Impact of Aggregation Efficiency on GIT Routing for Wireless Sensor Networks

Takafumi Aonishi, Takashi Matsuda
Graduate School of Science and Technology, Kobe University
1-1 Rokkodai, Nada, Kobe, Hyogo 657-8501, JAPAN
tkfm@cs28.cs.kobe-u.ac.jp

Shinji Mikami
Graduate School of Natural Science and Technology, Kanazawa University
Kakuma, Kanazawa, Ishikawa 920-1192, JAPAN

Hiroshi Kawaguchi, Chikara Ohta, Masahiko Yoshimoto
Faculty of Engineering, Kobe University
1-1 Rokkodai, Nada, Kobe, Hyogo 657-8501, JAPAN

Abstract

In most research work for sensor network routings, perfect aggregation has been assumed. Such an assumption might limit the application of the wireless sensor networks. We address the impact of aggregation efficiency on energy consumption in the context of GIT routing. Our questions are how the most efficient aggregation point changes according to aggregation efficiency and the extent to which energy consumption can decrease compared to the original GIT routing and opportunistic routing. To answer these questions, we analyze a two-source model, which yields results that lend insight into the impact of aggregation efficiency. Based on analytical results, we propose an improved GIT: "aggregation efficiency-aware GIT", or AGIT. We also consider a suppression scheme for exploratory messages: "hop exploratory." Our simulation results show that the AGIT routing saves the energy consumption of the data transmission compared to the original GIT routing and opportunistic routing.

1 Introduction

Sensor networks are expected to operate under severe energy constraints because it is not practical to replace their batteries because of the large number of sensor nodes. A salient issue is reduction of the amount of transmitted data because wireless communications at sensor nodes consume more power than any other activity[8, 9, 10, 11, 12].

Data centric routing is a promising paradigm for sensor network routing[10]. With data centric routing, routing decisions are based on the contents of the payloads of packets rather than their destination addresses. A sensor node might aggregate receiving packets that are temporally buffered, generate a new packet, and then send it to the next hop. Such a means of operation is expected to reduce the amount of transmitted data, engendering remarkable power savings. An example of data centric routing is directed diffusion (DD)[7].

In most studies, perfect aggregation has been assumed (e.g. [3, 5, 8, 9]). In this case, the most efficient data paths from sources to a sink form a Steiner tree and/or minimal spanning tree. This fact encourages research of heuristic distributed algorithms such as Greedy Incremental Tree (GIT)[8] and the Nearest Neighbor Tree (NNT)[9]. However, perfect aggregation is not universal and possibly limits applications of sensor networks, as mentioned above. Unfortunately, we do not have sufficient insight into the influence of the diversity of the aggregation to sensor network routings.

In this work, we address the impact of aggregation efficiency on the energy consumption in the context of the GIT routing[8]. The original GIT routing is a heuristic algorithm to find a Steiner tree on a hop-count basis. Our questions are how the most efficient incremental aggregation point changes according to aggregation efficiency and how much energy consumption can decrease compared to the original GIT routing. We analyze a simple two-source model to answer these fundamental questions. Based on results of our analysis, We improve the GIT routing algorithm to find a more efficient aggregation point according to aggregation efficiency. In this paper, we call the improved GIT "aggregation efficiency-aware GIT (AGIT)."

This paper is organized as follows: Section 2 describes the original GIT algorithm. In Section 3, we analyze a simple two-source model to investigate the impact of the efficiency of aggregation on energy consumption. In Section 4, we propose AGIT routing. Section 5 shows some simulation results. Finally, we present conclusions in Section 6.

2 Greedy Incremental Tree

2.1 Directed Diffusion

GIT routing is based on directed diffusion (DD), which is a typical data-centric routings for sensor networks. Before describing GIT routing, we briefly explain DD.

In DD, a task described as a list of attribute-value pairs is flooded into a network as an *interest*. Through the interest diffusion process, a sensor node receives the interest sets (or updates) a *gradient* toward the neighbor which sends the interest, and resends the interest to some subset of its neighbors (or broadcast) if it is different from the previously received one. A sensor node to take the task described in the interest sends an *exploratory* message to each neighbor to whom a gradient is set. Intermediate nodes relay the exploratory message toward the sink along gradients of the interest to match the task of the exploratory message. Because the sink possibly receives multiple exploratory messages originating at a source from its neighbors, it reinforces a preferable path by sending reinforcing messages to particular ones among the neighbors from which it received an exploratory message. Intermediate nodes receiving this reinforcing message treat it similarly, so that it is relayed in the reverse direction on the path. As a result, a data path is established from the source to the sink. Refer to [7] for more detail on DD.

