IPSJ Journal Vol. 51 No. 3 856–869 (Mar. 2010)

Regular Paper

Cooperative Data Buffering with Mobile Sinks for Wireless Multimedia Sensor Network

Yu Takada,[†1] Masaki Bandai,[†1] Tomoya Kitani[†2] and Takashi Watanabe[†3]

This paper discusses a buffering strategy for a delay-tolerant multimedia sensor network (DTMSN), whose typical application is video surveillance. In DTMSN, a sensor node observes events around it and stores the data in its own buffer memory. All the data is collected to the sink. Sensor nodes have restrictions on buffer memory as well as battery capacity. The entire data size is much larger than a single node's memory size. Thus, developing a strategy for buffering satisfying these restrictions is a critical issue for DTMSN. In this paper, we propose a novel buffering scheme for DTMSN called cooperative buffering (CB). In the proposed CB, the sensor node which has a large amount of data cooperates with its neighbor nodes to buffer the data in a distributed manner. CB uses mobile sinks. The cooperatively buffered data are transmitted directly to the mobile sink when it arrives. After proposing CB, this paper discusses extension for easy collection of the sink, extension for multi source nodes, and some sink mobility strategies of sink mobility. It evaluates the power consumption performance of CB via theoretical formulation and computer simulation. As a result, we show from the results that the proposed CB can handle multimedia data while operating at low-power.

1. Introduction

Recent advances in wireless communication and semiconductor technologies allow making sensor devices that are both small and sophisticated. Wireless sensor networks (WSNs) which consist of such sensor devices are the focus of much recent attention [1)–3)]. One promising feature offered by WSN is cooperation among sensor nodes. In WSNs, sensor nodes observe an event and the sensed data is transmitted to a sink by wireless communication. The sink that collects the information from the sensor node obtains precise information. This detailed information can be used for various applications such as environmental surveillance, home/office security and medical monitoring.

†1 Graduate School of Informatics, Shizuoka University
†2 Division of Global Research Leaders, Shizuoka University
†3 Graduate School of Science and Technology, Shizuoka University

In general, sensor nodes should be miniaturized and have limits on their resources that may include for example processing performance, communication functions, and memory capacity (buffer size). In addition, the power saving of sensor nodes is also a critical issue. Due to these various constraints, conventional WSNs can only handle small size of data such as temperature, humidity, pressure and fluctuation. Extremely small and energy-efficient cameras are recently being developed that can be incorporated into sensor nodes [4)]. These kinds of small cameras allow us to handle multimedia data such as images and video in WSNs. In Refs. 5), 6), video sensor networking technologies with low power hardware platform have been presented. The architecture of the WSN depends on its application, and should be designed for a specific application. Delay tolerant network (DTN) is a promising applications of multimedia sensor networks [7)]. There are various definitions of DTN. The IETF research group that studies DTN is currently discussing how to interconnect highly heterogeneous networks together even if end-to-end connectivity may never be available. In this paper, we target just delay tolerant applications. In DTN, delay performance is not required strictly, while other performance metrics such as network connectivity and energy efficiency are required.

This paper focuses on delay-tolerant multimedia sensor networks (DTMSNs). A typical example of a DTMSN is a surveillance camera. A scenario of the surveillance camera is the following:

(1) Sensor nodes are distributed in a target area.
(2) If a sensor node detects an event, the sensor node records the event by its camera for pre-defined period of time.
(3) A sink is moving in the area, and when the sink comes near the sensor node, the sensor node transmits the recorded video data to the sink directly.

In DTMSNs, the sensor node needs to wait for the sink's arrival. Therefore, sensor nodes need to store all the recorded data in their own memory (buffer). Even though the recorded period is short, the data size is relatively large for the node's buffer. Buffering large quantity of data at a sensor node is a critical issue in DTMSNs.

In this paper, we propose a novel buffering scheme for DTMSNs, called cooperative buffering (CB). In the proposed CB, the sensor node that records a large

amount of video data asks its neighbor nodes to cooperate in jointly buffering the data. CB is used with mobile sink such as MULEs [8]. The buffered data are transmitted directly to the sink when the sink arrives. Sensor nodes can handle large amounts of video data without having a large buffer. We evaluate the power consumption performance of the proposed CB via theoretical formulation and computer simulation. As a result, we show that the proposed CB can handle multimedia data, provide low-power operation.

Our motivation is cooperating with other neighbor nodes, and not having to add or create other nodes just for cooperative buffering. For example, CB can store 5 Gbytes data cooperatively with four neighbor nodes if a node has 1 Gbytes memory. On the other hand, if a node has only a 1-chip microcontroller including 10 Kbytes internal memory, a source node and its nine cooperative nodes can store 100 Kbytes without any external memory, which makes them effective for miniaturizing the nodes. In this paper, we focus on WSNs, especially DTMSNs. The CB concept is however useful for storing vast quantities of data under various memory size conditions.

2. Related Work

The following three methods are used in WSN to transmit large quantities of data from sensor nodes to a sink.

(1) Multi-hop transmission,
(2) Mobile-node method, and
(3) Mobile-sink method.

In (1) the multi-hop transmission method, nodes transmit data to a sink via multiple nodes. This method is simple. However, the power consumption in relay nodes increases.

In (2) mobile-node method, nodes carry the sensed data to the sink physically. Each mobile node can move by itself. In Ref. 9), a network architecture where mobile nodes carry data is presented. In this network, mobile nodes can enlarge the sensing area and to cover an area that is no longer functional due to node damage or low battery power. In the method, the power consumption of not only the wireless communication but also the movement of nodes should be taken into consideration.

In (3) mobile-sink method, mobile sinks move or visit sensor nodes and collect sensed data directly [8]. In this method, each sensor node buffers the sensed data, and transmits the data to a sink when it comes in proximity.

In general, sinks have less restriction in terms of power consumption than sensor nodes. Therefore, the power consumption of sink movement is negligible. In addition, sink movement keeps power consumption in sensor nodes low because the approach of the sink to the sensor node reduces the number of relay nodes between a sink and a sensor node. Based on these considerations, this paper focuses on (3) mobile-sink method because of its effectiveness for DTMSNs.

Moreover, the buffer size of a sensor node is restricted. Therefore, buffer overflow occurs when a sensor node tries to keep large amount of data. To solve this issue, this paper proposes the CB, where some sensor node cooperate in retaining a large amount of data.

3. Cooperative Buffering with Mobile Sinks

This paper proposes a novel buffering method called cooperative buffering (CB) with Mobile Sinks. The proposed CB is used with a mobile-sink data collection method. In CB, some sensor nodes with a small buffer size retain a large amount of data cooperatively in a concentric manner. Therefore, the WSN can handle large amount of data without preparing large buffers in each sensor node. Moreover, sink mobility reduces power consumption in the sensor node.

