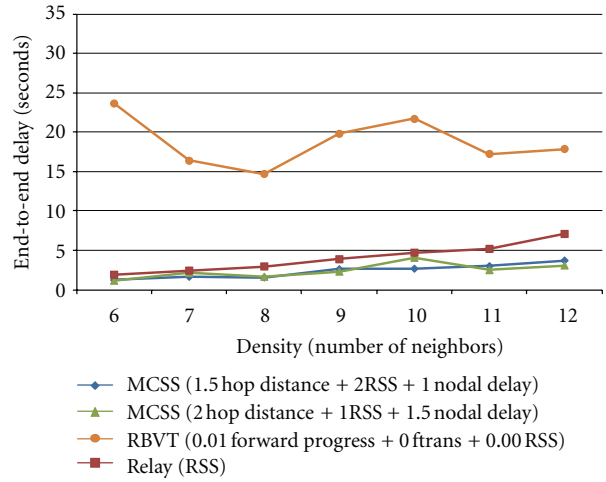


Figure 6: End-to-end delay of MCSS and conventional protocols.

the weights in our protocol has minor effect on its overall performance. Hence MCSS is resilient to poor weight assignments with respect to the protocols experimented upon.

We therefore conclude from the above experimental results that our MCSS protocol improves the quality of data delivery between the sinks and the GW in terms of end-to-end delivery ratio and delay.

5. Conclusion

We developed a fast and reliable data delivery protocol for large-scale heterogeneous sensor networks. The summary and contributions of this paper are as follows.

(1) We identified the data delivery problem in the distribution network of large-scale heterogeneous WSNs.

(2) We analyzed conventional wireless data delivery protocols and enumerated their weaknesses.

(3) We developed a selection algorithm of forwarders.

(4) We developed a sender-side waiting time assignment algorithm for forwarders.

(5) We developed a general hybrid multi-criteria sender-side election protocol which utilizes the two developed ideas.

(6) We applied our approach to the data delivery process between the sinks and the GW in the distribution network.

(7) We showed that our protocol outperforms conventional data delivery protocols in terms of data delivery ratio and end-to-end delay.

In our approach, we used three criteria to model WSN. As a future work, we will do research on more possible criteria and on ways to assign weights dynamically to correspond to a better performance.

Acknowledgment

This work was supported by the GRRC program of Gyeonggi province ([(GRRC SUWON2011-B4), Research on Precision Location Tracking System for Real-Time Situation).

References

[1] H. Chen, C. K. Tse, and J. Feng, "Minimizing effective energy consumption in multi-cluster sensor networks for source extraction," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, Article ID 4801500, pp. 1480–1489, 2009.

[2] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," *IEEE Computer*, vol. 37, no. 8, pp. 41–49, 2004.

[3] K. Romer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 54–61, 2004.

[4] D. Niculescu, "Communication paradigms for sensor networks," *IEEE Communications Magazine*, vol. 43, no. 3, pp. 116–122, 2005.

[5] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, 2004.

[6] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang, "A two-tier data dissemination model for large-scale wireless sensor networks," in *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking (MobiCom '02)*, pp. 148–159, Atlanta, Ga, USA, September 2002.

[7] D. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, no. 4, pp. 419–434, 2005.

[8] Y. Sankarasubramaniam, O.B. Akan, and I.F. Akyildiz, "ESRT: event-to-sink reliable transport in wireless sensor networks," in *Proceedings of the Annual ACM International Conference on Mobile Computing and Networking (MobiCom '03)*, pp. 177–188, Annapolis, Md, USA, June 2003.

[9] P. Cataldi, A. Tomatis, G. Grilli, and M. Gerla, "A novel data dissemination method for vehicular networks with rateless codes," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '09)*, pp. 1–6, April 2009.

[10] B. K. Szymanski and G. G. Chen, "Computing with time: from neural networks to sensor networks," *Computer Journal*, vol. 51, no. 4, pp. 511–522, 2008.

[11] W. Zhao, D. Liu, and Y. Jiang, "Distributed neural network routing algorithm based on global information of wireless sensor network," in *Proceedings of the WRI International Conference on Communications and Mobile Computing (CMC '09)*, pp. 552–555, Yunnan , China, January 2009.

[12] T. Osafune, L. Lin, and M. Lenardi, "Multi-hop vehicular broadcast (MHVB)," in *Proceedings of the 6th International Conference on ITS Telecommunications (ITST '06)*, pp. 757–760, Chengdu, China, June 2006.

[13] J. Nzouonta, N. Rajgure, G. Wang, and C. Borcea, "VANET routing on city roads using real-time vehicular traffic information," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 7, pp. 3609–3626, 2009.

[14] K. Egoh and De Swades, "A multi-criteria receiver-side relay election approach in wireless ad hoc networks," in *Proceedings of the ITST 2006 Military Communications Conference (MILCOM '06)*, Washington, DC, USA, October 2006.

[15] M. O. Cherif, S. M. Secouci, and B. Ducourthial, "How to disseminate vehicular data efficiently in both highway and urban environments?" in *Proceedings of the 6th Annual IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob '10)*, pp. 165–171, Niagara Falls, Canada, October 2010.

[16] J. Lee and W. Chen, "Reliably suppressed broadcasting for Vehicle-to-Vehicle communications," in *Proceedings of the 71st IEEE Vehicular Technology Conference, VTC 2010-Spring*, pp. 1–7, Taipei, Taiwan, May 2010.

[17] S. Toumpis and A. J. Goldsmith, "Capacity regions for wireless ad hoc networks," *IEEE Transactions on Wireless Communications*, vol. 2, no. 4, pp. 736–748, 2003.

[18] W. Wang, F. Xie, and M. Chatterjee, "Small-scale and large-scale routing in vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 9, Article ID 5089413, pp. 5200–5213, 2009.

[19] "The network simulator, ns-2," http://www.isi.edu/nsnam/ns/.

[20] T. S. Rappaport, *Wireless Communications, Principles and Practice*, Prentice Hall, New York, NY, USA, 1996.

[21] IEEE Std. 802.11, "Wireless LAN medium access control (MAC) and physical layer (PHY) specification: higher speed physical layer (PHY) extension in the 2.4 GHz Band," 1999.