2.2 Finding of Aggregation Point in GIT routing

In fact, GIT routing is a heuristic distributed algorithm to construct a Steiner tree on a hop-count basis. Also, GIT routing assumes perfect aggregation. Each source, one by one, tries to find the shortest hop from itself to the existing path tree or the sink.

To realize this process, each exploratory message in GIT routing involves an additional attribute E, which denotes the additional cost (hop-count) from the source originating itself to the current node. The value of E is set to zero initially. Whenever resending an exploratory message, the nodes increment the value of E by one. The exploratory message is distributed through the network according to the gradient of the corresponding interest; it will arrive at nodes on the existing path tree. Consequently, the nodes on the existing path tree can know the hop-count from the source that initiated the exploratory message.

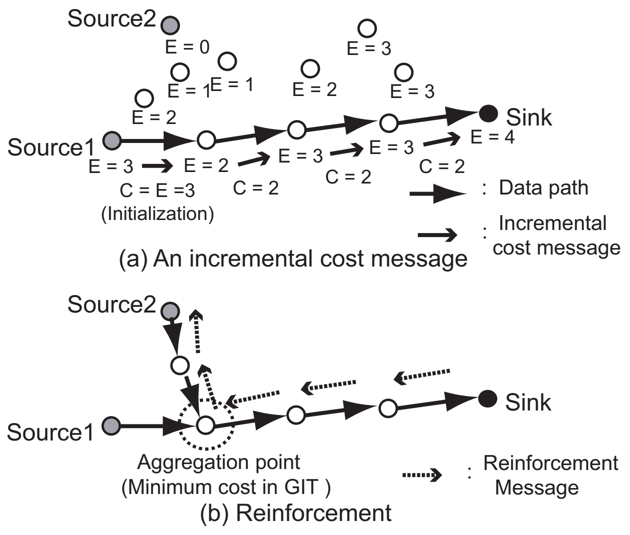

Figure 1. An example of path establishment in GIT routing

Each source involved in the existing path tree initiates an incremental cost message whenever it receives a previously unseen exploratory message that was initiated by other sources. The incremental cost message conveys two additional attributes: the random identifier of its corresponding exploratory message and the cost (hop-count) from the additional source (which initiated the exploratory message) to the existing path. The incremental cost message is relayed on the existing path from its originating source to the sink. The intermediate nodes update if the value of C in the incremental cost message is greater than or equal to the cached value of E.

The sink waits directly and late-arriving exploratory messages and other incremental cost messages for the predefined interval immediately after the arrival of the first incremental cost message. Then, the sink reinforces a neighbor, which sends an exploratory message or an incremental cost message with a lower additional energy cost C or E, respectively. In the case where an incremental cost message has the lowest additional energy cost C, the reinforcing message containing the value of C travels toward to the initiator of the incremental cost message on the existing established path until it encounters an intermediate node with $E = C$. This intermediate node becomes the aggregation point for the additional source that initiated the exploratory message. Then, the reinforcing message is diverted to the additional source.

As a result of the procedure described above, the lowest cost (minimal hop-count) branch is added to the existing path tree. Refer to [8] for more details regarding GIT routing.

2.3 Discussion

In the case of perfect aggregation, the energy consumption for data transmission on the newly added branch can be regarded as the net increase of that on the entire data path. This fact, however, is not always true in different aggregation schemes.

Let us consider the case where a packet has size $L_{\text{packet}} = L_{\text{header}} + L_{\text{payload}}$, where L_{header} is the header length and L_{payload} is the payload length in bytes and N packets are incoming and one packet is outgoing at an aggregation point. In the case of the perfect aggregation, the outgoing packet after aggregation has the same size $L_{\text{header}} + L_{\text{payload}}$ as an incoming packet. On the other hand, in the case of linear aggregation, the outgoing packet has larger size $L_{\text{header}} + N \times L_{\text{payload}}$ than that of perfect aggregation. Consequently, the linear aggregation consumes more energy on the path from the aggregation point to the sink than the perfect aggregation, thereby implying that a node near to the sink on the path tree might be more efficient as the aggregation point. But, how much energy can be saved if we choose the aggregation point more carefully? We analyze a simple two-source model in the next section to estimate the possible improvement.