3.1 Cooperative Buffering

In CB, sensor nodes with small buffers cooperate with each other to retain large quantities of data. A sensor node that generates a large amount of data is defined as *Source node*. At first, the source node buffers the data in its own buffer. When the generated data size exceeds the total buffer size of one-hop neighbor nodes, the source node requests buffering data to two-hop neighbor nodes. If the generated data size exceeds the total buffer size of two-hop neighbor nodes, the source node requests buffering data to three-hop neighbor nodes. Likewise, the source node requests buffering data from nearer neighbors in a concentric manner originating from the source node.

In general, sensor nodes are distributed in a wide area. In addition, the number of moving sinks is limited. Therefore, sinks are not always near the source node;

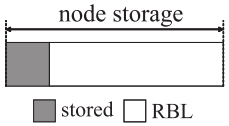


Fig. 1 Buffer memory of node.

it takes time for sink's arrival even if a source node calls the sink just after generating a data. During the waiting time for sink's arrival, the source node cannot retain the generated data without CB because the node buffer space is limited. Therefore, cooperative buffering is therefore needed to retain large amounts of data until the sink arrives.

We describe the usage of buffer memory in Section 3.1.1, and explain how to make a neighbor table in each sensor node in Section 3.1.2. Finally, we show the detailed operation of CB in Section 3.1.3.

3.1.1 Usage of Buffer Memory

We assume that each sensor node has a constant amount of buffer memory capacity. In the buffer memory, a sensor node buffers its own sensing data or cooperatively buffers data requested from a neighbor source node. The buffered data is kept until a sink arrives. An example of buffer memory usage is shown in **Fig. 1**.

As shown in Fig. 1, we define the total node storage minus the stored data as the remaining buffer level (RBL). We assume that each node has another buffer for data relays. The size of the buffer for data relays is just several packets; this is sufficiently smaller than total node storage.

3.1.2 Neighbor Table

Sensor nodes exchange Hello packets between neighbor sensor nodes periodically. Each sensor node makes a neighbor table based on the received Hello packets. The neighbor table has an entry for each neighbor node. This entry is made for each source node. Each Hello packet includes the following information:

- Node_ID: i
- Hop Count (HC): hc_i
- Remaining Buffer Level (RBL): rb_i

Node_ID is the node identification; RBL is the available amount of buffer memory via the node; HC is the number of hops from the source node to the node with some available buffer memory.

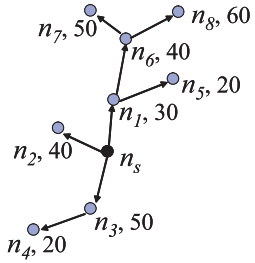


Fig. 2 Example operation of CB.

Table 1 Neighbor table of n_6 in Fig. 2.

Node_ID i	*RBL* r_i	*HC* hc_i
1	30	1
5	20	2
6	40	2
7	50	3
8	60	3

The algorithm for making a neighbor table of a sensor node is shown in the following: rb' denotes the currently remaining buffer memory via the sensor node; hc' denotes current number of hop counts from the source node to nodes which can buffer data via the node.

Algorithm for making a neighbor table

Step 1. For each neighbor node i, the node initializes $hc_i = \infty$, $rb_i = 0$.

Step 2. The node transmits a Hello packet with own Node_ID, rb' and hc'.

Step 3. Receiving the Hello packets from all neighbor nodes, the node makes/renews each entries of its neighbor table.

Step 4. If $rb' > 0$, this turn ends. If $rb' = 0$, the node finds an entry with $hc > hc'$ and smallest hc, and puts $hc' \leftarrow hc$. In addition, it puts the summation of rb_is of all entries with $hc_i = hc'$ to rb'. For example, in **Fig. 2** and **Table 1**, when the remaining buffer of n_6 is 0, $rb_6 \leftarrow rb_7 + rb_8$. Return to Step 2.

When a source node detects generation of a large amount of data, it determines within how many hop counts to request cooperative buffering with neighbors. The number of hop counts is defined as H_{MAX}, which is a pre-defined system parameter. Setting H_{MAX} prevents propagation of unneeded data packets to nodes that are not part of the cooperative buffering. How to determine H_{MAX} is our future work. After setting H_{MAX}, the source node broadcasts a CB join request packet to all neighbors within H_{MAX}-hop from the source node. When a node receives a CB join request, the node sets the hop count into the HC field

of its neighbor table and increments the hop count of the CB join request and broadcasts it. A node receiving the CB join request will then know the distance from the source node due to the included hop count. The initial value of the hop count is 0. If a node receives a CB join request with hop count H_{MAX}, the node does not further broadcast it.

3.1.3 Detailed Operation of CB

When a large amount of data generates at a node, the node initiates cooperative buffering based on the information in its neighbor table. In CB, the source node cooperates with as near a neighbor node as possible to reduce power consumption during multi-hop transmissions. Therefore, the source node selects a node with maximum RBL, and transmits data to the node.

The operation of a source node n_s is as follows: Here, the size of the generated data is D.

CB algorithm at a source node n_s

Step 1. Initialize the generated data size $d \leftarrow D$ and set current hop count $h \leftarrow 0$.

Step 2. Select a node n_a $(i = a)$ with maximum RBL among the nodes with $HC = h$ in the neighbor table. If there are no nodes with $HC = h$ in the neighbor table or $rb_a = 0$, then h is incremented, i.e., $h \leftarrow h+1$. If $h > H_{\mathrm{MAX}}$, then the operation ends because the amount of the generated data are larger than the total available buffer capacity of all neighbor nodes, that is buffer overflow.

Step 3. If n_a is the node itself then buffers the data in its own buffer. Otherwise, the node requests a CB request packet with h and d to node n_a, and transmits the data via multi-hop communications. After that, the node renews its neighbor table according to the Hello packet from node n_a.

Step 4. Renew $d \leftarrow d - rb_a$. If $d <= 0$, this turn ends. If $d > 0$, return to Step 2.

Next, we show the operation of a node which receives a CB request packet with h and d.

CB algorithm at general nodes

Step 1. Select a node n_b $(i = b)$ with maximum RBL among the nodes with h. If that node is itself n_b then go to Step 2. Otherwise, go to Step 3.

Step 2. The node communicates with the source node, and receives the data from rb_b.
If $rb_b \geq d$, $rb_b \leftarrow rb_b - d$. Otherwise, find a node with $hc > h$ and minimum hc in the neighbor table and puts $hc_b = hc$, and places the summation of rb_i of the entries with $hc_i = hc$ to rb_b.
According to hc_b and rb_b, it makes a Hello packet and transmits it to the neighbors. Go to Step 4.

