Data Aggregation and Forwarding Route Control for Efficient Data Gathering in Dense Mobile Wireless Sensor Networks

Kazuya Matsuo, Keisuke Goto, Akimitsu Kanzaki, Takahiro Hara, and Shojiro Nishio

Osaka University, 1-5 Yamadaoka, Suita, Osaka, Japan
{matsuo.kazuya,goto.keisuke,kanzaki,hara,nishio}@ist.osaka-u.ac.jp

Abstract. This chapter presents a data gathering method considering geographical distribution of data values for reducing traffic in dense mobile wireless sensor networks. First, we present our previous method (DGUMA) which is a data gathering method that efficiently gathers sensor data using mobile agents in dense mobile wireless sensor networks. Second, we introduce an extended method of DGUMA, named DGUMA/DA (DGUMA with Data Aggregation), that exploits geographical distribution of data values in order to further reduce traffic. Finally, we analyze DGUMA/DA and confirm the effectiveness of the method through some simulation experiments.

Keywords: mobile agent, data gathering, data aggregation, forwarding route control.

1 Introduction

Recently, *participatory sensing*, where sensor data are gathered from portable sensor devices such as smart phones, has attracted much attention [2, 9, 14, 15]. In participatory sensing, it is general that sensor data are uploaded to the Internet through some infrastructures such as 3G and LTE networks. It is undesirable for participatory sensing to generate a large amount of traffic that may exhaust the limited channel bandwidth shared by a wide variety of applications. For this reason, *MWSNs (Mobile Wireless Sensor Networks)*, which are constructed by mobile sensor nodes held by ordinary people without any infrastructure [17], have recently attracted much attention as a way of realizing participatory sensing. In a MWSN, sensor readings are gathered to a sink through multi-hop wireless communication [5, 20].

In MWSNs constructed by mobile sensor nodes held by ordinary people, the number of sensor nodes is generally very large. For example, there are generally more than 10,000 people in the daytime in some stations in major cities such as Tokyo. When assuming that all of them hold sensor devices with Wi-Fi interface whose communication range is about 100[m], a sensor node can directly communicate with about 100 sensor nodes. In such an environment, an arbitrary geographical point in the sensing area can be sensed by many sensor nodes when an application monitors the geographical distribution of temperature in the station. We call this networks *dense MWSNs*.

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_5

On the other hand, most of MWSN applications require a certain geographical granularity of sensing in a specific area (e.g., sensor data of every 100[m]×100[m] square in a 1,000[m]×1,000[m] flatland) at every sensing time. In such a situation, if a sink gathers sensor data from all sensor nodes in the entire area, the network bandwidth and the battery of sensor nodes are unnecessarily wasted. Thus, it is desirable to efficiently gather sensor data from the minimum number of sensor nodes which are necessary to guarantee the geographical granularity required by the application. For this aim, as a data gathering method which efficiently gathers sensor data in dense MWSNs, *DGUMA (Data Gathering method Using Mobile Agents)* has been proposed [6]. DGUMA uses mobile agent, which is an application software that autonomously operates on a sensor node and moves among sensor nodes. Mobile agents are generated by the sink and allocated on sensor nodes located near the sensing points, which are determined from the requirement on geographical granularity. For gathering sensor data, DGUMA constructs a static tree-shaped logical network whose nodes are mobile agents (sensing points). Every time when the sensing time comes, sensor nodes where agents locate perform sensing and send the sensor data to the sink according to the tree-shaped network. By doing so, DGUMA can reduce the traffic for gathering sensor data since mobile agents control transmissions of sensor data.

Here, sensor readings on environmental information such as sound and temperature tend to have a same or similar value at adjacent sensing points. However, DGUMA does not consider such a characteristic of sensor readings, and gathers sensor readings acquired by all agents even when there are ones with the same value. If we can aggregate such sensor readings with the same value, further reduction of traffic for gathering sensor data can be expected [1, 10, 11, 13]. In addition, it is general that the geographical distribution of data values changes over time. In such a case, it is effective to dynamically construct communication routes (tree-shaped network in DGUMA) so that many sensor readings with the same value are aggregated. However, since DGUMA constructs a static tree-shaped network for gathering sensor data, effective data aggregation cannot be achieved in some geographical distribution of data values.

In this chapter, we present an extended method of DGUMA, named *DGUMA/DA (DGUMA with Data Aggregation)*, that considers the geographical distribution of data values in dense MWSNs. In DGUMA/DA, each mobile agent aggregates multiple readings with the same value in order to reduce the traffic for gathering sensor data. Moreover, at every sensing time, each mobile agent searches its adjacent agents which have the same reading during gathering sensor data by changing their direction of forwarding sensor data from lengthwise (up or down) to crosswise (right or left) or vice versa. When an adjacent agent which has the same reading is found, the mobile agent fixes its direction of forwarding sensor data so that the adjacent agent can continuously aggregate the readings. In addition, when the reading of the adjacent agent on the fixed direction becomes different, the mobile agent releases its direction of forwarding sensor data. This mechanism increases the chances of aggregating multiple readings with the same value. Using these two mechanisms (i.e., data aggregation and forwarding route control), DGUMA/DA further reduces the traffic for gathering sensor data to the sink.

Furthermore, we confirm the effectiveness of DGUMA/DA by comparing with DGUMA through a theoretical analysis and some simulation experiments.

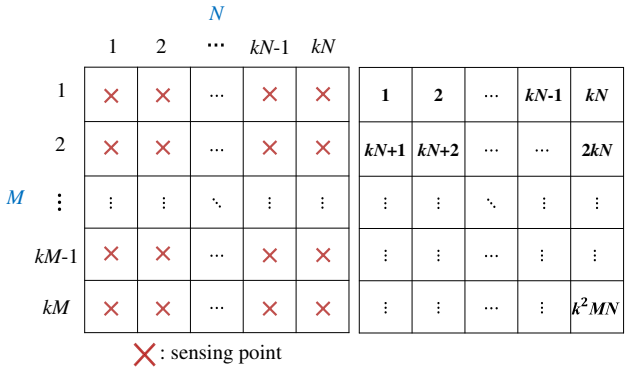


Fig. 1. A sensing area, sensing points and gridIDs

The remainder of this chapter is organized as follows. In Section 2, we describe assumptions in this chapter. In Section 3, we introduce related work. In Section 4, we describe our previous method (DGUMA). In Section 5, we explain the details of DGUMA/DA, which is an extended method of DGUMA. In Section 6, we discuss the performance gain of DGUMA/DA. In Section 7, we show the results of the simulation experiments. Finally, in Section 8, we summarize this chapter.

2 Assumptions

We assume dense MWSNs constructed by mobile sensor nodes which are held by ordinary people and equipped with a radio communication facility. These sensor nodes periodically observe (sense) a physical phenomenon (e.g., sound, temperature, and light), and communicate with each other using multi-hop radio communication. According to the requirement from an application, the sink periodically monitors the sensing area while guaranteeing the geographical granularity of sensing. More specifically, the sink gathers sensor readings from sensor nodes located near the sensing points which are determined from the requirement of the geographical granularity at the timing of sensing. We call the interval of data gathering the *sensing cycle*.

2.1 System Environment

The sensing area is assumed to be a two-dimensional plane whose horizontal to vertical ratio is $M : N$ (M and N are positive integers). The application specifies its requirement of the geographical granularity of sensing as an integer of $k^2 \cdot M \cdot N$ ($k = 1, 2, \cdots$). Then, the sink divides the sensing area into $k \cdot M \times k \cdot N$ lattice-shaped subareas (*grid*) and determines the center point of each grid as a sensing point, which is the target of data gathering. The sink assigns the *gridID* $\{1, 2, \cdots, k^2MN\}$ to each grid from left-upper grid to right-lower grid in order (see Fig. 1). Since no infrastructure for communication is available in the sensing area, the sink gathers sensor data by using a *MANET*

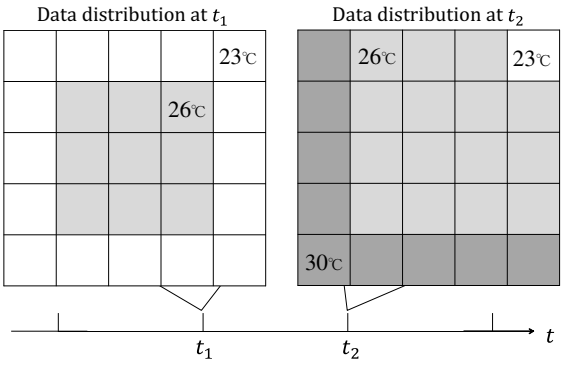


Fig. 2. An example of distribution of data values

(Mobile Ad Hoc Network) constructed by sensor nodes. The communication range of each sensor node is a circle with a radius of r. Each sensor node is equipped with a positioning device such as GPS, and communicates with other sensor nodes using *geo-routing*, which is a multi-hop radio communication based on their positions (the details are described in the next section). Each sensor node freely moves in the sensing area, while the sink is stationary. Since we assume that an enormous number of mobile nodes densely exist in the sensing area, there are many sensor nodes for each geographical point that can sense (cover) the point in the entire sensing area. Here, we assume that a sensing point is covered when at least one sensor node exists inside of a circle inscribed in the grid. We define this area the *valid area*, and the sensor reading sensed by the sensor nodes located in the valid area is defined as the *valid data*.

When assuming a physical phenomenon such as temperature in an urban area, the values of sensor readings observed at geographically close points tend to become the same with a high probability. In addition, the geographical distribution of data values changes as time passes. Fig. 2 shows an example of dynamic change in the geographical distribution of data values assuming temperature. In this figure, a number in a grid denotes the temperature observed in the corresponding grid. The same colored grids show that the same data values are observed. As shown in this example, the same data value tends to be observed at adjacent grids.