3 Analysis of a Two-Source Model

This section shows the analysis for the simple two-source model. We also have the result for the three-source model, but we omit it because of space limitations.

3.1 Model Description

Figure 2 shows the two-source model that we analyze. In this model, we assume that the nodes exist densely. In this figure, however, we show only two sources – one sink and one aggregation point – for simplicity. The aggregation point is denoted as "p" in the figure. From the above assumption, the hop-count between two nodes can be proportional to Euclidean distance between them. In this model, we assume that the distance between a source and the sink is equal to one. We also assume that the energy consumption to transfer a data packet per hop is proportional to the packet size. Furthermore, we assume that the path between the first source and the sink is an existing path and that the second source is going to establish the path. Note that, in the case of the original GIT routing, the second source will have a perpendicular line as the additional path to the existing path.

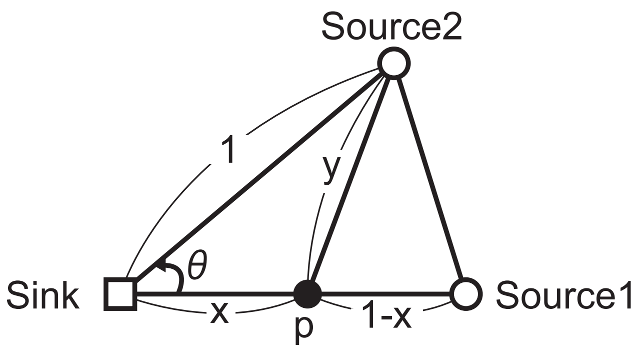

Figure 2. Two-source model.

Here we introduce the following notations: Let x and y respectively denote the distances between the aggregation point and the sink and the distance between source 2 to the aggregation point, $(0 \leq x \leq 1)$. We denote by θ the angle between source 1 to source 2, as seen from the sink $(0^\circ \leq \theta \leq 90^\circ)$. Let r denote the aggregation ratio of the size of the aggregated packet to the total size of the original packets $(\frac{1}{2} \leq r \leq 1)$. In the case of the perfect aggregation, the value r of the aggregation ratio is equal to $\frac{1}{2}$. Let E denote the energy consumed to transfer the data packets from the sources to the sink on the path tree.

3.2 Aggregation Point and Energy Consumption

The above assumptions suggest the following relationship:

$$E \propto 2rx + (1 - x) + y, \tag{1}$$

where

$$y = \sqrt{x^2 - 2x\cos\theta + 1}. \tag{2}$$

By performing some algebra for $\frac{dE}{dx} = 0$, we obtain the value x' to minimize the energy consumption for data packet transmission on the path tree:

$$x' = \begin{cases} 0, & \frac{\cos\theta+1}{2} \leq r < 1, \\ \cos\theta - \sin\theta\sqrt{\frac{1}{4r(1-r)} - 1}, & \frac{1}{2} \leq r < \frac{\cos\theta+1}{2}. \end{cases} \tag{3}$$

By substituting $x = x'$ in (1), we have the scaled value E' of the energy consumption for data packet transmission on the path tree in the case of the efficiency aggregation point. We have the value E_{GIT} expected for GIT routing, as

$$E_{\text{GIT}} \propto 2r\cos\theta + (1 - \cos\theta) + \sin\theta. \tag{4}$$

In the case of $r = 0.5$,

$$E_{\text{GIT}} \propto 1 + \sin\theta. \tag{5}$$

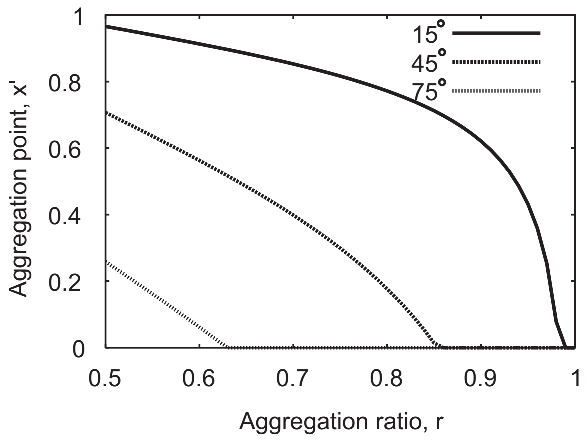

Figure 3. Aggregation point in two-source model.