Step 3. Send a CB request packet with h and d to node n_b, and transmit data via multi-hop communications. After that, the node renews its neighbor table according to the Hello packet from node n_b.

Step 4. Renew $d \leftarrow d - rb_b$. If $d <= 0$, this turn ends. If $d > 0$, return to Step 1.

Figure 2 illustrates a sample operation of CB. We assume that the neighbor tables of all nodes have already been completed.

As shown in Fig. 2, there are a source node n_s, and the other nodes n_1 to n_8. The value at the right-hand of each node ID shows the available buffer capacity of each node. At first, the source node n_s compares the RBLs of its neighbor nodes n_1 to n_3. In this example, since n_3 has the maximum RBL, n_s transmits data to n_3 first. After completing the transmission to n_3, n_3 informs n_s of the available buffer capacity of n_4. Next, n_s transmits data to n_2, which has the second largest RBL. After completing the transmission to n_2, n_2 sends a Hello packet with $rb_2 = 0$ to n_s because n_2 does not have any data to transmit. After that, n_s transmits data to n_1. After completing the transmission to n_1, n_1 informs n_s of the total capacity of n_5 and n_6. Next, n_s transmits data to n_1, whose neighbors have the maximum capacity. n_1 buffers the received data in its own buffer memory, and forwards data to the neighbor node with the maximum available buffer capacity. The source node continues the same procedure until completing buffering of all the generated data with all the nodes within H_{MAX} hops from the source node. If the source node cooperates with all the nodes within H_{MAX} hops from the source node and still has data to transmit then an overflow occurs.

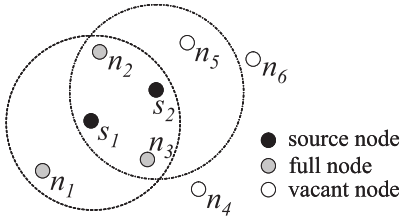


Fig. 3 Example of multiple source nodes.

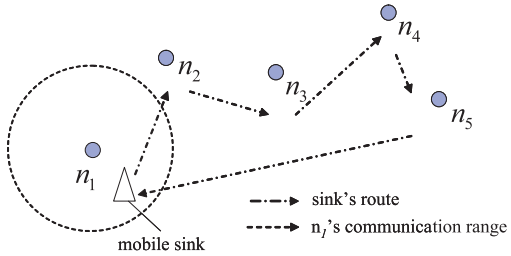


Fig. 4 Example of cyclic visit.

3.1.4 Data Collection by Sink

A sink transmits a notification of sink arrival packet when the sink arrives at the transmission range of a node. A node receiving a notification of sink arrival packet transmits the buffered data to the sink.

3.1.5 Multiple Source Nodes

When some source nodes transmit data to a sensor node, the node buffers the data from earlier requesting source node first.

Figure 3 is an example in which s_1 and s_2 become source nodes simultaneously. In Fig. 3, we assume that s_1 has already cooperated with neighbors and n_2 has been a source node after that. In this case, s_2 can cooperate only with n_5 among one-hop neighbors. If s_2 still has data to transmit after completion to transmit to n_5, then s_2 has to transmit data to n_4 or n_6 via multi-hop communications. In this case, s_2 has to cooperate with far nodes. As a result, power consumption increases. As solutions to reduce the power consumption, we consider the following two methods: overwriting and exclusive buffering. As an example of the overwriting method, data are prioritized by generated time or its importance, and data with low priority is overwritten. Appropriate priorities should be selected for the application. On the other hand, partitioning buffer is one of the exclusive buffering methods. In the partitioning buffer, a buffer memory is partitioned into pre-defined areas. The number of areas is the same as the maximum number of acceptable source nodes. The partitioning buffer ensures the buffer memory for some source nodes, and the node can buffer data from all these nodes. However, the buffer capacity for a source node decreases in the partitioning buffer. Therefore, the partitioning buffer might incur larger power consumption than the overwriting. Some applications might not know the maximum number of source nodes beforehand. Selecting the correct number of partitioned areas is therefore important. Thus, how to choose the number of partitioned areas is important.

We assume a situation where sensor nodes are distributed in a wide area and an event may not occur so frequently. Therefore, we may not need to consider the case where more than two source nodes exist. However, more than two events could occur. In that situation, in fact, all data cannot be buffered because of buffer overflow. However, the conventional multi-hop method also cannot collect all the generated data in that situation. The CB differs from the conventional method in this respect.

3.2 Sink Movement

We consider four sink movements: cyclic visit, random movement, regular movement on route, and on-demand movement.

3.2.1 Cyclic Visit

Cyclic visit is the method where a sink visits all nodes in a cyclic manner. Since the sink visits all nodes, each node has a definite opportunity to transmit data to the sink. When the number of nodes is small, the interval of sink arrivals becomes short. Therefore, buffer overflow can be avoided. However, when the number of nodes is large, then buffer overflow might frequently occur.

Figure 4 illustrates an example of cyclic visit. As shown in Fig. 4, there are five nodes in a network. Figure 4 shows a case where a sink near n_1 moves to near n_2, n_3, n_4 and n_5, sequentially. After visiting n_5, the sink moves to near n_1.

3.2.2 Random Movement

Random movement is the method where a sink moves in a random manner

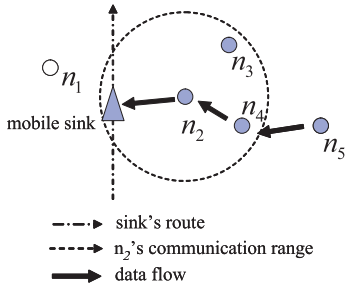


Fig. 5 Example of when multi-hop communication is needed.

and collects data from nodes. The sink can collect data without positions of nodes. However, the interval of sink arrivals at a node is not constant because of the random movement. Therefore, some nodes could have little opportunity to transmit data to the sink and some data could need longer time until received by the sink.

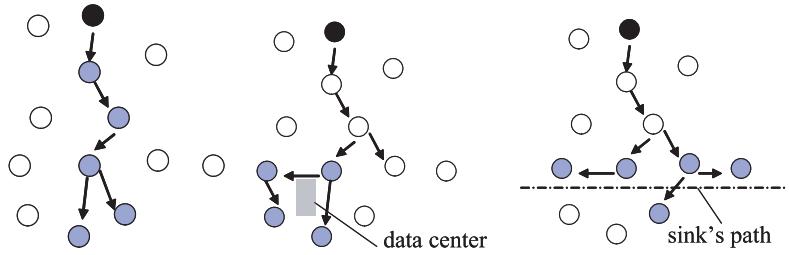
3.2.3 Regular Movement on Route

In this movement, a sink moves on a route in a regular manner. When the sink moves at a constant speed, the interval of the sink arrivals is a constant. Therefore, each node knows the timing of the data transmission to the sink. However, a node far from the route of the sink cannot transmit data to the sink directly as shown in **Fig. 5**. In this case, node n_5 has to transmit data via multi-hop communications, which incurs more power consumption.