2.2 Geo-Routing

Sensor nodes adopt a geo-routing protocol based on that proposed in [7] to transport a message to the destination specified as a position (not a node). In this protocol, nodes perform a transmission process using the information on positions of the transmitter and the destination specified in the packet header. Specifically, the transmitter records the coordinates of the destination and itself into the packet header of the message, and broadcasts the message to its neighboring nodes. Each node which received this message judges whether it locates within the *forwarding area*. The forwarding area is determined based on the positions of the transmitter, the destination and the communication

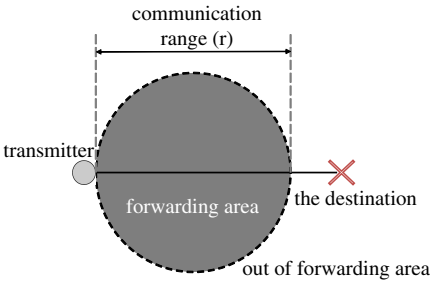


Fig. 3. Forwarding area

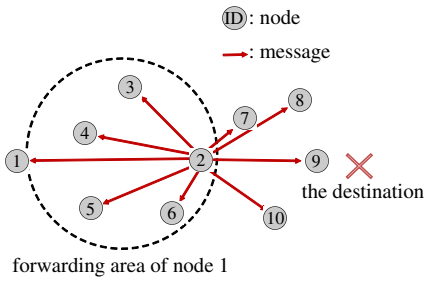


Fig. 4. An example of message forwarding in geo-routing

range, so that any node in the forwarding area is closer to the destination than the transmitter and can communicate directly to all nodes in the area (see Fig. 3). A node within the forwarding area sets the waiting time, and then it forwards the message after the waiting time elapses. The waiting time is set shorter as the distance between the node and the destination gets shorter. Each node within the forwarding area cancels its transmission process when it detects the message forwarded by another node. For example, in Fig. 4, nodes {2, . . ., 6} receive a message from node 1, and set the waiting time. Since node 2 is the closest node to the destination in the forwarding area of node 1, it first sends the message. On receiving this message, all nodes in the forwarding area of node 1 (i.e., nodes {3, . . . , 6}) cancel their transmission process. By repeating this procedure, the message is forwarded to nodes which are closer to the destination. If the transmitter node exists within half of the communication range ($r/2$) from the destination, each node which received the message sends an ACK to the transmitter node after the waiting time elapses instead of forwarding the message. As a result, the nearest node from the destination (which has first sent the ACK) can find that it is the nearest one because all nodes within $r/2$ from the destination can detect the ACK sent by the node and cancel sending it. If the transmitter node did not receive an ACK from any node, it also can find that the node itself is the nearest node.

3 Related Work

3.1 Location-Based Data Management in Dense MANETs

In [8], the authors proposed a data management method which realizes efficient access to location-based data in dense MANETs. In this method, nodes exchange data so that the data is always held by a node within the range of $r/2$ from the position corresponding to the data. By doing so, data access can be realized by using geo-routing. This method is similar to our method in the way that data (or agent) is held near a certain position. However, it is different from ours which aims data gathering to guarantee the geographical granularity of sensing specified from an application.

3.2 Data Gathering Utilizing Correlation of Data in Wireless Sensor Networks

In [16], the authors proposed a traffic reduction method utilizing temporal correlation of data in wireless sensor networks. In this method, each sensor node stores a sensor reading at last sensing time and compares a new sensor reading and the previous one at every sensing time. If the difference between the sensor readings is smaller than the threshold predetermined by the sink, the node does not send its sensor reading to the sink at the sensing time. Thus, the traffic for data gathering is reduced by utilizing temporal correlation of data. It is different from ours which reduces traffic by using the geographical distribution of data values. However, it is similar to ours in the way that the node does not send its sensor reading if the sensor reading corresponds to another one.

In [19, 21–24], the authors proposed data gathering methods which construct an overlay network to effectively aggregate sensor data using the spatial correlation of data in wireless sensor networks. In [19], sensor data are gathered using a routing tree, which is a combination of the minimum spanning tree and the shortest path tree. Raw data is sent according to the former tree and aggregated data is gathered according to the latter tree. In [21], an energy efficient routing tree is constructed using game theory that considers transmission energy, the effect of wireless interference, and the opportunity for aggregating correlated sensor data. In [22], clusters are constructed based on the compression ratio, which indicates the ratio that a node compresses the data received from its neighbors using spatial correlation. In [23], a routing tree is dynamically constructed. In this routing tree, each node sends its holding packet to its neighbors which hold more packets and exist closer to the sink. This is because a sensor node can have more opportunities to aggregate sensor data when holding more packets. In [24], assuming a circular sensing area, a data gathering method using mobile sensor nodes is proposed. This method divides the area into polar grids. In each grid, a node is chosen in each grid to aggregate data observed in the grid and to join a routing tree. These methods are similar to ours in the way that sensor data are gathered using spatial correlation. However, these methods do not consider the change in the geographical distribution of data values.

In [18], assuming that sensor nodes which have spatial correlated data tend to exist close to each other, and continuous queries which specify a condition on a sensor reading to be gathered, a dynamic route construction method for the queries is proposed. This method detects sensor nodes which satisfy the condition, and constructs some clusters consisting of sensor nodes which exist close to each other. For gathering sensor data, a minimum spanning tree is constructed by those clusters. When sensor nodes which satisfy the condition are changed by the change in the distribution of data value, the sink computes a new minimum spanning tree and informs it to all sensor nodes. This method is similar to ours in the way that routing tree is constructed dynamically according to the distribution of data value. However, this method is conducted in a centralized manner.

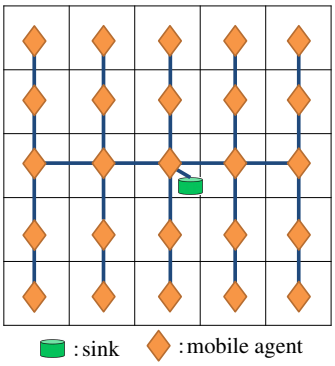


Table 1. Forwarding directions of agent data

Location of the sensor node that previously forwarded the agent data	Forwarding directions of the agent data
-(sink)	up, down, right, left
grid next to right	up, down, left
grid next to left	up, down, right
grid next to down	up
grid next to up	down

Fig. 5. A forwarding tree in DGUMA

4 DGUMA: Our Previous Method

In this section, we briefly describe our previous method, DGUMA [6], which is a sensor data gathering method using mobile agents.

4.1 Mobile Agent

A mobile agent is an application software which autonomously operates on a sensor node and moves between sensor nodes. A sensor node boots a mobile agent by referring to the agent data, which consists of the information on geographical granularity of sensing, the sensing cycle, and the position of the sink. The role of mobile agents is to transmit sensor data to the sink at every sensing time. Mobile agents are deployed on sensor nodes so that they can guarantee the requirement from the application regarding the geographical granularity of sensing according to the procedure described in Section 4.2.

4.2 Deployment of Mobile Agents

In DGUMA, the sink sends the agent data, which consists of information on the geographical granularity of sensing, the sensing cycle, and the position of the sink, along the static tree-shaped network created based on the geographical relationships among sensing points (see Fig. 5) according to the following procedure.

First, the sink generates the agent data and sends it to the sensing point in the grid where the sink itself locates using the geo-routing protocol described in Section 2.2. When the sensor node located at the closest position of the sensing point receives the agent data, it boots a mobile agent. As the initial operation, the mobile agent retransmits the agent data to the sensing points in the adjacent grids existing in the directions shown in Table 1. This retransmission of the agent data is repeated until mobile agents in grids on top and bottom edges of the sensing area are booted.

By doing so, a tree-shaped network whose root is the sink and nodes are mobile agents is constructed. We call this network the *forwarding tree*.

4.3 Movement of Mobile Agent

In DGUMA, if a sensor node on which a mobile agent operates moves away from the sensing point, the mobile agent moves from the current sensor node to another sensor node which locates closest to the corresponding sensing point. Specifically, a mobile agent starts moving when the distance between the sensing point and itself becomes longer than the threshold. This threshold is a system parameter which is set as a constant smaller than $r/2$ and the radius of the valid area for sensing, which can guarantee that a sensor node on which a mobile agent operates can communicate with all sensor nodes located near (within $r/2$) from the sensing point and can sense the data at the sensing point. In order to move to the sensor node located closest to the sensing point, the mobile agent issues a message containing the agent data, and broadcasts it to neighboring nodes within $r/2$ from the sensing point. As in Section 2.2, the sensor node located closest to the sensing point first sends an ACK and boots a mobile agent. Other sensor nodes cancel to send the same ACK because they can detect this ACK. Also, the original mobile agent stops its operation when detecting the ACK.

4.4 Transmission of Sensor Data

Mobile agents deployed at (near) sensing points send the sensor data held by the sensor nodes on which these agents operate to the sink at every sensing time. Here, a sensor data consists of a sensor reading and a gridID.

First, mobile agents located in grids of the top and bottom edges in the sensing area start to send their sensor data to their parents in the forwarding tree. In doing so, the geo-routing protocol is used. Each mobile agent can receive the sensor data from its children in the forwarding tree, because it keeps its own position within $r/2$ from the sensing point according to the procedure described before. When mobile agents except for that located in the grid where the sink exists receive the sensor data from all their children, they pack the received sensor data and their own sensor readings in a packet as their own sensor data, and forward the sensor data to their parents. This procedure is repeated until the mobile agent located in the grid where the sink exists receives sensor data from all its children. Finally, the mobile agent located in the grid where the sink exists packs all the received sensor data and its own sensor reading in a packet, and forwards it to the sink.

Fig. 6 shows an example of the above procedure. First, mobile agents at grids $\{1, \ldots, 6\}$ send a packet with their own sensor data (i.e., their own sensor readings and gridIDs) to their parents. On receiving the packet from the mobile agent at grid 1, the mobile agent at grid 7 sends the packet after adding its sensor data. Mobile agent at grid 12 performs in the same way as that at grid 7. The mobile agent at grid 8 receives multiple packets from grids 2 and 7, and packs sensor data included in these packets and its sensor data in a packet. Mobile agents at grids 9 and 11 perform in the same way as that at grid 8. Also, the agent at grid 10 receives multiple packets from grids $\{4, 9, 11\}$, packs them, and sends the packet with the aggregated sensor data to the sink.

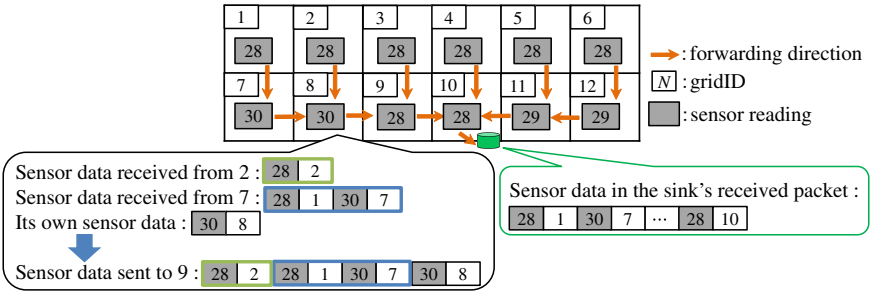


Fig. 6. An example of transmission of sensor data in DGUMA

5 DGUMA/DA: The Extended Method

As discussed in Section 1, DGUMA does not consider geographical distribution of data values, and gathers sensor readings acquired by all agents even when there are ones with the same value. In addition, DGUMA constructs a static forwarding tree for gathering sensor data. Considering dynamic change in the geographical distribution of data values, it is expected that traffic can be further reduced if we control the topology of the forwarding tree according to the data distribution.