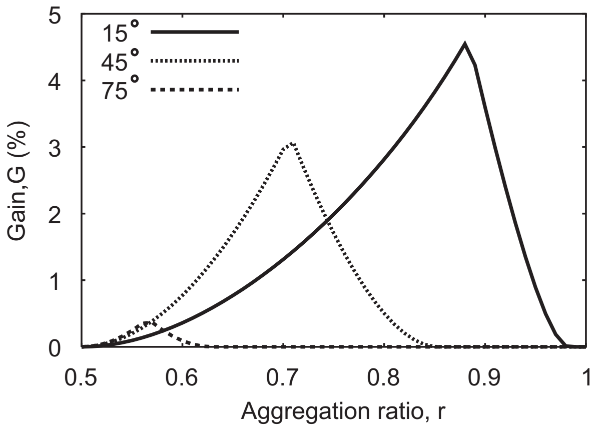

Figure 4. Gain by aggregation point in a two-source model.

Since the aggregation point becomes nearer to the sink, the path tree will become similar to that of "opportunistic routing"[8], where data from different sources can be opportunistically aggregated at intermediate nodes along the established paths. In the case of $x = 0, y = 1$, we have the value E_{opp} expected for opportunistic routing, as

$$E_{\mathrm{opp}} \propto 2. \tag{6}$$

To evaluate how much the aggregation point saves energy compared to the original GIT routing and opportunistic routing, we introduce the following metric, "*gain*", G:

$$G(r, \theta) = \frac{\min(E_{\mathrm{GIT}}, E_{\mathrm{opp}}) - E'}{\min(E_{\mathrm{GIT}}, E_{\mathrm{opp}})} \times 100. \tag{7}$$

3.3 Numerical Results

Figure 3 shows how the aggregation point changes according to the values of the aggregation ratio and the angle between the first and second sources. Figure 4 shows how much gain can be achieved.

From Fig. 3, we can see that the aggregation point changes widely according to the value of r as the angle becomes narrower. Furthermore, the aggregation point becomes nearer to the sink compared to the foot of perpendicular from the additional source to the existing path in the case of $\frac{1}{2} < r$.

Fig. 4 shows that the larger the aggregation ratio is (in other words, the smaller aggregation efficiency is), the larger the gain is obtained by choosing the optimal aggregation point. This tendency becomes noticeable in the case where the angle is around $45°$. Figure 4 shows that the value of gain has a peak in the middle region of r, and the larger the peak value is (up to 4.5% for $15°$), the smaller the angle is. The value r to give the peak gain increases as the angle decreases. That is, the AGIT routing is more effective in the case where sources exist near and the aggregation efficiency is not so high. The value of gain converges to zero toward to the both ends. This is because the path tree becomes similar to that of the GIT routing for $r = 0.5$ and that of the opportunistic routing for $r = 1$.

Although we do not show the results of the three-source model, more gain is obtained compared to the two-source model.

4 Aggregation Efficiency-Aware GIT

In this section, we propose "aggregation efficiency-aware GIT (AGIT)" routing in order to find a more efficient aggregation point to reduce the energy consumption inherent in transmitting data packets.

4.1 Suppression of Exploratory Messages

In the DD, which is the basis of GIT routing, exploratory messages are distributed widely according to the nodes' gradients because interests do not contain any information about a sink. As a result, the gradients are set in many directions. (See Section 2.2.2 in [7].)

To some extent, GIT-like routing necessarily distributes exploratory messages in order to determine the aggregation point for the existing path tree. Results of our analysis showed, however, that the aggregation point becomes nearer to the sink than the foot of the perpendicular from the additional source to the existing path in the case of $\frac{1}{2} < r$.

In the AGIT routing, we consider the following scheme to suppress the excessive exploratory messages: "hop exploratory." In the following, we assume that each node can know the hop-count from the sink through interest dissemination. Each node caches the hop-count from the sink for each interest as "own_hop." To do so, we also assume that each interest has a random identifier to be distinguished from the others.

4.1.1 Hop Exploratory

Each exploratory message contains the additional field '$previous\_hop$" to store the value own_hop of its sender's. In addition, each exploratory message also contains the field "hop" to store the hop count from the source that initiated the exploratory message. Whenever a source initiates the exploratory message with both $previous\_hop$ and hop set to own_hop.