3.2.4 On-demand Movement

In this movement, a sink moves according to a request from a node. A node broadcasts a data collection request with own position periodically or in a specified timing such as initiating a CB and completing all the data transmission by CB. Receiving the data collection request, a sink moves to the node according to the position information. The position information is acquired by GPS or set in system installation.

We consider more sophisticated movement in addition to the four movements such as the shortest path movement and the movement for the minimum delay for data collections.



(a) Line Type (b) Near-point Type (c) Along-route Type

Fig. 6 Extended CB for efficient data collection.

3.3 Extended CB for Efficient Data Collection

In the original CB (Concentric type, hereafter) as shown in Section 3.1, when a source node cooperates in a concentric pattern with neighbor nodes, the greater the quantity of generated data the more the cooperative nodes that are needed. Data collection is difficult when the number of cooperative nodes increases because it is difficult for the mobile sink to visit a large number of nodes.

In this paper, we propose the following three extensions of cooperative buffering for efficient mobile sink data collection. The concept of these schemes are shown in **Fig. 6**. These schemes are based on a localization method such as GPS and the methods [10),11)].

Line type In the line type, the nodes on a line buffer the data cooperatively. The nodes in both directions buffer data cooperatively. The nodes in one direction buffer data cooperatively in the half line type, which is a variation of the line type. When collecting the cooperatively-buffered data from nodes, a mobile sink such as car, moves in a straight line which is an easier way to collect data than with the concentric type.

Near-point type In some cases, mobile sinks such as city buses and garbage collection trucks move along a predetermined route. In this case, a data collection point such as bus stop may be used for convenient data collecting. Mobile sinks visit the data collection point such as data center periodically. In the near-point type, the nodes near the point buffer data cooperatively. A source node transmits data to the nodes near the point via multi-hop communications. For example, nodes buffer data cooperatively in a concentric

manner originated from that point.

Along-route type As well as the near-point type, the along-route type is used with mobile sinks moving through a determined route, such as a city bus. In this type, nodes along the route of mobile sinks buffer data cooperatively. A source node transmits data to the nodes along the route of mobile sinks via multi-hop communications. Mobile sinks collect the cooperatively-buffered data from the nodes without going off the route.

4. Performance Evaluation

We evaluate the power consumption of the proposed CB. We show the power consumption of each scheme for various sizes of data generation. **Table 2** shows the simulation parameters. We refer to the transmission rate of Zigbee, and the transmission current, the receive current. In addition, we select of node's maximal buffer size as 512 Kbytes, which is the same as MICA Mote. Based on this value and considering the reserved area for data relay, we select that the buffer size 128 or 256 Kbytes. In addition, we use CSMA/CA 2way as MAC protocol. The performance metric is the total power consumption which consists of power consumption of data transmission from a source node to cooperative nodes, that from the cooperative nodes to a sink node and that of receive data including overhearing. We assume that data transmission from the cooperating nodes to the sink is always single-hop communication. Moreover, the sink arrives at the nodes after buffering all the data cooperatively. A large amount of generated data is fragmented to small size of packets. The data fragment size indicates the fragmented packet size in application layer.

We formulate cooperative buffering in a simple analytical model in Section 4.1. In Section 4.2, some simulation results of the proposed CB for various node selections of buffering are provided.

Table 2 Simulation parameter.

Channel Rate	250 kbps
Transmission Power	2 dBm
Receive Threshold	−74 dBm
Transmission Current	17.4 mA
Receive Current	19.7 mA
Fragment Size	1,024 bit

4.1 Formulation

We formulate and compare the power consumption of the proposed CB and the conventional multi-hop scheme when a source node generates a large amount of data. We assume that the power consumption is proportional to the amount of data which the node sends to a sink or cooperative nodes.

B^h_{CB} $(h = 1, 2, 3, \cdots)$: Power consumption of CB when nodes within h-hops from the source node buffer data cooperatively

B_{MH} : Power consumption of the conventional multi-hop communications

We use the following several parameters:

P_t : power consumption for data transmission

P_r : power consumption for data receive

C : buffer size of a node

D : total amount of generated data $(D = kC|k = 1, 2, 3, \cdots)$

R : transmission range of a node

N : average number of neighbor nodes (average number of nodes within a circle whose radius is R)

K : hop count from a source node to a sink

4.1.1 Power Consumption of CB

In this evaluation, we do not consider the power consumption of the sink and that of control packet exchanges. Therefore, the power consumption of CB B^h_{CB} is composed of the following elements:

B^h_n : power consumption for data transmission to a neighbor when the node is h-hop away from the source node

B_s : power consumption for data transmission to a sink

B_r : receive power consumption (including overhearing)

Let us consider a case where a source node cooperates with one-hop neighbors. The number of one-hop neighbors is N including itself, so when the amount of the generated data is $1 \leq \frac{D}{C} \leq N$, the power consumption for data transmission of the neighbor node B^1_n is formulated as Eq. (1).

$$B^1_n = P_t \cdot C \cdot \left(\frac{D}{C} - 1\right), \qquad (1 \leq \tfrac{D}{C} \leq N). \tag{1}$$

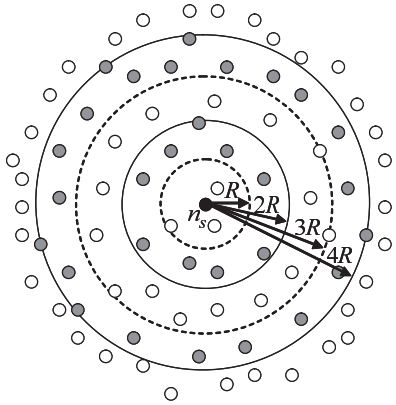


Fig. 7 Number of nodes within transmission range.

The source node retains a data size C and requests buffering the remaining data to its neighbor nodes.