In DGUMA/DA, each mobile agent aggregates multiple readings with the same value in order to reduce the traffic for gathering sensor data. Moreover, it dynamically constructs the forwarding tree so that more sensor readings can be aggregated when gathering sensor data.

5.1 Outline

Similar to DGUMA, DGUMA/DA deploys the agent data to $k^2 \cdot M \cdot N$ sensing points. Each mobile agent sets up *timer*, which will be described in Section 5.2. When the timer expires, the mobile agent sends sensor data to the sink. In this process, each mobile agent aggregates multiple readings with the same value as described in Section 5.3. In addition, DGUMA/DA dynamically constructs the forwarding tree according to the procedure described in Section 5.4.

When the sink receives the sensor data, it restores aggregated sensor readings, according to the procedure described in Section 5.5.

5.2 Timer Setting

In DGUMA, each mobile agent sends sensor data after receiving those from all its children. On the other hand, DGUMA/DA dynamically changes the forwarding tree. This means that the set of children for each mobile agent dynamically changes. Thus, each mobile agent cannot recognize whether its child(ren) exist(s) or not. In order to gather sensor data to the sink even in that situation, DGUMA/DA sets the timer for each mobile agent. Each mobile agent starts to send its sensor data to its parent when its timer has expired at each sensing time.

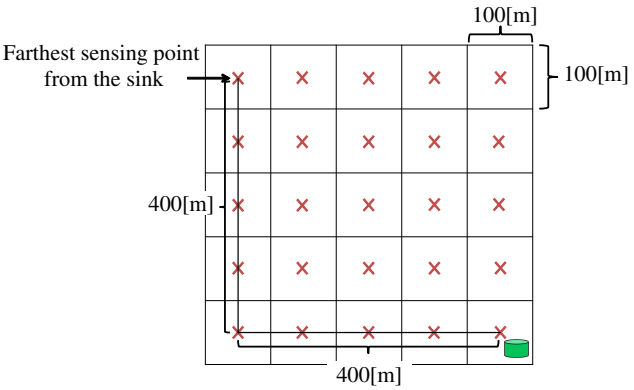


Fig. 7. An example of relationship of timer and the location of the sink

The timer, T, is determined by the following equation:

$$T = \frac{Dist_{max} - (|x_{sink} - x| + |y_{sink} - y|)}{Grain} \cdot (t_a + rand_{max}) + rand. \quad (1)$$

Here, t_a is the time required to send data to the adjacent agent, (x_{sink}, y_{sink}) is the coordinate of the sink, (x, y) is the coordinate of the sensing point, *Grain* is the distance between adjacent sensing points, and $Dist_{max}$ is the sum of the distances in x- and y-direction between the sensing point in the grid where the sink exists and that of the farthest sensing point from the sink. For example in Fig. 7, $Dist_{max}$ is 800[m] because the farthest sensing point from the sink is the top left one. *rand* is a random number within the range [0, $rand_{max}$] to avoid packet collision. By setting the timer for each mobile agent according to Eq.(1), mobile agents send sensor data in descending order of distance from the sink on the forwarding tree without any knowledge about their children.

5.3 Transmission of Sensor Data

Similar to DGUMA, each mobile agent sends a packet which contains multiple sensor data. However, unlike DGUMA, multiple gridIDs can be attached to a sensor reading. Mobile agents aggregate sensor readings using this packet structure.

At each sensing time, each mobile agent whose timer has expired puts its own sensor reading and gridID into a packet, and sends the packet to its parent. Here, the parent of the mobile agent is determined according to the forwarding route control described in Section 5.4. When a mobile agent received packets from other mobile agents before its timer expires, it aggregates its own sensor data and those in the received packets (details are described below), puts the aggregated sensor data into a packet, and sends the packet to its parent (or the sink). Here, since the timer of a mobile agent is set so as to expire after those of its descendants in the forwarding tree, every mobile agent can receive sensor data from all its children before its timer expires.

Each mobile agent aggregates sensor data according to the following procedure:

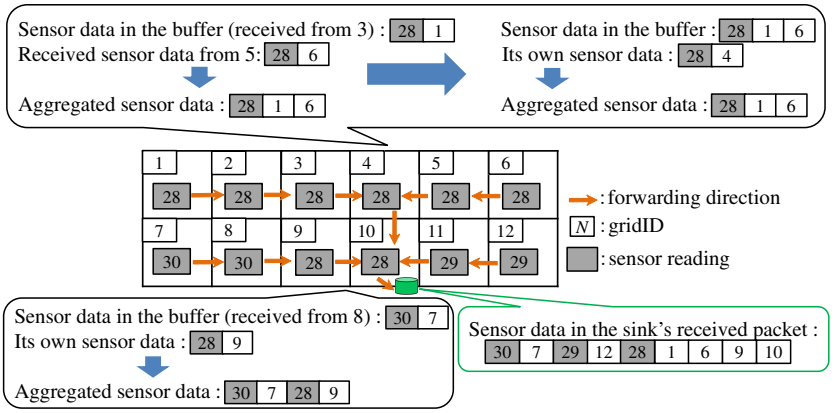


Fig. 8. An example of aggregating data values

(1) When a mobile agent has received multiple packets, it first copies sensor data included in the packet first received to its buffer.

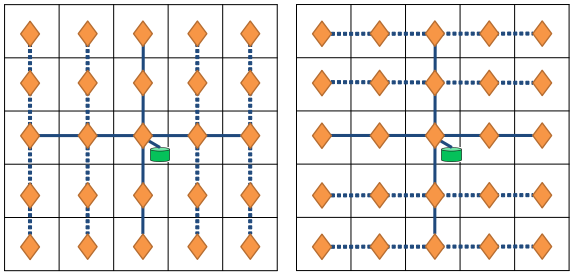
(2) It checks whether there is a sensor reading in another received packet, which is identical with that in the buffer. If so, the mobile agent adds only the gridIDs of the sensor reading after the gridIDs of the corresponding reading in the buffer. After that, the mobile agent adds other sensor data, whose sensor reading is not identical with any of those in the buffer, to the end of buffer.

(3) When its timer expires, the mobile agent adds its own sensor data to the buffer in the same way. If its sensor reading is identical with one of them in the buffer, it adds only its gridID after the gridIDs of the corresponding reading in the buffer and moves the corresponding sensor data to the end of the buffer. After that, the agent creates a new packet that contains all sensor data in the buffer, and sends the packet to its parent.

Here, when the mobile agent has received only one packet, or when there is only one sensor data in the buffer, it checks whether its own sensor reading and that at the end of the buffer is identical. If so, the mobile agent adds neither its own sensor reading nor its gridID to the buffer.

According to the above procedure, each mobile agent can aggregate sensor readings by adding only its gridID when there is a sensor reading identical with its own reading in the received packet. In addition, when the sensor readings become identical between adjacent sensing points on the forwarding tree, more traffic can be reduced since nothing is added to the packet. Note that all sensor readings can be restored at the sink from the aggregated sensor data. The details are presented in Section 5.5.

Fig. 8 shows an example of the above procedure. First, mobile agents at grids {1, 6, 7, 12} send a packet with its own sensor data (i.e., their own sensor readings and gridIDs) to their parents. On receiving the packet from the mobile agent at grid 1, the mobile agent at grid 2 copies the received sensor data to its buffer. After the expiration of its timer, the agent at grid 2 sends the packet without adding its sensor data (its own



(a) Lengthwise tree (b) Crosswise tree

Fig. 9. Two fundamental trees in DGUMA/DA

sensor reading and gridID) because its own sensor reading is identical with that at the end of the buffer. Mobile agents at grids {3, 5, 8, 11} perform in the same way as that at grid 2. On the other hand, after copying sensor data received from the agent at grid 8 to its buffer, the mobile agent at grid 9 adds its sensor data to the end of the buffer because there is no sensor data with the same sensor reading as its own reading (i.e., 28). Then, the mobile agent at grid 4 receives multiple packets from grids 3 and 5, and aggregates sensor data included in these packets according to the procedure described in step (2). After the aggregation, the buffer contains only one sensor data whose sensor reading is identical with its own reading (i.e., 28). Thus, the mobile agent sends the packet without adding its sensor data. Also, the agent at grid 10 receives multiple packets from grids {4, 9, 11}, aggregates them, and sends the packet with the aggregated sensor data to the sink.

5.4 Forwarding Route Control

DGUMA/DA dynamically constructs the forwarding tree based on two fundamental trees, the *lengthwise tree* and the *crosswise tree* (see Fig. 9) in order to aggregate more sensor readings. Here, the lengthwise tree has the same topology as the forwarding tree for deploying mobile agents. DGUMA/DA switches between the two fundamental trees at every sensing time, and searches routes on which sensor readings have the same value. By doing so, DGUMA/DA can construct the forwarding tree by which more sensor readings are aggregated. Here, mobile agents at the grids which are on the same column or row of the grid where the sink locates (solid lines in Fig. 9) do not change their forwarding directions.

The detailed procedure is as follows:

(1) At every sensing time, each mobile agent changes its forwarding direction, and sends a packet following the procedure in Section 5.3. Note that the lengthwise tree is used at the first sensing time.

(2) When each mobile agent receives a packet, the mobile agent checks whether the sensor reading at the end of the received packet is identical with its own reading. If

so, and the path from the child to itself is not fixed, it sends a *route fix message* to the child. On the other hand, if its own sensor reading is not identical with that of the end of the received packet, and when the route from the child to itself is fixed, the mobile agent sends a *fixed-route release message* to the child.

(3) When a mobile agent receives a route fix message from its current parent, it fixes the route from itself to the parent. After that, it does not change its forwarding direction at the subsequent sensing times.

(4) When a mobile agent receives a fixed-route release message, it stops fixing the route. After that, it restarts to change its forwarding direction at the subsequent sensing times.

By doing so, DGUMA/DA can construct the forwarding tree which aggregates more sensor readings even when the geographical distribution of data values dynamically changes.

Fig. 10 shows the detailed procedure of the forwarding route control. In this figure, grids with the same color indicate the sensor readings with the identical value. In Fig. 10(a), at the first sensing time, each mobile agent sends a packet using the lengthwise tree according to the procedure described in Section 5.3. In this phase, the agent at grid 6 sends the route fix message to its child, the agent at grid 1, because its sensor reading is identical with that of grid 6. On receiving this message, the agent at grid 1 fixes the route to the agent at grid 6. The agents in the other grids perform in the same way. As a result, the red-colored lengthwise routes as shown in Fig. 10(b) are fixed in this phase. At the second sensing time, each mobile agent without receiving route fix message switches its forwarding direction from lengthwise to crosswise, and sends a packet using the forwarding tree in Fig. 10(c). In this phase, the agent at grid 7 sends the route fix message to the agent at grid 6 because its sensor reading is identical with that of grid 7. The agent at grid 6 fixes the route to the agent at grid 7. After performing this procedure at other grids, the red-colored crosswise routes as shown in Fig. 10(d) are also fixed.