When the node receives the exploratory message with $previous\_hop$, it rebroadcasts the exploratory messages with $previous\_hop$ set to own_hop and with hop decremented by one if

$$own\_hop \leq previous\_hop \quad \text{and} \quad hop > 0. \tag{8}$$

Figure 6 shows the phenomenon of dissemination of the exploratory messages, where an arrow denotes the direction in which an exploratory message is sent. From this figure, we can see that this scheme prevents network-wide diffusion compared to traditional scheme in Fig. 5, which indicates the dissemination of exploratory messages using original scheme described in [7].

4.2 Adjustment of the Incremental Cost Message Phase

The above suppression scheme involves some adjustments of the incremental cost message phase because the source nodes on the existing path tree might not receive the exploratory messages. Consequently, the incremental cost message is issued in such a case.

We take the following approach to overcome this problem. The intermediate nodes aside from the sources on the existing path tree can initiate the incremental cost message. In order to suppress the multiple incremental cost message, the more distant intermediate node from the sink issues the incremental cost message earlier. To do so, each intermediate node sets up an incremental cost message timer as

$$t_i = (max\_hop - own\_hop) \times \delta, \tag{9}$$

where max_hop and δ respectively denote the predefined network diameter and the timer granularity. The intermediate node issues the exploratory message if its timer expires before receiving another exploratory message; otherwise it suspends the issue.

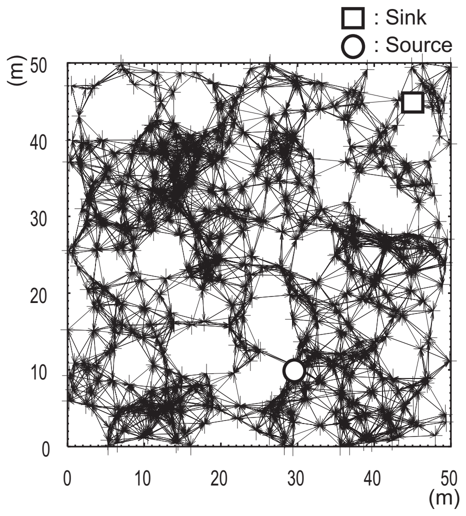

Figure 5. Phenomenon of dissemination of exploratory messages in the traditional approach.

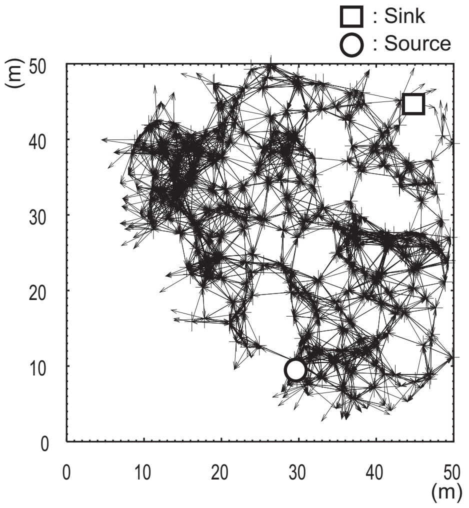

Figure 6. Phenomenon of dissemination of exploratory messages in the hop approach.

4.3 Finding of Aggregation Point in AGIT routing

In the following, we assume that linear aggregation is employed, whereby a packet has size $L_{\text{packet}} = L_{\text{header}} + L_{\text{payload}}$ where L_{header} is the header length and L_{payload} is the payload length in bytes. Furthermore, we assume that all sources send the data packet at the same rate. The procedure shown here can be extended easily to function in different cases.

In the AGIT routing, the incremental cost message contains an additional field to store the hop-count H from an interim aggregation point. Whenever the source and/or the intermediate nodes issue a new incremental cost message, they set $H = 1$.

The intermediate nodes receiving the incremental cost message execute the following:

$$\begin{array}{ll} \text{if}(E \leq C + H \cdot d) & C = E, \quad H = 1, \\ \text{else} & H = H + 1, \end{array} \tag{10}$$

where $d = L_{\text{payload}}/L_{\text{packet}}$. Recall that E denotes the additional cost (hop-count) from the source joining to the existing path tree to the current node.

In (10), $C + H \cdot d$ represents the net increase of power consumption from the source nodes to the current node when using the current interim aggregation point. If this value is greater than or equal to the value of E, the current node is more efficient than the interim aggregation point. In such a case, the current node substitutes for the interim aggregation point, so that it sets $C = E$ and $H = 1$. Otherwise, it increments the value of H by one.