Next, we consider the case where a source node cooperates within h-hop neighbors. As shown in **Fig. 7**, the h-hop neighbors from the source node are the nodes within a circle whose radius is hR. Therefore, the number of nodes within a circle with radius hR, is h^2N. The number of nodes within h-hop from the source node, N_h, is derived as follows:

$$N_h = h^2N - (h-1)^2N, \qquad (h > 1). \tag{2}$$

In this formulation, nodes are uniformly distributed. We assume that there are averagely N nodes within a circle whose radius is R. Therefore the number of nodes in a unit area is $N/\pi R^2$. On the other hand, D/C nodes are necessary for keeping data size D. The area of a circle whose radius is HR is $\pi(HR)^2$. Therefore, the maximum number of hops from the source node and cooperative nodes for given D, H is derived from the equation $N/\pi R^2 \times \pi(HR)^2 = D/C$. Therefore,

$$H = \left\lceil \sqrt{\frac{1}{N} \cdot \frac{D}{C}} \right\rceil. \tag{3}$$

In this case, B_n^H is obtained as:

$$B_n^H = \sum_{h=1}^{H-1} (h \cdot P_t \cdot C \cdot N_h) + H \cdot P_t \cdot C \cdot \left\{ \frac{D}{C} - \sum_{h=1}^{H-1} N_h \right\} \tag{4}$$

$$= P_t \left\{ (D-C) \cdot h - \frac{1}{6} \cdot h \cdot (h-1) \cdot (2h-1) \cdot C \cdot N \right\}. \tag{5}$$

After buffering all the generated data in the cooperative nodes, all nodes send the buffered data to the sink. In this formulation, we assume that all cooperative nodes including the source node send data to the sink directly. The total amount of data is D, so the power consumption of data transmission to the sink B_s becomes:

$$B_s = P_t \cdot D \tag{6}$$

In cases where a source node cooperates with nodes more than two-hops away, B_s is calculated by Eq. (6).

From Eqs. (5) and (6), the summation of B_n^h and B_s is formulated as:

$$B_n^h + B_s = P_t \left\{ (D-C) \cdot h - \frac{1}{6} \cdot h \cdot (h-1) \cdot (2h-1) \cdot C \cdot N + D \right\}. \tag{7}$$

Nodes consume energy, when they receive data. Here, we define α:

$$\alpha = (D-C) \cdot h - \frac{1}{6} \cdot h \cdot (h-1) \cdot (2h-1) \cdot C \cdot N + D \tag{8}$$

From Eqs. (7) and (8), the receive power consumption including overhearing B_r is derived as the following equation:

$$B_r = P_r \cdot \alpha \cdot N. \tag{9}$$

The total power consumption of CB B_{CB}^h is the summation of B_h, B_s and B_r. Thus, from Eqs. (7), (8), and (9), we can obtain the following equation:

$$B_{CB}^h = B_n^h + B_s + B_r = P_t \cdot \alpha + P_r \cdot \alpha \cdot N. \tag{10}$$

4.1.2 Power Consumption of Multi-hop Communications

We derive the power consumption of the conventional multi-hop communications. In this case, a source node sends data to a sink with K hops. So, B_{MH}^k becomes:

$$B_{MH}^K = P_t \cdot D \cdot K + P_r \cdot D \cdot K \cdot N. \tag{11}$$

4.1.3 Evaluation 1: Power Consumption of CB

We evaluate the total power consumption of CB. We calculate the power con-

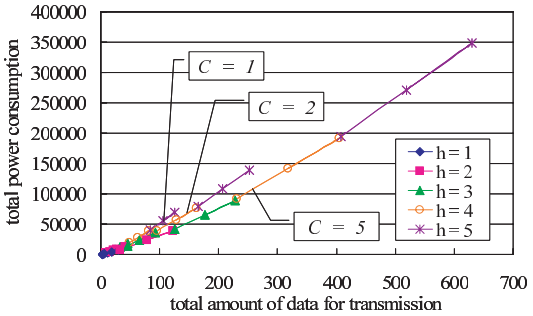


Fig. 8 Power consumption versus total generated data.

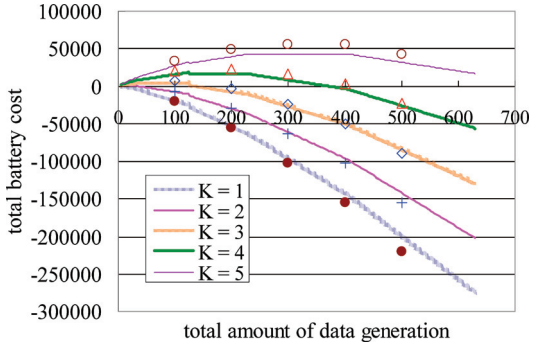


Fig. 9 Comparison of power consumption of CB and multi-hop scheme.

sumption of CB by means of Eq. (10). The ratio of the power consumption to the total amount of generated data when the number of neighbors is $N = 5$ is evaluated for various buffer size of nodes $C = 1, 2$ and 5. The analytical result of power consumption is shown in **Fig. 8**. The ratio of transmission and receive current is 17.4 : 19.7 and is based on MICA Mote modules. In Fig. 8, we find the following two characteristics. First, the larger the value of C is, the lower the increase of total power consumption. This is because a larger buffer C leads to a decrease the number of nodes which take part in CB. Second, if D becomes larger then more power is required. This is because a source node needs to buffer data with farther nodes by the increase of D. Therefore, cooperative nodes need power to relay data.

4.1.4 Evaluation 2: Comparison with Multi-hop Communications

We compare the power consumption of the proposed CB and the conventional multi-hop communications. The performance improvement of CB over the conventional multi-hop scheme is shown in **Fig. 9**. The power consumption is calculated by Eq. (11), which is shown in lines. In addition, the simulated results (plots) are also shown in Fig. 9. The performance improvement of CB is calculated by the following equation:

$$Improvement = Energy_{Multi} - Energy_{CB}, \quad (12)$$

where $Energy_{Multi}$ and $Energy_{CB}$ are the respective energy consumptions of multi-hop scheme and CB. The positive value means the performance improvement of CB. We set $N = 5$, $C = 5$ and $h = 1, 2, \cdots, 5$. From Eq. (3), the maximum value of D is 625 when $h = 5$. Therefore, the case where $D > 625$ is beyond the scope of available buffer because a buffer overflow occurs.

In Fig. 9, the greater the value of K, the more the advantage provided by CB. This is because in the multi-hop scheme, the total power consumption increases with an increase in the hop count from the source node to a sink. In contrast, the power consumption of CB is not affected by the sink's location. In addition, when $K = 1$, the advantage of CB is always negative value. A source node sends its data to a sink directly in multi-hop scheme when $K = 1$. In contrast, a source node sends its data to a sink via cooperative nodes in CB. Moreover, when $2 \leq K \leq 4$, the CB has both positive and negative values. This is because when D is small, a source node performs CB with a small number of cooperative nodes. When D is large, a source node needs a large number of cooperative nodes. In addition, when $K = 5$, the advantage of CB is always a positive value. In CB, the farther a sink is from a source node, the more CB can decrease the total power consumption than multi-hop scheme. From these results, when $K \geq 5$, CB can achieve better performance of power consumption than multi-hop communications.