5.5 Restoring Sensor Readings at the Sink

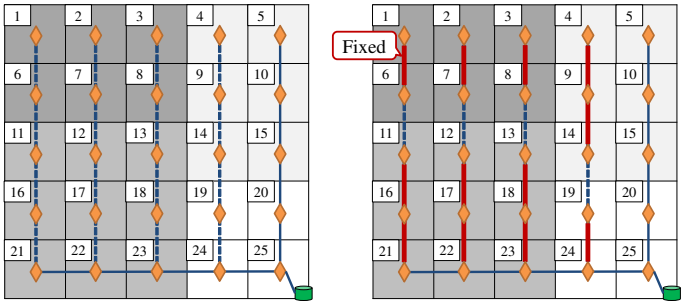
In DGUMA/DA, some sensor readings can be excluded from the packet. Thus, the sink needs to restore these excluded sensor readings.

In order to do this, the sink maintains the *tree information*, which stores the information on the topology of the forwarding tree. Note that the topology in the tree information is initially set as the lengthwise tree. The sink updates the tree information while restoring excluded sensor readings by referring to sensor data in the received (aggregated) packet. The detailed procedure is as follows:

(1) The sink extracts sensor readings with gridIDs from sensor data in the received packet and assigns them to the corresponding grids.

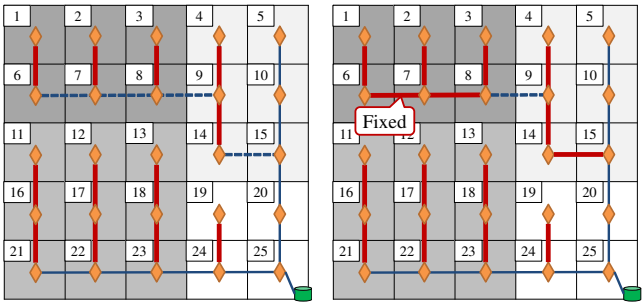
(2) For each grid without sensor reading, the sink assigns the sensor reading of the grid which is the child of the corresponding grid in the current tree information. This step is repeated until sensor readings are assigned to all grids.

(3) The sink recognizes that the routes have been fixed between grids where step (2) is applied. Thus, the sink fixes these routes in the tree information. For other routes, the sink switches the directions from lengthwise to crosswise or vise versa.



(a) Forwarding tree at the 1st sensing time.

(b) Route fixing after the 1st sensing time.



(c) Forwarding tree at the 2nd sensing time.

(d) Route fixing after the 2nd sensing time.

Fig. 10. An example of forwarding route control

(4) If the sensor readings become different between grids on a fixed route, the sink recognizes that the fixed route has been released. Thus, the sink stops fixing the corresponding route in the tree information, and switches the forwarding direction at the next sensing time.

According to the above procedure, the sink can recognize the topology of the forwarding tree and restore sensor readings at all grids.

Fig. 11 shows an example of the above procedure when the sink received sensor data in the environment shown in Fig. 8. First, the sink extracts sensor readings at grids {1, 6, 7, 9, 10, 12} from the received (aggregated) packet. Next, the sensor reading at grid 2 is restored by referring to its current child (i.e., grid 1). The sensor reading at grid 3 is also restored by referring to that at its current child (i.e., grid 2). In addition, The sink fixes the routes between these grids (i.e., from 1 to 2, and from 2 to 3), where the restoring process is applied. By applying this procedure for other grids {4, 5, 8, 11}, the sink can restore all the sensor readings. Note that, routes on the same column or row of the grid where the sink locates (i.e., from 7 to 10, from 12 to 10, and from 4 to 10) are not fixed.

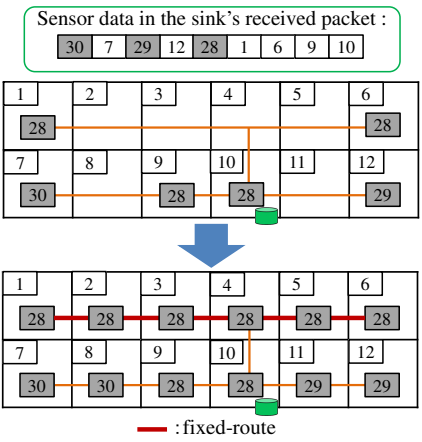
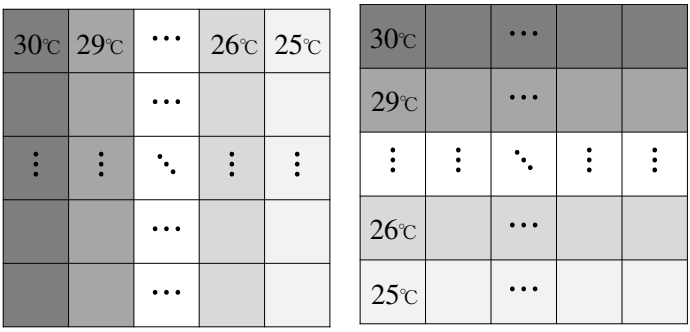


Fig. 11. An example of restoring sensor readings



(a) Lengthwise distribution (b) Crosswise distribution

Fig. 12. Geographical distribution of data values

6 Discussion

DGUMA/DA can reduce traffic for data gathering by data aggregation and forwarding route control. On the other hand, the forwarding route control generates the overhead since it needs to send route fix and fixed-route release messages. In this section, we discuss the performance gain (traffic reduction) and the overhead by the forwarding route control. Here, it is difficult to cover all situations of distribution of data values. In order to simplify the discussion, we assume a situation in which the distribution changes from Fig. 12(a) (*lengthwise distribution*) to Fig. 12(b) (*crosswise distribution*). In the lengthwise distribution, sensor readings can be aggregated efficiently without forwarding route control (only using the lengthwise tree). On the other hand, in the crosswise distribution, all routes between grids need to change (the forwarding tree needs to change to the crosswise tree) in order to efficiently aggregate sensor readings.

Table 2. The variables in this section

meaning of variable	variable
The size of packet header	s_{header}[B]
The size of sensor reading	s_{data}[B]
The size of gridID	s_{ID}[B]
The size of ACK	s_A[B]
The average distance of 1-hop transmission	l[m]
The average of the number of hops between the adjacent agents	h

In other words, the performance gain and the overhead generated by the forwarding route control become the largest in this situation.

For the discussion, we assume a D[m]×D[m] flatland as the sensing area. The sink divides the area into G lattice-shaped grids whose size is $D/\sqrt{G}$[m] × $D/\sqrt{G}$[m]. In addition, variables in Table 2 are used in this section. h in Table 2 is expressed by the following equation:

$$h = \begin{cases} 1 & (\frac{D}{\sqrt{G} \cdot l} \leq 1). \\ \frac{D}{\sqrt{G} \cdot l} & (Otherwise). \end{cases} \tag{2}$$

We assume that the mobile agent located in the grid where the sink exists can communicate directly to the sink. The sink locates at the grid of nth row and mth column.

Assuming the above situation, we calculate the following theoretical values:

- The overhead which is generated by changing the forwarding tree from Fig. 13(b) to Fig. 13(c).
- The traffic which is generated by data gathering in Fig. 13(a).
- The traffic which is generated by data gathering in Fig. 13(c).
- The traffic which is generated by data gathering in Fig. 13(b).

6.1 Overhead Generated by the Forwarding Route Control

We assume that all lengthwise routes are fixed in Fig. 13(b). In this case, all the fixed routes are released in the data gathering process with the lengthwise tree. The number of released routes is $(\sqrt{G}-1)^2$ because routes on nth row and those on mth column are not fixed. Considering that the size of a fixed-route release message is equal to s_{header}[B], and that the message is sent using the geo-routing protocol (shown in Fig 14), the traffic generated when releasing a route becomes $(s_{header} \cdot h + s_A)$[B]. Thus, the total traffic for releasing fixed routes, $S_{release}$, is expressed by the following equation:

$$S_{release} = (\sqrt{G} - 1)^2 (s_{header} \cdot h + s_A)\,[\mathrm{B}]. \tag{3}$$

At the next sensing time, the forwarding route becomes the crosswise tree. When gathering data with the crosswise tree, all routes are fixed except for those on nth row or mth column. Thus, the number of fixed routes is $(\sqrt{G} - 1)^2$. Considering that the size