Figure 7 shows the search procedure of the efficient aggregation point. Here we assume that the packet length is one and the payload length is 0.6. The first source receives the exploratory message from the second source, sets $C = E = 3$ and $H = 1$ in it, and then forwards it the neighbor node on the path from the first source and the sink. The neighbor receiving it compares the value of E and the value of $C + H \cdot d$, where $E = 3$, $C = 2$, $H = 1$ and $d = 0.6$. In this case, the node merely increments the value of H, and forwards the incremental cost message to its neighbor to the sink. This manner is repeated until the incremental cost message arrives at the sink.

The overhead of AGIT routing compared to the original GIT routing merely comprises the hop-count field to store H; it can be negligible.

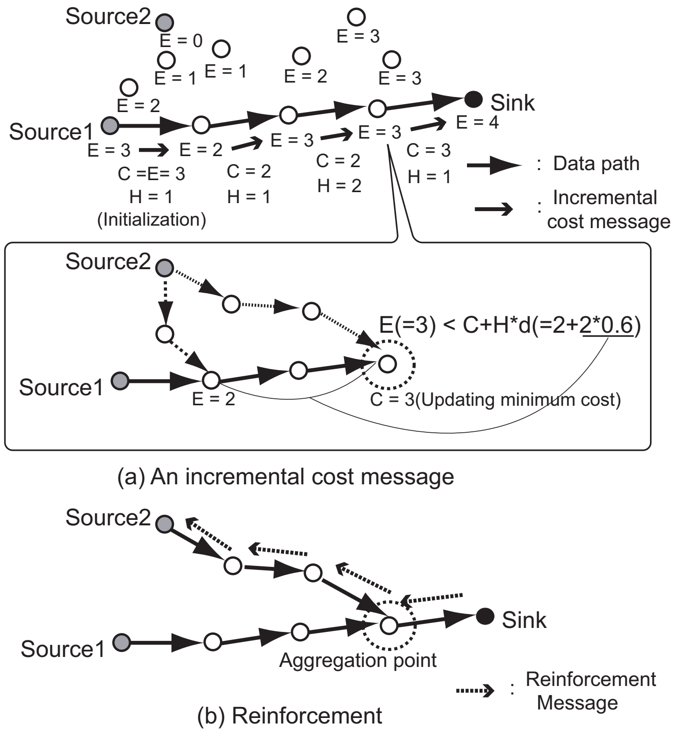

Figure 7. An Example of Path Establishment in AGIT

5 Simulation

In this section, we briefly explain our simulation conditions; then we show some simulation results. The aim of the simulation experiments is to confirm the effectiveness of the AGIT routing in more complicated situations.

5.1 Model and Assumption

We implemented the original GIT routing, opportunistic routing and the AGIT routing on a self-developed event-driven simulator engine.

In this simulator, 500 sensor nodes are deployed randomly in a 50×50 m^2 field. The transmission range is 5 m. One sink is located at $(45, 45)$ of the two-dimensional coordinate. The number of sources is varied from two to ten; they are arranged randomly in the field.

The packet has a 36-byte header. The payload length is varied as 4, 36, 108, and 216 bytes.

We implemented two schemes of the dissemination of the exploratory messages: "traditional exploratory" and "hop exploratory."

We implemented the ideal media access control (MAC) on our simulator, where no collisions occur.

Assuming the case by which the pass loss coefficient of $n = 2$, we modeled the energy consumption for transmission and reception of the packet of length l bits with distance R m, E_{tx} and E_{rx}, as follows:

$$E_{\text{tx}} = (\alpha_{\text{tx}} + \beta \cdot R^2) \cdot l, \tag{11}$$
$$E_{\text{rx}} = \alpha_{\text{rx}} \cdot l, \tag{12}$$

where α_{tx} and α_{rx} respectively denote the energy consumptions of the transmission circuit and the reception circuit, expressed as nanojoules per bit, and β denotes the radiation energy in appropriate units (nJ/bit/m^2)[4].

In simulation experiments, we use $\alpha_{\text{tx}} = 50$ nJ/bit, $\alpha_{\text{rx}} = 300$ nJ/bit, and $\beta = 1.6$ nJ/bit/m^2. In each case, 50 simulation trials are executed. Each source to take the task described in the interest sends a data packet toward the sink only once by using the data path tree. In Figs. 8, 9 and 11, we will plot out the average value of them.