4.2 Simulation Evaluation

4.2.1 Original CB (Concentric type)

First, we evaluate the case where there is one source node. The simulation topology and the result are shown in **Fig. 10** and **Fig. 11**, respectively. In Fig. 10,

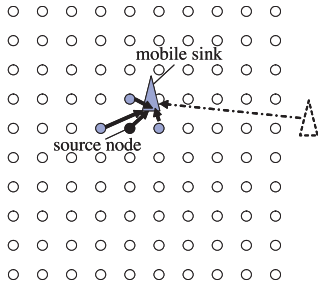


Fig. 10 Simulation topology of concentric type.

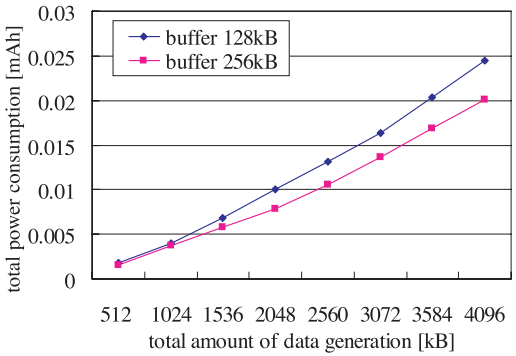


Fig. 11 Simulation result of concentric type.

there are 10×10 nodes in the network, and a source node is placed in a fixed point. The sink is outside the network. Data are generated at the source node. After cooperatively buffering the data in a concentric manner, the sink moves to the source node and its cooperative nodes to collect the data. The buffer capacity of a node is 128 and 256 Kbyte. As shown in Fig. 11, the model with 256 Kbyte achieves less power consumption than that with 128 Kbyte. This is because a node can buffer much data when it has a large buffer. In this case, all data can be buffered only with near nodes. Moreover, the greater the amount of generated data, the greater the difference between the models with 128 and 256 Kbyte because the source node must cooperate with nodes that are farther away in cases where the amount of generated data is large.

Next, we evaluate the case where multiple sources exist. The simulation topology is 10×10, as shown in **Fig. 12**. In Fig. 12, two source nodes, Source 1 and 2, are placed in the fixed points. The sink is initially outside the network. Data are generated at the source nodes. After cooperatively buffering the data in a concentric manner, the sink moves to the source node and its cooperative nodes to collect the data. We assume that data is generated at Source 2 after completing the cooperative buffering from Source 1. The buffer capacity of a node is 128 or 256 Kbyte. The result is shown in **Fig. 13**. The dashed lines in the figure show the result of buffer partitioning; the solid lines results without buffer partitioning. We find that the model with buffer partitioning increases the power

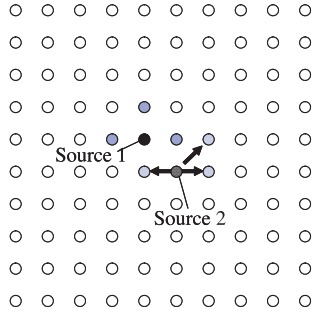


Fig. 12 Multiple source nodes.

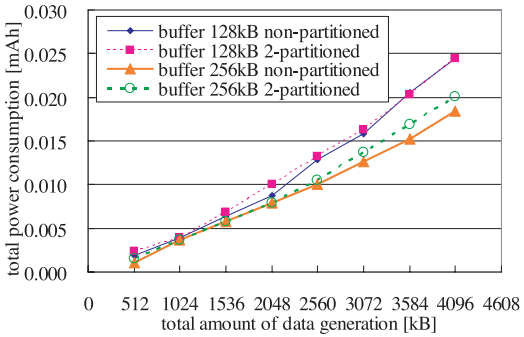


Fig. 13 Simulation result of multiple source nodes.

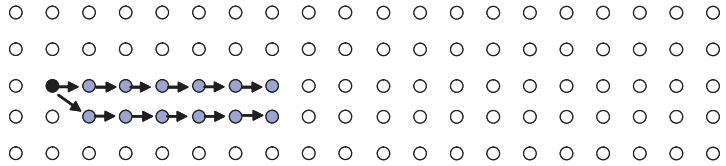


Fig. 14 Line type (Two half lines).

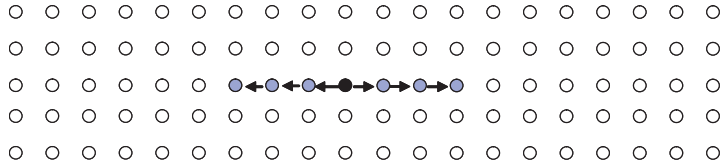


Fig. 15 Line type (One line).

consumption slightly. The buffer partitioning causes an increase in the number of cooperative nodes which leads to increased power consumption. Simple buffer partitioning might increase the power consumption.

4.2.2 Line Type

We evaluated the performance of the line type in the topology of 5×20. The buffer capacity of a node is 256 Kbyte. We consider four cases: 1-line, 2-line, 1-half-line and 2-half-line. We illustrate the Two Half Lines and One Line in **Fig. 14** and **Fig. 15**, respectively. In both Fig. 14 and Fig. 15, a source node is

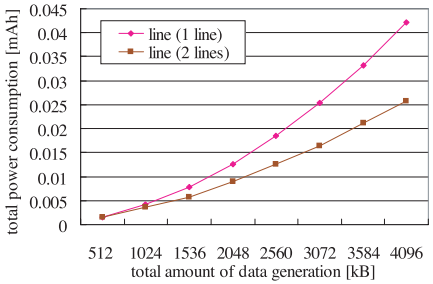


Fig. 16 Result of line type.

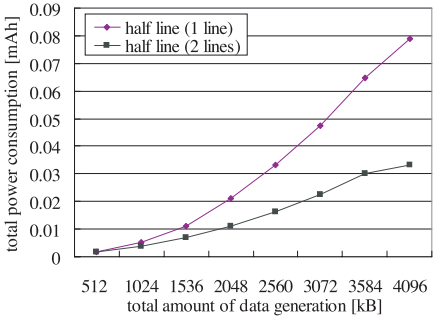


Fig. 17 Result of line type (Half).

placed in a fixed point. The sink is outside the network. Data are generated at the source node. In Fig. 14, after cooperatively buffering the data in a two-half-line manner, the sink moves to the source node and its cooperative nodes to collect the data. On the other hand, in Fig. 15, the generated data are buffered in a one-line manner. The results of the line and the half line types are shown in **Fig. 16** and **Fig. 17**, respectively. In both figures, the power consumption increases as the generated data size increases. This tendency is especially obvious in 1-line because the number of hops for data transmission to the cooperative nodes always increases in 1-line case. In addition, the power consumption of Fig. 16 is totally less than that of Fig. 17. This is because the line type easily cooperates with more nodes in both directions, however the half line type needs more hop counts to cooperate with the same number of nodes as the line type.