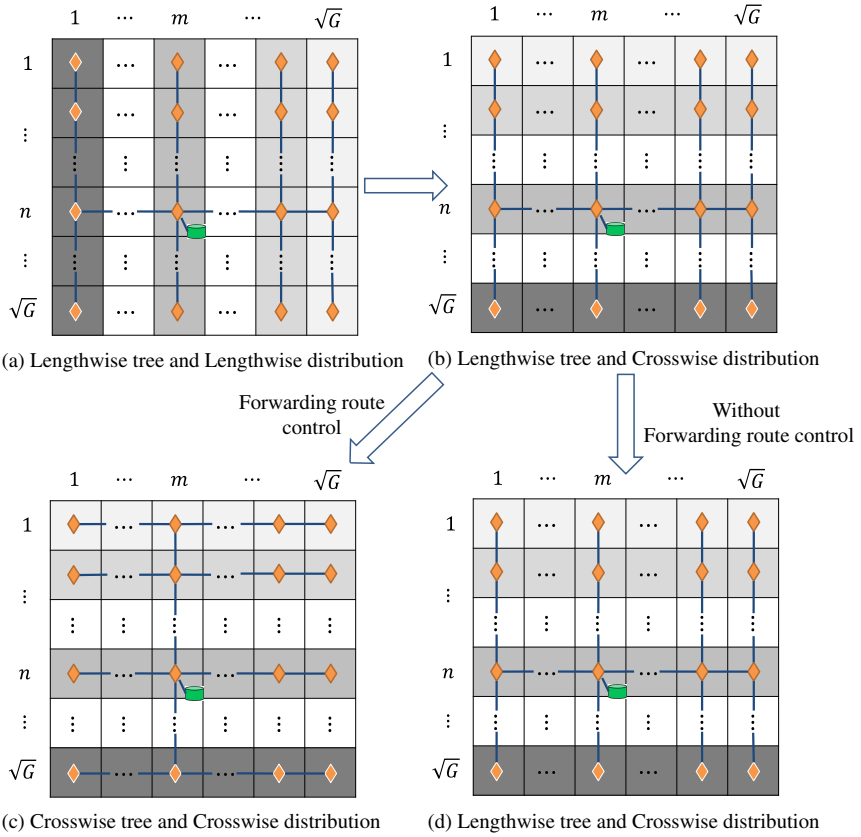


Fig. 13. Forwarding tree and distribution of data values

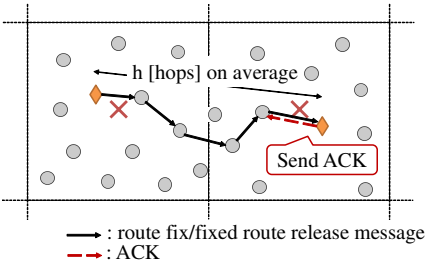


Fig. 14. An example of sending a message using the geo-routing

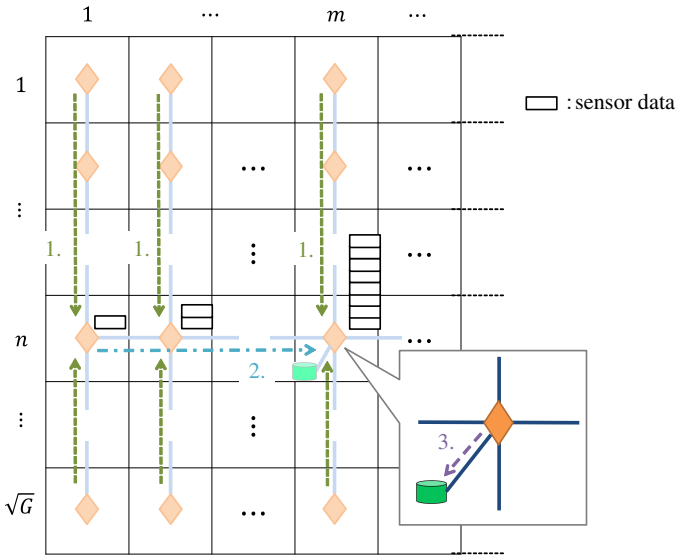


Fig. 15. An example of data gathering using lengthwise tree in the lengthwise distribution

of a route fix message is also equal to s_{header}[B], and that these messages are sent using the geo-routing protocol, the total traffic for fixing routes, S_{fix}, is expressed by the following equation:

$$S_{fix} = (\sqrt{G} - 1)^2(s_{header} \cdot h + s_A)\,[\mathrm{B}]. \tag{4}$$

As a result, the total overhead generated by the forwarding route control, $S_{overhead}$, is derived by the following equation:

$$S_{overhead} = S_{release} + S_{fix} = 2(\sqrt{G} - 1)^2(s_{header} \cdot h + s_A)\,[\mathrm{B}]. \tag{5}$$

6.2 Traffic for Data Gathering Using Lengthwise Tree in the Lengthwise Distribution

In this case, the data gathering tree can be treated as *optimized*. In order to derive the theoretical value of the traffic for data gathering, S_{opt}, we separately derive traffic in the following steps (see Fig. 15):

1. Lengthwise packet transmission (at every column).
2. Crosswise packet transmission (at nth row).
3. Packet transmission from the mobile agent to the sink at the grid where the sink locates.

STEP1. Lengthwise Packet Transmission: At first, a mobile agent located in a grid of the top or bottom edge in the sensing area starts to send its packet to its parent in the

forwarding tree. The size of this packet is $(s_{header} + s_{data} + s_{ID})$[B]. So, the traffic for delivering this packet to the parent becomes $(h \cdot (s_{header} + s_{data} + s_{ID}) + s_A)$[B].

The parent sends the packet to its parent without adding its sensor reading nor gridID because its own sensor reading is identical with that included in the received packet. Thus, the traffic for delivering this packet to the next parent becomes the same, that is, $(h \cdot (s_{header} + s_{data} + s_{ID}) + s_A)$[B].

This packet is delivered in the same way until a mobile agent at the grid on n-th row receives it. Since the number of mobile agents which send the packet is $(\sqrt{G} - 1)$ in a column, the total traffic generated at this column, $S_{opt,col}$, is expressed by the following equation:

$$S_{opt,col} = ((\sqrt{G} - 1)\{(s_{header} + s_{data} + s_{ID})h + s_A\})\,[\mathrm{B}].$$

Considering that there are $\sqrt{G}$ columns in the sensing area, the total traffic generated by mobile agents for this phase, $S_{opt,step1}$, is expressed by the following equation:

$$S_{opt,step1} = \sqrt{G} \cdot S_{opt,col} = (\sqrt{G}(\sqrt{G} - 1)\{(s_{header} + s_{data} + s_{ID})h + s_A\})\,[\mathrm{B}].$$

STEP2. Crosswise Packet Transmission (At *n*th Row): A mobile agent at the left end or the right end grid of nth row receives two packets from its children (at upper and lower grids). The agent sends the packet to its parent without adding its sensor reading nor gridID because its own sensor reading is identical with that included in the packet. Thus, the size of packet transmitted by the agent becomes $(s_{header} + s_{data} + 2s_{ID})$[B], and the traffic for delivering this packet to its parents becomes $(h \cdot (s_{header} + s_{data} + 2s_{ID}) + s_A)$[B].

On the other hand, mobile agents except for those at the left- and right-end grids receive multiple packets with different sensor readings, and aggregate them. Let us focus on the parent of the agent at the left end grid (shown in Fig. 16). This agent receives three packets from its children at the upper, the lower, and the left grids. Here, since sensor readings in the packets received from upper and lower grids are identical, the size of sensor data in the buffer is $(s_{data} + 2s_{ID})$[B] after aggregating these packets. On the other hand, since the sensor reading in the packet from the left grid is different from that in the buffer, sensor data whose size is $(s_{data} + 2s_{ID})$[B] is added to the buffer. In addition, the mobile agent adds only its own gridID after the gridIDs with the sensor reading which is identical with its own reading. As a result, the agent creates a packet whose size is $(s_{header} + 2(s_{data} + 2s_{ID}) + s_{ID})$[B]. Thus, the size of a new created packet increases by $((s_{data} + 2s_{ID}) + s_{ID})$[B] at the agent at each grid on nth row. Therefore, the size of packet transmitted by the agent at the kth grid from the left end becomes $(s_{header} + k(s_{data} + 2s_{ID}) + (k - 1)s_{ID})$[B], and the traffic for delivering this packet to its parent becomes $(h \cdot \{s_{header} + k(s_{data} + 2s_{ID}) + (k - 1)s_{ID}\} + s_A)$[B].

There are $(m - 1)$ grids from the left end grid to the mth grid. Thus, the total traffic generated at these grids, $S_{opt,step2(left)}$, is expressed by the following equation:

$$S_{opt,step2(left)} = (h \cdot \left\{(m-1)s_{header} + \sum_{k=1}^{m-1} k(s_{data} + 2s_{ID}) + \sum_{k=1}^{m-1}(k-1)s_{ID}\right\} + (m-1)s_A)\,[\mathrm{B}].$$

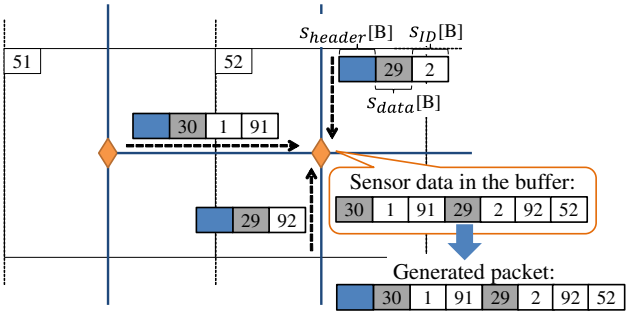


Fig. 16. An example of aggregation by the parent of the agent at the left-end grid in the lengthwise distribution (G=100, n=5, m=5)

In the same way, the total traffic generated at ($\sqrt{G} - m$) grids between the right end and mth grids, $S_{opt,step2(right)}$, is expressed by the following equation:

$$S_{opt,step2(right)} = (h \cdot \left\{ (\sqrt{G} - m)s_{header} + \sum_{k=1}^{\sqrt{G}-m} k(s_{data} + 2s_{ID}) + \sum_{k=1}^{\sqrt{G}-m} (k-1)s_{ID} \right\} + (\sqrt{G} - m)s_A)\,[\mathrm{B}].$$

STEP3. Packet Transmission from the Mobile Agent to the Sink at the Grid Where the Sink Locates: As shown in Fig. 17, the mobile agent at the grid where the sink locates records different $\sqrt{G}$ readings, 2 $\sqrt{G}$ gridIDs of the top and bottom grids in the sensing area, and ($\sqrt{G} - 2$) gridIDs of grids in nth row except for those of the left- and right-end grids, in the buffer. Thus, the size of the packet transmitted by this agent becomes ($s_{header} + \sqrt{G} \cdot s_{data} + (3\sqrt{G} - 2)s_{ID}$)[B]. Since we assume that this agent can directly communicate with the sink, the traffic generated at this grid, $S_{opt,step3}$, is expressed by the following equation:

$$S_{opt,step3} = (s_{header} + \sqrt{G} \cdot s_{data} + (3\sqrt{G} - 2)s_{ID} + s_A)\,[\mathrm{B}].$$

Consequently, the total traffic, S_{opt}, is expressed by the following equation:

$$\begin{aligned} S_{opt} &= S_{opt,step1} + S_{opt,step2(left)} + S_{opt,step2(right)} + S_{opt,step3} \\ &= h(G - \sqrt{G})(s_{data} + s_{ID}) + h(\sum_{k=1}^{m-1} k + \sum_{k=1}^{\sqrt{G}-m} k)(s_{data} + 2s_{ID}) \\ &+ h(\sum_{k=1}^{m-1}(k-1) + \sum_{k=1}^{\sqrt{G}-m}(k-1))s_{ID} + h(G-1)s_{header} \\ &+ s_{header} + \sqrt{G} \cdot s_{data} + (3\sqrt{G} - 2)s_{ID} + G \cdot s_A\,[\mathrm{B}]. \end{aligned} \quad (6)$$

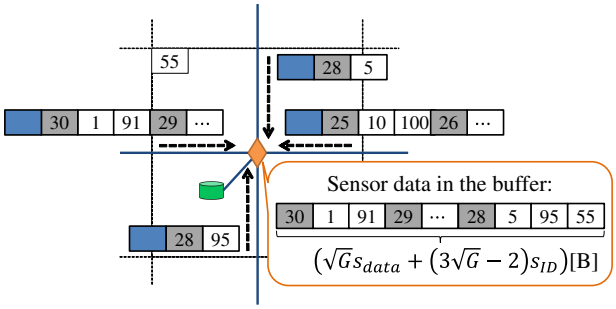


Fig. 17. An example of aggregation at the grid where the sink locates in the lengthwise distribution (G=100, n=5, m=5)

6.3 Traffic for Data Gathering Using Crosswise Tree in the Crosswise Distribution

In this case, the relation between the forwarding tree and the distribution of data values is the same as that in Section 6.2. Thus, the total traffic becomes S_{opt}, which is expressed by Eq.(6).