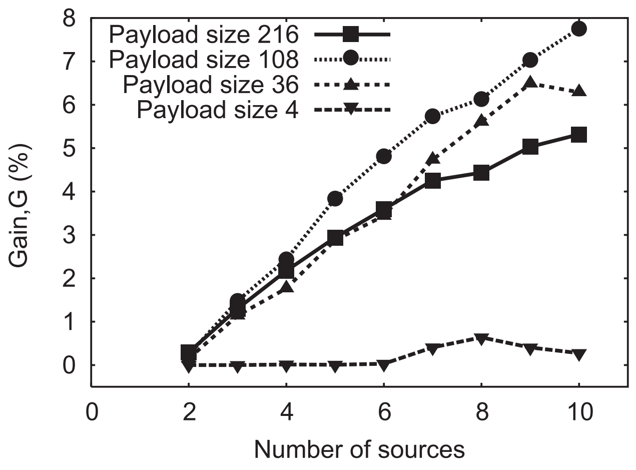

Figure 8. Gains by nodes on the path tree in the case of the traditional exploratory scheme.

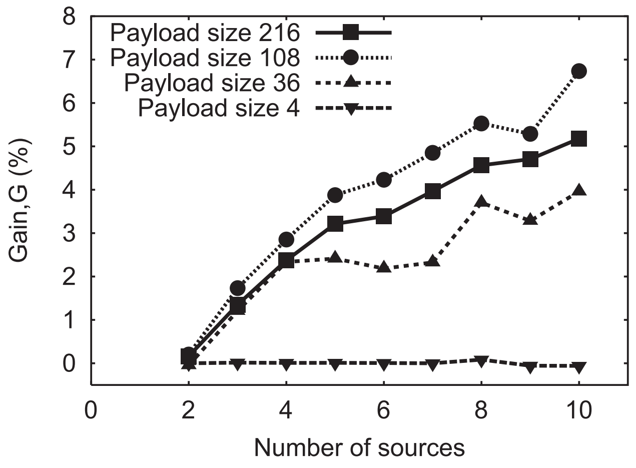

Figure 9. Gains by nodes on the path tree in the case of the hop exploratory scheme.

5.2 Simulation Results

Figures 8 and 9 show the characteristics of gain defined in Section 3 as a function of the number of sources for different payload lengths.

Figure 8 shows the results of the traditional exploratory scheme. In this case, the exploratory messages are distributed network-wide. In the case of the small payload, 4 bytes, the gain is quite low because the aggregation ratio of the linear aggregation is almost identical that of the perfect aggregation. However, in the case of the medium payload, 36 bytes, which is the same as the header, the gain increases concomitant with the number of sources. This tendency is more remarkable in the case of the large payload, 108 bytes. However, in the case of too large payload, 216 bytes, the path tree will become similar to that of the opportunistic routing. Therefore, the gains decrease. These results coincide with predictions by our analysis shown in Section 3.

Figure 9 shows results of the hop exploratory scheme. From this figure, we can see that the AGIT routing is still more efficient than GIT routing and opportunistic routing, but the values of gain are decreased in comparison to those of the traditional exploratory scheme because the spread area of the exploratory messages is smaller than the traditional exploratory scheme.

The gain values are smaller than the expected values obtained from analysis. For analysis, we assume a dense network. However, in the simulation, the nodes are deployed in a discrete fashion. For that reason, the range of choices for the efficient aggregation point in the simulation is smaller than that for the analysis.

Figures 8 and 9 show the gain in the data transmission phase. From this viewpoint, the traditional exploration is preferable. However, it includes the most overhead to construct the path tree. For that reason, we investigate the amount of the overhead. Figure 10 shows the total energy consumption of the entire network between the issue of the interest and the completion of the receptions of one data packet from every source. This figure indicates that the traditional exploratory scheme has more overhead than the others. A trade-off exists between the gain of the data transmission phase and the overhead of the path tree construction phase. The answer to the problem depends on the applications: more precisely, it depends on how long the data transmission phase lasts.

6 Conclusions

This paper presented the aggregation efficiency-aware GIT (AGIT), and also described analyses incorporating the suppression scheme for exploratory messages: hop exploratory.

The AGIT routing can construct a more efficient path tree than the original GIT routing and the opportunistic routing. The improvement becomes more remarkable as the payload packet length becomes larger and/or more sources exist. Our simulation results demonstrate that the AGIT routing achieves about 8% of the gain for the energy consumption of the data transmission compared to the original GIT routing. Our simulation results also emphasize that the suppression scheme, hop exploratory, reduces energy consumption up to 40%.