4.2.3 Near-point Type

We evaluate the near-point type in the following cases: simple and shortest hop cases. In the simple case, the source node cooperates with nodes in a concentric manner originated from a data center as shown in **Fig. 18**. In Fig. 18, there are 10×10 nodes in the network; a source node and a data center are placed in a fixed point. A sink is out of the network. Data are generated at the source node. After cooperatively buffering the data with the nodes around the data center in a simple-near-point manner, the sink moves to the data center and the cooperative nodes to collect the data. In the shortest hop case as shown in **Fig. 19**, the source node cooperates with nodes in the shortest hop manner, which is a more sophisticated case than the simple case. As shown in Fig. 19, after cooperatively

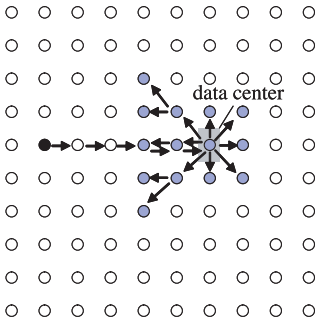


Fig. 18 Near-point type (Simple).

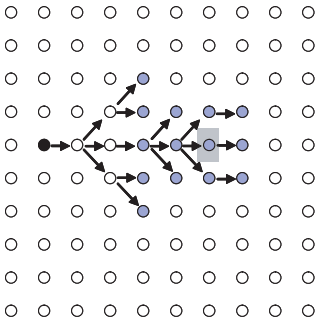


Fig. 19 Near-point type (Shortest hop).

buffering the data with the nodes around the data center in a shortest-hop-near-point manner, the sink moves to the data center and the cooperative nodes to collect the data.

This simulation result is shown in **Fig. 20**. The shortest hop case achieves slightly less power consumption than the simple case because the source node transmits data to the cooperative nodes in a shortest hop manner.

4.2.4 Along-route Type

Besides the near-point type, we evaluate the along-route type in the following cases: simple and shortest hop cases. In the simple case, the source node cooperates with nodes in a concentric manner originated from a node near the path. As

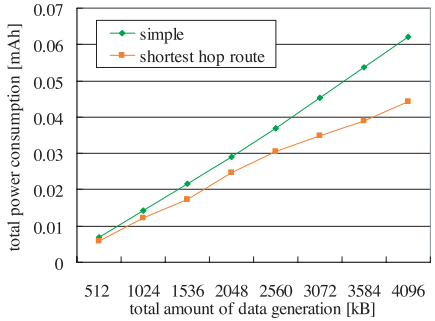


Fig. 20 Result of near-point types.

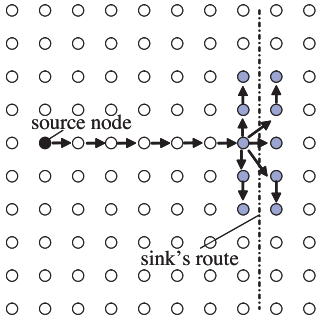


Fig. 21 Along-route type (Simple).

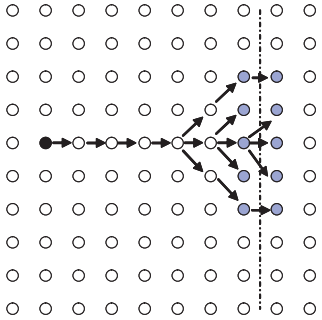


Fig. 22 Along-route type (Shortest hop).

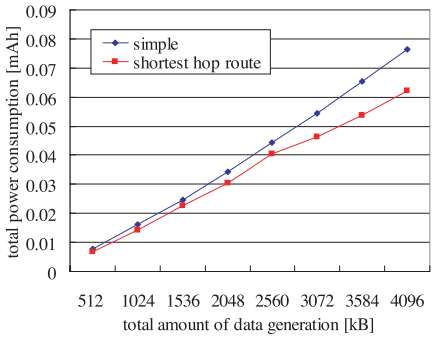


Fig. 23 Result of along-route types.

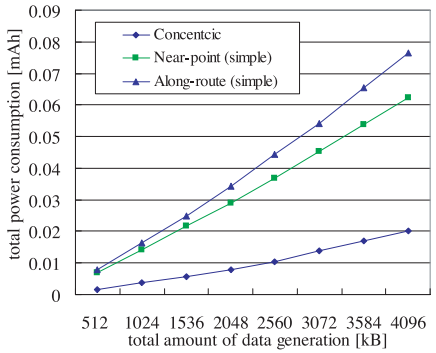


Fig. 24 Comparison of concentric, near-point and along-route types.

shown in **Fig. 21**, there are 10×10 nodes in the network; a source node is placed in a fixed point. The sink is outside the network. Data are generated at the source node. After cooperatively buffering the data along the route in a simple-along-route manner, the sink moves along the route to collect the data. As shown in **Fig. 22**, in the shortest hop case, the source node cooperates with nodes in the shortest hop manner, which is a more sophisticated case than the simple case. In Fig. 22, after cooperatively buffering the data along the route in a shortest-hop-along-route manner, the sink moves along the route to collect the data.

This simulation result is shown **Fig. 23**. The shortest hop case achieves slightly less power consumption than the simple case because the source node transmits data to the cooperative nodes in a shortest hop manner.

4.2.5 Comparison of Buffering Schemes

The near-point and along-route types are compared with the concentric type in **Fig. 24**. The concentric type achieves superior performance as can be seen in Fig. 24. The near point type is next best in terms of energy consumption. The near-point and along-route types need multi-hop communications so these types consume more power than the concentric type. The reason why the along-route type power consumption is worst is that the number of hops increases because of the node cooperation along a route.

These buffering schemes should be used in appropriate applications or situa-

tions. The concentric type should be used with mobile sinks moving randomly such as cars, taxis, and people and is suitable for the following two cases. The first one is the case where there are small number of nodes in a network and mobile sinks can visit all nodes. The other case is where the data generation frequency is quite low. In these cases, the concentric type is suitable because of its superior performance in energy consumption in sensor nodes. However, the concentric type needs position information of each node for data collection. The line type is an easier way of data collection than the concentric type because the cooperative nodes are arranged on a line. When mobile sinks move through a determined route such as bus, the near-point and along-route types are suitable. These types do not need any node position information. The mobile sinks collect the cooperatively-buffered data just by moving along the route. If some data collection points exist in a network such as bus stop, the near-point type is appropriate. If not, the along-route type should be useful.