6.4 Traffic Generated by Data Gathering Using Lengthwise Tree in the Crosswise Distribution

In the same way as described in Section 6.2, in order to derive the theoretical value of the traffic for data gathering in this situation, S_{no_opt}, we separately derive traffic in the following steps (see Fig. 18):

1. Lengthwise packet transmission (at every column).
2. Crosswise packet transmission (at nth row).
3. Packet transmission from the mobile agent to the sink at the grid where the sink locates.

STEP1. Lengthwise Packet Transmission: At first, a mobile agent located in a grid of the top or bottom edge in the sensing area starts to send its packet to its parent in the forwarding tree. Since the size of this packet is $(s_{header} + s_{data} + s_{ID})$[B], the traffic for delivering this packet to the parent becomes $(h \cdot (s_{header} + s_{data} + s_{ID}) + s_A)$[B].

The parent adds its own sensor reading and gridID to the end of the received packet because its own sensor reading is different from that included in the received packet. Thus, the size of a new created packet increases by $(s_{data} + s_{ID})$[B] at every agent. Therefore, the size of packet transmitted by the agent at the kth grid from the grid of the top or bottom edge becomes $(s_{header} + k(s_{data} + s_{ID}))$[B], and the traffic for delivering this packet to its parent becomes $(h \cdot \{s_{header} + k(s_{data} + s_{ID})\} + s_A)$[B].

There are $(n - 1)$ grids from the top edge to the n-th row on a column. Thus, the total traffic generated at these grids becomes $(h \cdot \left\{(n-1)s_{header} + \sum_{k=1}^{n-1} k(s_{data} + s_{ID})\right\} +$

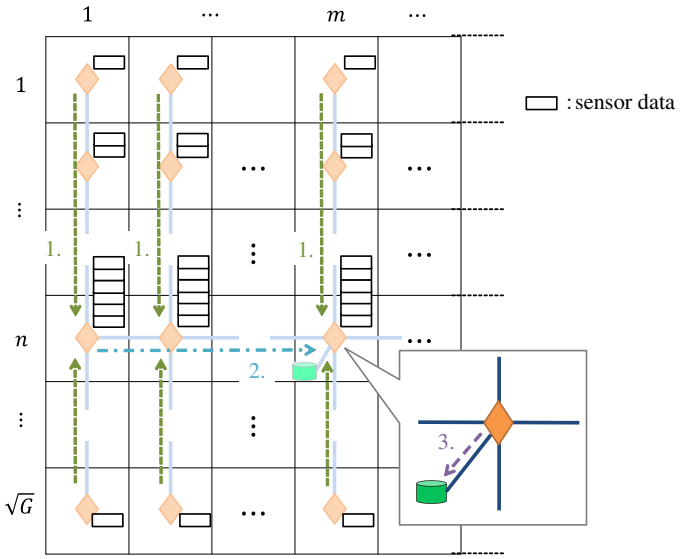


Fig. 18. An example of data gathering using lengthwise tree in the crosswise distribution

$(n-1)s_A)$[B]. In the same way, the total traffic generated at $(\sqrt{G}-n)$ grids between the bottom edge and nth row becomes $(h\cdot\left\{(\sqrt{G}-n)s_{header}+\sum_{k=1}^{\sqrt{G}-n}k(s_{data}+s_{ID})\right\}+(\sqrt{G}-n)s_A)$[B]. Therefore, the total traffic generated at a column, $S_{no\_opt,col}$, is expressed by the following equation:

$$S_{no\_opt,col}=(h\cdot\left\{(\sqrt{G}-1)s_{header}+(\sum_{k=1}^{\sqrt{G}-n}k+\sum_{k=1}^{n-1}k)(s_{data}+s_{ID})\right\}+(\sqrt{G}-1)s_A)\,[\mathrm{B}].$$

Considering that there are $\sqrt{G}$ columns in the sensing area, the total traffic generated by mobile agents in this phase, $S_{no\_opt,step1}$, is expressed by the following equation:

$$\begin{aligned}S_{no\_opt,step1} &= \sqrt{G}\cdot S_{no\_opt,col}\\ &= (\sqrt{G}\cdot h\left\{(\sqrt{G}-1)s_{header}+(\sum_{k=1}^{\sqrt{G}-n}k+\sum_{k=1}^{n-1}k)(s_{data}+s_{ID})\right\}\\ &+ \sqrt{G}(\sqrt{G}-1)s_A)\,[\mathrm{B}].\end{aligned}$$

At nth row, every agent receives the packet from the upper grid (the size is $(s_{header}+(n-1)(s_{data}+s_{ID}))$[B]) and lower grid (the size is $(s_{header}+(\sqrt{G}-n)(s_{data}+s_{ID}))$[B]). Thus, each agent stores sensor data whose size is $((\sqrt{G}-1)(s_{data}+s_{ID}))$[B] in its buffer.

STEP2. Crosswise Packet Transmission (At *n*th Row): A mobile agent at the left-end or the right-end grid of nth row first adds its own sensor reading and gridID to its

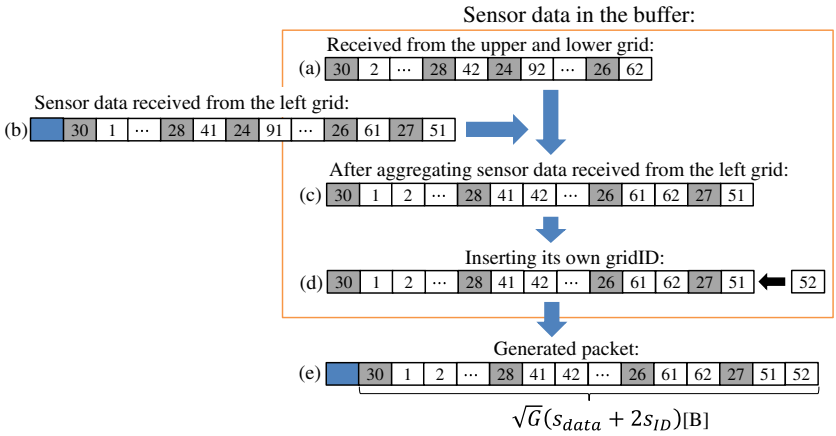


Fig. 19. An example of aggregation by the parent of the agent at the left-end grid in the crosswise distribution (G=100, n=5, m=5)

buffer because its own reading is identical to none of sensor readings in the buffer. Thus, the size of packet transmitted by the agent becomes $(s_{header} + \sqrt{G}(s_{data} + s_{ID}))$[B], and the traffic for delivering the packet to its parent becomes $(h \cdot \{s_{header} + \sqrt{G}(s_{data} + s_{ID})\} + s_A)$[B].

Every other agent on nth row also holds sensor data whose size is $((\sqrt{G} - 1)(s_{data} + s_{ID}))$[B] in the buffer. On receiving a packet from the child at nth row (from the left or right grid), an agent first merges the sensor data included in the packet with those in its buffer. Let us focus on the parent of the agent at the left-end grid (in Fig. 19). First, every sensor reading included in the packet from the left-end grid (i.e., {30, . . ., 28, 24, . . ., 26}, (b) in Fig. 19) except for the last one (i.e., 27) is included in the buffer ((a) in Fig. 19). Thus, according to step (2) in Section 5.3, only the gridIDs in the packet are added to the buffer ((c) in Fig. 19), and the size of sensor data in the buffer becomes $(\sqrt{G} \cdot s_{data} + (2\sqrt{G} - 1)s_{ID})$[B]. Second, according to step (3) in Section 5.3, the mobile agent adds only its own gridID after the gridIDs with the sensor reading which is identical with its own reading ((d) in Fig. 19). As a result, the agent creates a packet whose size is $(s_{header} + \sqrt{G} \cdot s_{data} + 2\sqrt{G} \cdot s_{ID})$[B] ((e) in Fig. 19). Thus, the size of a new created packet increases by $\sqrt{G} \cdot s_{ID}$[B] at the agent at each grid on nth row. Therefore, the size of packet transmitted by the agent at the kth grid from the left-end becomes $(s_{header} + \sqrt{G} \cdot s_{data} + k\sqrt{G} \cdot s_{ID})$[B], and the traffic for delivering this packet to its parent becomes $(h \cdot (s_{header} + \sqrt{G} \cdot s_{data} + k\sqrt{G} \cdot s_{ID}) + s_A)$[B]. Since there are $(m - 1)$ grids from the left-end grid to the mth grid, the total traffic generated at these grids, $S_{no\_opt,step2(left)}$, is expressed by the following equation:

$$S_{no\_opt,step2(left)} = (h \cdot \left\{(m-1)(s_{header} + \sqrt{G} \cdot s_{data}) + \sum_{k=1}^{m-1} k\sqrt{G} \cdot s_{ID}\right\} + (m-1)s_A)\ [\mathrm{B}].$$

In the same way, the total traffic generated at ($\sqrt{G} - m$) grids between the right end and mth grids, $S_{no\_opt,step2(right)}$, is expressed by the following equation:

$$S_{no\_opt,step2(right)} = (h \cdot \left\{ (\sqrt{G} - m)(s_{header} + \sqrt{G} \cdot s_{data}) + \sum_{k=1}^{\sqrt{G}-m} k \sqrt{G} \cdot s_{ID} \right\} + (\sqrt{G} - m) s_A) \, [\mathrm{B}].$$

STEP3. Packet Transmission from the Mobile Agent to the Sink at the Grid Where the Sink Locates: The mobile agent at the grid where the sink locates records different $\sqrt{G}$ readings, G gridIDs. Thus, the size of the packet transmitted by this agent becomes $(s_{header} + \sqrt{G} \cdot s_{data} + G \cdot s_{ID})$[B]. Thus, the traffic generated at this grid, $S_{no\_opt,step3}$, is expressed by the following equation:

$$S_{no\_opt,step3} = (s_{header} + \sqrt{G} \cdot s_{data} + G \cdot s_{ID} + s_A) \, [\mathrm{B}].$$

Consequently, the total traffic, S_{no_opt}, is expressed by the following equation:

$$\begin{aligned} S_{no\_opt} &= S_{no\_opt,step1} + S_{no\_opt,step2(left)} + S_{no\_opt,step2(right)} + S_{no\_opt,step3} \\ &= h \sqrt{G} \left\{ (\sum_{k=1}^{\sqrt{G}-n} k + \sum_{k=1}^{n-1} k)(s_{data} + s_{ID}) + (\sum_{k=1}^{\sqrt{G}-m} k + \sum_{k=1}^{m-1} k) s_{ID} + (\sqrt{G} - 1) s_{data} \right\} \\ &+ h(G-1) s_{header} + s_{header} + \sqrt{G} \cdot s_{data} + G \cdot s_{ID} + G \cdot s_A \, [\mathrm{B}] \end{aligned} \tag{7}$$

6.5 The Relation between the Performance Gain and the Overhead Generated by the Forwarding Route Control

In the situation discussed in this section, the performance gain by the forwarding route control becomes $(S_{no\_opt} - S_{opt})$[B], while the overhead $S_{overhead}$[B] is needed to reconstruct the forwarding tree. Assuming that $s_{header} = 21$[B], $s_{data} = 2$[B], $s_{ID} = 1$[B], $s_A = 5$[B], $D = 1,000$[m], $G = 100$, $l = r = 100$[m], $m = n = 5$, the performance gain and the overhead respectively, become 0.866[B] and 4.212[B]. This indicates that the overhead becomes larger.