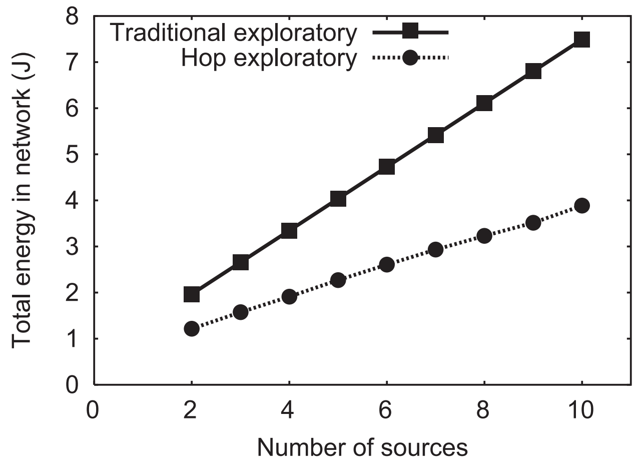

Figure 10. Total energy consumption in whole network until the reception of the data packets from all sources.

Acknowledgment

This research work was partially supported by Strategic Information and Communications R&D Promotion Programme (SCOPE), Ministry of Internal Affairs and Communications, Japan, by the Ministry of Education, Culture, Sports, Science and Technology (MEXT), Grant-in-Aid for Scientific Research (C), No. 18500052, by the Ministry of Education, for Grant-in-Aid for Young Scientists (B), No. 16700066 and No. 16760271, and by the Kayamori Foundation for Advancement of Information Science.

References

[1] T. F. Abdelzaher, T. He, and J. A. Stankovic, "Feedback Control of Data Aggregation in Sensor Networks," *IEEE Conference on Decision and Control*, Dec. 2004.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks (Elsevier) Journal*, vol.38, pp.393–422, March 2002.

[3] K. Akkaya, M. Younis, and M. Youssef, "Efficient Aggregation of Delay-Constrained Data in Wireless Sensor Networks," *Proc. of Internet Compatible QoS in Ad Hoc Wireless Networks 2005*, Jan. 2005.

[4] M. Bhardwaj, T. Garnett, A. P. Chandrakasan, "Upper Bounds on the Lifetime of Sensor Networks," *Proc. of ICC*, pp.785–790, June 2001.

[5] M. Enachescu, A. Goel, R. Govindan, and R. Motwani. "Scale Free Aggregation in Sensor Networks," *Proc. of the First International Workshop on Algorithmic Aspects of Wireless Sensor Networks*, pp.71–84, July 2004.

[6] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, "Building Efficient Wireless Sensor Networks with Low-level Naming," *Proc. of the ACM Symposium on Operating Systems Principles*, Oct. 2001.

[7] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed Diffusion for Wireless Sensor Networking," *IEEE/ACM Transactions on Networking*, Feb. 2003.

[8] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of Density on Data Aggregation in Wireless Sensor Networks," *Proc. of the 22nd International Conference on Distributed Computing Systems*, Nov. 2001.

[9] M. Khan, G. Pandurangan and B. Bhargava, "Energy-Efficient Routing Schemes for Wireless Sensor Networks," *Tech. Rep. of Department of Computer Science, Purdue University*, CSD TR 03-013, July 2003.

[10] B. Krishnamachari, D. Estrinf, and S. Wicker, "Modelling Data-Centric Routing in Wireless Sensor Networks," *IEEE INFOCOM*, June 2002.

[11] B. Krishnamachari, D. Estrin, and S. Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks," *Proc. of the 22nd International Conference on Distributed Computing Systems*, July 2002.

[12] D. Petrovic, C. Shah, K. Ramchandran, and J. Rabaey, "Data Funneling: Routing with Aggregation and Compression for Wireless Sensor Networks," *IEEE Sensor Network Protocols Applications, Anchorage*, May 2003.

[13] A. Wang, W. B. Heinzelman, A. Sinha, and A. P. Chandrakasan, "Energy-Scalable for Battery-Operated MicroSensor Networks," *Kluwer Journal of VLSI Signal Processing*, vol.29, pp.223–237, Nov. 2001.

[14] J. Zhao, R. Govindan, and D. Estrin, "Computing Aggregates for Monitoring Wireless Sensor Networks," *Proc. of IEEE International Workshop on Sensor Network Protocols and Applications*, May 2003.