4.2.6 Discussion on Energy Consumption Model

In this paper for the sake of simplicity, we focus on energy consumption during transmit and receive to show the fundamental performance of the proposed CB for simplicity. It is known that idle states may consume energy in some situations. There are many proposals to reduce the energy consumption of idle states such as reducing idle times of CSMA-type protocols and designing TDMA-type protocols. Actually, the paper 12) in the reference literature, which proposes Directed Diffusion, shows the energy consumption of Directed Diffusion with and without considering idle states. As the result, the authors therefore suggest that Directed Diffusion is effective in situations where adopting a MAC protocol with a small number of idle times such as TDMA-type MAC protocols. Besides Directed Diffusion our proposed CB is also preferable for use with MAC protocol that can reduce the effect of idle state. More detailed investigation of how MAC protocols affect energy consumption is an issue for future study.

5. Conclusion

In this paper, we have proposed a novel buffering scheme for DTMSN called CB or cooperative buffering. In the proposed CB, the sensor node which has a large amount of data cooperates with its neighbor nodes to buffer them in a distributed manner. The cooperatively buffered data are transmitted directly to the mobile sink when it arrives. After proposing CB, this paper discusses use of an extension for easy collection of the sink, an extension for multi source nodes, and some sink mobility strategies. As a result of the theoretical formulation and simulation, we show that the proposed CB can handle multimedia data and achieve low power operation. An appropriate CB version should be used according to the particular application or situation. In the future, we will evaluate CB performance for cases where there are three or more source nodes. In addition, we will also investigate CB performance including delays under various sink movement patterns.

Acknowledgments This work is supported by a Grant-in-Aid for Scientific Research (A)(No.17200003 and 20240005) and a Grant-in-Aid for Young Scientists (B) 20700063.

References

1) Heinzelman, W., Murphy, A., Carvalho, H. and Perillo, M.: Middleware to support sensor network applications, *IEEE Network Magazine*, Vol.18, No.1, pp.6–14 (2004).
2) Mainwaring, A., Polastre, J., Szewczyk, R. and Culler, D.: Wireless Sensor Networks for Habitat Monitoring, *First ACM International Workshop on Wireless Sensor Networks and Applications* (*WSNA*), pp.88–97 (2002).
3) Yan, T., He, T. and Stankovic, J.: Differentiated Surveillance for Sensor Networks, *First ACM Conference on Embedded Networked Sensor Systems* (*SenSys*), pp.51–62 (2003).
4) Rahimi, M., Baer, R., Iroezi, O., Garcia, J., Warrior, J., Estrin, D. and Srivastava, M.: Cyclops: In Situ Image Sensing and Interpretation in Wireless Sensor Networks, *Third ACM Conference on Embedded Networked Sensor Systems* (*SenSys*), pp.192–204 (2005).
5) Ma, H. and Liu, Y.: Correlation Based Video Processing in Video Sensor Networks, *2005 International Conference on Wireless Networks, Communications and Mobile Computing* (*ICWIN*), pp.987–992 (2005).
6) Akyildiz, I., Melodia, T. and Chowdhury, K.: Wireless Multimedia Sensor Networks: A Survey, *IEEE Wireless Communications Magazine*, Vol.14, No.10, pp.32–39 (2007).
7) Jain, S., Fall, K. and Patra, R.: Routing in a delay tolerant network, *ACM SIGCOMM 2004*, pp.145–158 (2004).
8) Shah, R., Roy, S., Jain, S. and Brunette, W.: Data MULEs: Modeling and Analysis of a Three-tier Architecture for Sparse Sensor Networks, *Elsevier Ad Hoc Networks Journal*, Vol.1, pp.215–233 (2003).
9) Wang, G., Cao, G., Porta, T. and Zhang, W.: Sensor Relocation in Mobile Sensor

Networks, *IEEE INFOCOM 2005*, pp.2302–2312 (2005).
10) Niculescu, D. and Nath, B.: DV Based Positioning in Ad Hoc Networks, *Telecommunication Systems*, Vol.22, No.1-4, pp.267–280 (2003).
11) He, T., Huang, C., Blum, B., Stankovic, J. and Abdelzaher, T.: Range-Free Localization Schemes for Large Scale Sensor Networks, *ACM MOBICOMM 2003*, pp.81–95 (2003).
12) Intanagonwiwat, C., Govindan, R., Estrin, D., Heidemann, J. and Silva, F.: Directed Diffusion for Wireless Sensor Networking, *IEEE/ACM Transactions on Networking*, Vol.11, No.1, pp.2–16 (2003).

(Received May 25, 2009)
(Accepted December 17, 2009)
(Original version of this article can be found in the Journal of Information Processing Vol.18, pp.96–109.)

Yu Takada was born in 1983. He received a B.E. degree from Shizuoka University in 2008. He later joined the Graduate School of Informatics in Shizuoka University Japan. His research interests include sensor networks.

Masaki Bandai was born in 1973. He a received his B.E. and M.E. degrees in electrical engineering and Ph.D. degree in information and computer science from Keio University, in 1996, 1998 and 2004, respectively. He joined the Sony Corporation in 1998 serving there through 2000. Since 2004, he has been an assistant professor at Department of Computer Science, Shizuoka University, Japan. From 2006 to 2007, he was a visiting scholar at Department of Electrical and Computer Engineering, the University of British Columbia, Canada. He is engaged in research on computer networks. He is a recipient of the IPSJ Yamashita SIG Research Award in 2007.

Tomoya Kitani was born in 1979. He received his B.E., M.E., and Ph.D. degrees in information science and technology from Osaka University in 2002, 2004 and 2006, respectively. From 2005 to 2008, he was an assistant professor at Graduate School of Information Science, Nara Institute of Science and Technology. Since 2008 he has been serving as an assistant professor of at the Division of Global Research Leaders, Shizuoka University. His research interests include combinatorial problems, embedded systems and vehicular ad hoc networks. He is a member of IEEE, IEICE, and IPSJ.

Takashi Watanabe was born in 1959. He received his B.E., M.E., and Ph.D. degrees in communications engineering from Osaka University in 1982, 1984 and 1987, respectively. In 1987, he joined the Engineering Faculty at Tokushima University as an assistant professor. In 1990, he moved to Faculty of Engineering, Shizuoka University. He was a visiting researcher at University of California, Irvine from 1995 through 1996. He is currently a professor of Graduate School of Science and Technology, Shizuoka University. He is a translator of the Japanese version of 802.11 Wireless Networks: The Definitive Guide (published by O'RELLY, 2003). His current research interests include computer networks, mobile networking, ubiquitous networks, ad hoc networks and sensor networks. He is a member of IEEE, IEEE Communications Society, IEEE Computer Society, ACM SIGMOBILE, and IPSJ.