However, when the distribution of data values does not change during successive T sensing times after changing from lengthwise to crosswise distributions, the performance gain becomes larger as T increases. We derive the minimum number of sensing times, T, in order for the performance gain to be larger than the overhead.

First, when the forwarding route control is not implemented, the total traffic for T times of data gathering becomes $T \cdot S_{no\_opt}$[B]. On the other hand, in DGUMA/DA, the total traffic for T times of data gathering becomes $T \cdot S_{opt}$[B]. Thus, in order for the performance gain to be larger than the overhead, T must be satisfied the following condition:

$$\begin{aligned} T \cdot S_{no\_opt} - T \cdot S_{opt} &> S_{overhead}. \\ T &> \frac{S_{overhead}}{S_{no\_opt} - S_{opt}}. \end{aligned} \tag{8}$$

Assuming the case described above, T must be larger than 4.863. This indicates that, in the situation discussed in this section, the performance gain becomes larger than the overhead generated by the forwarding route control when the distribution of data values does not change in successive five sensing times.

7 Simulation Experiments

In this section, we show the results of simulation experiments for validating the discussion in Section 6, and evaluating performance of DGUMA/DA. For the simulations, we used the network simulator, Scenargie 1.5[1].

7.1 Simulation Model

There are 2,000 mobile sensor nodes and a sink in a two-dimensional area of 1,000[m]×1,000[m] ($D = 1,000$). The sink is fixed of the point of (400[m], 400[m]) from the left and the bottom edges of the sensing area. Each sensor node moves according to the random waypoint mobility model with a home area [3]. Specifically, a grid is assigned to each node so that the number of nodes assigned to each grid becomes nearly equal. Each node initially located at its assigned grid, and randomly selects its destination in its assigned grid with the probability of 90%, or in the entire sensing area with the probability of 10%. After determining its destination, the node moves there at a constant speed uniformly determined within the range of [0.5, 1][m/sec]. After arriving at the destination, it stops there for 60[sec] before determining the next destination. The sink and sensor nodes communicate with IEEE 802.11p whose transmission rate is 3[Mbps] and communication range r is about 100[m] (the average distance of 1-hop transmission, l, is set to 100[m]). Each sensor node continuously senses the area. The sink divides the area into G ($10^2 \leq G \leq 15^2$) lattice-shaped grids whose size is $1,000/\sqrt{G}$[m] $\times$ $1,000/\sqrt{G}$[m], and sets the center point of each grid as a sensing point. The sink deploys a mobile agent at each sensing point when the simulation starts. The size of the agent data is set as 60[B], assuming that each sensor node has the source code of the mobile agent in advance. The sensing cycle is set as 30[sec]. Moreover, a mobile agent moves to the sensor node located closest to the sensing point when the distance between the sensing point and itself becomes longer than 47[m], which is set as an appropriate value according to our preliminary experiments, or the mobile agent is out of the valid sensing area. As the geographical distribution of data values, we used two different situations, lengthwise distribution as shown in Fig. 12(a) and the crosswise distribution as shown in Fig. 12(b). Table 3 shows the size of each message at the application layer. The size of a sensor reading, s_{data}, is set as 2[B], and that of a gridID, s_{ID}, is 1[B]. In Table 3, i denotes the number of sensor readings and j denotes the number of gridIDs.

[1] Scenargie1.5 Base Simulator revision 8217, Space-Time Engineer, https://www.spacetime-eng.com/

Table 3. Message size

Roll	Message name	Size[B]
Deploying a mobile agent	Deployment message	s_{header} + 60 = 21 + 60 = 81
Sending sensor data	Data message	$s_{header} + s_{data} \cdot i + s_{ID} \cdot j$ = 21 + 2·i + 1·j
Moving a mobile agent	Movement message	s_{header} + 60 = 81
Fixing a route	Route fix message	s_{header} = 21
Releasing a fixed route	Fixed-route release message	s_{header} = 21
ACK	ACK message	s_A =5

Table 4. m and n in a certain G

G	10^2	11^2	12^2	13^2	14^2	15^2
m	5	5	5	6	6	6
n	5	5	5	6	6	6

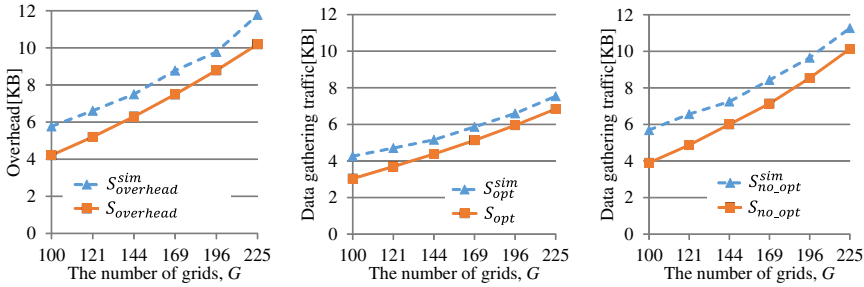
7.2 Validation of the Discussion in Section 6

First, in order to validate the discussion in Section 6, we simulated an environment in Section 6, and measured traffics for the forwarding route control and data gathering in DGUMA/DA. Specifically, we simulated the following three sensing times and measured traffics in each sensing time:

1. At the first sensing time, the data gathering is performed using the lengthwise tree in the lengthwise distribution. In this sensing time, the forwarding routes are fixed.
2. At the second sensing time, the distribution of data values changes from the lengthwise to the crosswise. In this sensing time, fixed routes are released. Let the traffics for data gathering and for releasing fixed routes at this sensing time be $S^{sim}_{no\_opt}$ and $S^{sim}_{release}$, respectively. These values respectively correspond to the theoretical values, S_{no_opt} and $S_{release}$.
3. At the third sensing time, the data gathering is performed using the crosswise tree in the crosswise distribution. In this sensing time, the forwarding routes are fixed. Let the traffics for data gathering and fixing routes at this sensing time be S^{sim}_{opt} and S^{sim}_{fix}, respectively. These values respectively correspond to the theoretical values, S_{opt} and S_{fix}.

Note that the theoretical values ($S_{overhead}$, S_{opt} and S_{no_opt}) are calculated from Eqs.(5), (6) and (7). Here, m and n (i.e., the coordinates of the grid where the sink exists) are respectively set according to the number of grids, G, as shown in Table 4. In the experiment, we have simulated the above situation 100 times, and derives the average of S^{sim}_{opt}, $S^{sim}_{no\_opt}$ and $S^{sim}_{overhead}$ (= $S^{sim}_{fix} + S^{sim}_{release}$).

Fig. 20 shows the experimental results and the theoretical values. The horizontal axis of all graphs is the number of grids, G. The vertical axis respectively indicates $S^{sim}_{overhead}$ and $S_{overhead}$ in Fig. 20(a), S^{sim}_{opt} and S_{opt} in Fig. 20(b), and $S^{sim}_{no\_opt}$ and S_{no_opt} in Fig. 20(c).



(a) $S^{sim}_{overhead}$ and $S_{overhead}$. (b) S^{sim}_{opt} and S_{opt}. (c) $S^{sim}_{no\_opt}$ and S_{no_opt}.

Fig. 20. Comparison of experimental results and theoretical values

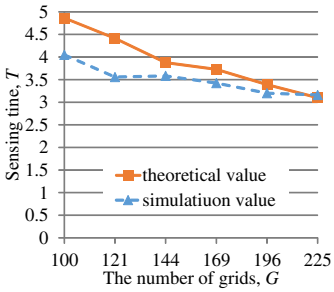


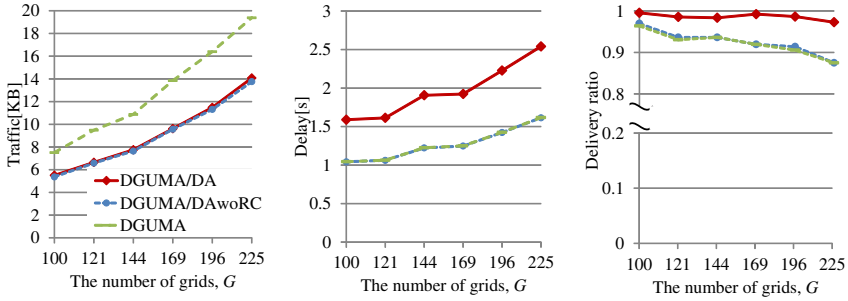
Fig. 21. The minimum number of sensing times in order for the performance gain to be larger than the overhead

From these results, we can see that the theoretical values show similar tendency as the experimental results. Here, the experimental results become larger than the theoretical values especially when G is small. This is mainly because the average of the number of hops in the simulation experiment becomes different from h calculated by Eq.(2).

In addition, we derived the minimum number of sensing times, T, in order for the performance gain to be larger than the overhead using the results in Fig. 20. Fig. 21 shows the result. The horizontal axis of this graph is G. We can see that the difference between the experimental results and the theoretical values become smaller as G increases. This is because the average of the number of hops in the simulation experiment becomes close to h as G increases.

7.3 Performance Evaluation of DGUMA/DA

Second, in order to verify the efficiency of DGUMA/DA, we evaluated the performances of DGUMA/DA and some other methods. For comparison, we evaluated the performances of DGUMA and DGUMA/DA without route control (DGUMA/DAwoRC for short), which only aggregates sensor data according to the procedure in Section 5.3



(a) Traffic. (b) Delay. (c) Delivery ratio.

Fig. 22. Effects of the number of grids (lengthwise distribution)

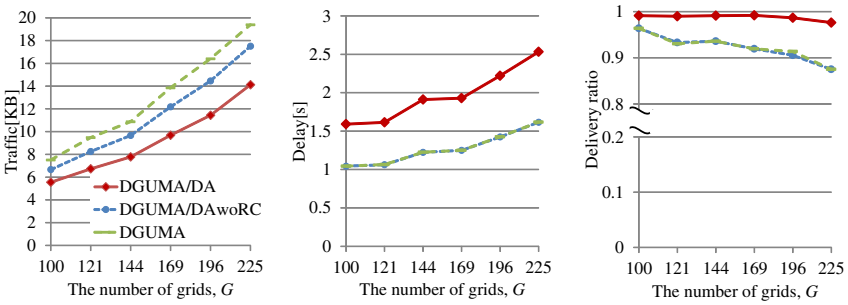
using the fixed (initial) forwarding tree. The simulation time is 3,600[sec] and we evaluated the following three criteria:

- **Traffic:** The traffic is defined as the average of the summation of the size of all packets sent by the sink and all sensor nodes between two consecutive sensing times.
- **Delay:** The delay is defined as the average elapsed time from the start of each sensing time to the time that the sink successfully receives sensor data.
- **Delivery ratio:** The delivery ratio is defined as the ratio of the number of sensor readings which the sink correctly restored to that observed in all grids.

We examined the effects of G, the number of grids. Figs. 22 and 23 show the simulation results. The horizontal axis of all graphs is the number of grids, G.

Lengthwise Distribution: Fig. 22(a) shows the traffic. From this result, we can see that the traffics in all methods increase as G increases. This is because the number of sensor data increases as G increases. We can also see that DGUMA/DA and DGUMA/DAwoRC can gather sensor data with less traffic than DGUMA. This is because DGUMA gathers all sensor data without aggregating them. The traffic in DGUMA/DA is almost same as that in DGUMA/DAwoRC in the lengthwise distribution. This is because the topology of the forwarding tree incidentally becomes suitable for data aggregation even in DGUMA/DAwoRC. Here, the traffic in DGUMA/DA is slightly larger than that in DGUMA/DAwoRC. This is because DGUMA/DA has to send messages to fix routes for data aggregation.

Fig. 22(b) shows the delay. From this result, we can see that the delays in all methods increase as G increases. This is obvious because the number of sensor data increases as G increases. We can also see that the delay in DGUMA/DA becomes longer than those in other methods. This is because mobile agents in DGUMA/DA have to wait until their timers expire before sending a packet, while they can send their packet immediately after receiving packets from all their child agents in other methods.



(a) Traffic. (b) Delay. (c) Delivery ratio.

Fig. 23. Effects of the number of grids (crosswise distribution)

Fig. 22(c) shows the delivery ratio. From this result, we can see that the delivery ratio in DGUMA/DA is higher than those in other methods. In DGUMA and DGUMA/DAwoRC, mobile agents cannot send their packet until receiving packets from all their children. Thus, no packet is sent to the sink once a packet collision occurs. On the other hand, thanks for introducing the timer, mobile agents in DGUMA/DA can send their packet even when packet collisions occur at their descendant. As G increases, the delivery ratio in all methods become lower. This is because a chance of packet collisions becomes larger due to the increase of the number of sensor data. Among the three methods, the delivery ratio in DGUMA/DA keeps high since mobile agents with the different distances from the sink sends their packet at different timings according to their timers. However, DGUMA/DA cannot completely eliminate packet collisions. This is because mobile agents which have almost the same distance to the sink send their packets at almost the same time when *rand* in Eq.(1) becomes very close between these agents.

Crosswise Distribution: Fig. 23(a) shows the traffic. From this result, we can see that the traffic in DGUMA/DA is still small even in the crosswise distribution, while the traffic in DGUMA/DAwoRC becomes much larger. This is because, in the crosswise distribution, less sensor data are aggregated on the forwarding tree with the initial (lengthwise) tree. On the other hand, DGUMA/DA appropriately changes the topology of the forwarding tree according to the geographical distribution of sensor data. Thus, more sensor data can be aggregated. As G increases, the difference in traffic increases between methods. This is because the number of sensor data increases as G increases.

Figs. 23(b) and 23(c), respectively, show the delay and the delivery ratio. These results are almost the same as in Figs. 22(b) and 22(c). This is because the differences in delay and delivery ratio between DGUMA/DA and other methods are caused not only by the difference of forwarding route, but also by the introduction of timer.

8 Conclusion

In this chapter, we have presented DGUMA/DA, which is a data gathering method considering geographical distribution of data values in dense MWSNs. DGUMA/DA

can reduce traffic for gathering sensor data by aggregating the same sensor readings and dynamically constructing the forwarding tree for data aggregation. The results of the simulation experiments show that DGUMA/DA can gather sensor data with high delivery ratio and small traffic.

In this chapter, we assume that DGUMA/DA gathers sensor readings which have only one attribute. However, in a real environment, it is possible that each sensor reading has multiple attributes (e.g., temperature and light intensity). Therefore, it is necessary to extend DGUMA/DA in order to efficiently gather sensor readings with multiple attributes. In addition, DGUMA and DGUMA/DA do not consider erroneous or missing data. Thus, it is necessary to extend DGUMA in order to handle them.

Acknowledgments. This research is partially supported by the Grant-in-Aid for Scientific Research (S)(21220002), (B)(24300037) of MEXT, and for Young Scientists (B)(23700078) of JSPS, Japan.

References

1. Ali, A., Khelil, A., Szczytowski, P., Suri, N.: An adaptive and composite spatio-temporal data compression approach for wireless sensor networks. In: Proc. Int. Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2011), pp. 67–76 (2011)
2. Burke, J., Estrin, D., Hansen, M., Parker, A., Ramanathan, N., Reddy, S., Srivastava, M.B.: Participatory sensing. Proc. Int. Workshop on World-Sensor-Web (WSW) at Embedded Networked Sensor Systems (Sensys) (2006)
3. Camp, T., Belong, J., Davies, V.: A survey of mobility models for ad hoc network research. Wireless Communications and Mobile Computing 2(5), 483–502 (2002)
4. Campbell, A.T., College, D., Eisenman, S.B., Lane, N.D., Miluzzo, E., Peterson, R.A., Lu, H., Zheng, X., Musolesi, M., Fodor, K., Ahn, G.S.: The rise of peple-centric sensing. IEEE Internet Computing 12(4), 12–21 (2008)
5. Di Francesco, M., Das, S.K., Anastasi, G.: Data collection in wireless sensor networks with mobile elements: a survey. ACM Transactions on Sensor Networks 8(1), 1–34 (2011)
6. Goto, K., Sasaki, Y., Hara, T., Nishio, S.: Data gathering using mobile agents in dense mobile wireless sensor networks. In: Proc. Int. Conf. on Advances in Mobile Computing and Multimedia (MoMM), pp. 58–65 (2011)
7. Heissenbüttel, M., Braun, T., Bernoulli, T., Wälchli, M.: BLR: beacon-less routing algorithm for mobile ad hoc networks. Computer Communications 27(11), 1076–1086 (2004)
8. Landsiedel, O., Götz, S., Wehrle, K.: Towards scalable mobility in distributed hash tables. In: Proc. Int. Conf. on Peer-to-Peer Computing (P2P), pp. 203–209 (2006)
9. Lane, N.D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., Campbell, A.T.: A survey of mobile phone sensing. IEEE Communications Magazine 48(9), 140–150 (2010)
10. Luo, C., Wu, F., Sun, J., Chen, C.: Compressive data gathering for large-scale wireless sensor networks. In: Proc. Int. Conf. on Mobile Computing and Networking (MobiCom), pp. 145–156 (2009)
11. Luo, H., Liu, Y., Das, S.K.: Routing correlated data in wireless sensor networks: a survey. IEEE Network 21(6), 40–47 (2007)
12. Matsuo, K., Goto, K., Kanzaki, A., Hara, T., Nishio, S.: Data gathering considering geographical distribution of data values in dense mobile wireless sensor networks. In: Proc. Int. Conf. on Advanced Information Networking and Applications (AINA), pp. 445–452 (2013)

13. Pattem, S., Krishnamachari, B., Govindan, R.: The impact of spatial correlation on routing with compression in wireless sensor networks. ACM Trans. Sensor Networks 4(4), 28–35 (2008)
14. Reddy, S., Samanta, V., Burke, J., Estrin, D., Hansen, M., Strivastava, M.: Examining micropayments for participatory sensing data collections. In: Proc. Int. Conf. on Ubiquitous Computing (UBICOMP), pp. 33–36 (2010)
15. Reddy, S., Samanta, V., Burke, J., Estrin, D., Hansen, M., Strivastava, M.: Mobisense-mobile network services for coordinated participatory sensing. In: Proc. Int. Symposium on Autonomous Decentralized Systems (ISADS), pp. 231–236 (2009)
16. Sharaf, M.A., Beaver, J., Labrinidis, A., Chrysanthis, P.K.: TiNA: a scheme for temporal coherency-aware in-network aggregation. In: Proc. Int. Workshop on Data Engineering for Wireless and Mobile Access (MobiDE), pp. 66–79 (2003)
17. Shi, J., Zhang, R., Liu, Y., Zhang, Y.: Prisense: privacy-preserving data aggregation in people-centric urban sensing systems. In: Proc. Int. Conf. on Computer Communications (INFOCOM), pp. 758–766 (2010)
18. Umer, M., Kulik, L., Tanin, E.: Optimizing query processing using selectivity-awareness in wireless sensor networks. Computers, Environment and Urban Systems 33(2), 79–89 (2009)
19. Weng, H., Chen, Y., Wu, E., Chen, G.: Correlated data gathering with double trees in wireless sensor networks. IEEE Sensors Journal 12(5), 1147–1156 (2012)
20. Yick, J., Mukherjee, B., Ghosal, D.: Wireless sensor network survey. Computer Networks 52(12), 2292–2330 (2008)
21. Zeydan, E., Kivanc, D., Comaniciu, C., Tureli, U.: Energy-efficient routing for correlated data in wireless sensor networks. Ad Hoc Networks 10(6), 962–975 (2012)
22. Zhang, C.: Cluster-based routing algorithms using spatial data correlation for wireless sensor networks. Journal of Communications 5(3), 232–238 (2010)
23. Zhang, J., Wu, Q., Ren, F., He, T., Lin, C.: Effective data aggregation supported by dynamic routing in wireless sensor networks. In: Proc. Int. Conf. on Communications (ICC), pp. 1–6 (2010)
24. Zhu, Y., Vedantham, R., Park, S.J., Sivakumar, R.: A scalable correlation aware aggregation strategy for wireless sensor networks. Information Fusion 9(3), 354–369 (2008)