Master's Thesis

Title

Robustness of Receiver-driven Multi-Hop Wireless Network with Soft-State Connectivity Management

Supervisor

Professor Masayuki Murata

Author

Daichi Kominami

February 15th, 2010

Department of Information Networking

Graduate School of Information Science and Technology

Osaka University

Master's Thesis

Robustness of Receiver-driven Multi-Hop Wireless Network with Soft-State Connectivity Management

Daichi Kominami

Abstract

Energy saving and ensuring robust data collection are the big subjects in realization of wireless sensor networks. In the intermittent receiver-driven data transmission (IRDT) protocol, which aims to save energy and get high reliability, communication between nodes commences when multiple receiver nodes transmit their own IDs intermittently and a sender node receives them. Then we focused on the analogy between this periodic transmission of an ID and a periodic message in the soft-state management which is broadly used for the maintenance of state. Soft state is often considered to have robustness against failures, therefore, we introduce it to IRDT for constructing the robust network.

In this thesis, we propose a soft-state management of routing tables in IRDT, where each node uses the periodic transmission of an ID not only for communication but also update of the routing table. By computer simulation, we show that IRDT can achieve 21% improvement in the robustness against node failure by the soft-state management. Moreover, we show that the receiver-driven asynchronous intermittent transmission protocol suits for the soft-state management by comparing with the sender-driven asynchronous intermittent transmission protocol.

Keywords

Sensor Network
Intermittent Operation
Soft State
Robustness

Contents

List of Figures

List of Tables

1 Introduction

Recently, due to advances in wireless and micro-electromechanical (MEMS) technologies, ad hoc networks have received considerable attention. Among ad hoc networks, sensor networks are expected to be useful in a wide range of applications as they have sensing ability without infrastructure. However, wireless sensor networks have critical technical problems that remain to be solved, one of which is saving energy in sensor nodes with limited battery life. Various approaches for saving energy have been proposed, for example, miniaturizing sensor nodes, media access control (MAC) with sleep control, and multi-hop routing [1-4].

In particular, considerable energy can be saved through intermittent operation, in which wireless nodes sleep to save power and wake up periodically to communicate with other nodes. Then we call this wake-up interval 'intermittent interval'. This power-saving operation is based on the fact that sleeping nodes consume significantly less energy than idling nodes [5]. In intermittent operation, nodes must control wake-up times in order to communicate with each other. There are two types of control method for intermittent operation; synchronous [3, 6, 7] and asynchronous [8-11]. For saving energy and scalability, the latter is superior in terms of the overhead for synchronization control with other nodes [12]. *Intermittent receiver-driven data transmission* (IRDT) protocol, which aims to save energy and to get high reliability, makes use of the intermittent receiver-driven asynchronous media access control (MAC). This protocol is devised for an actual product under development and is proposed as a standard protocol of smart meter systems. By implementing IRDT, we are developing a meter reading system which can be operated with a battery for a long period of time. Our previous research [13] clarified the performance of IRDT by comparing *low power listening* (LPL) protocol [9], which is a sender-driven asynchronous intermittent MAC protocol.

IRDT provides from physical layer to network layer as shown in Fig. 1. In physical layer, IRDT uses GFSK modulation to obtain tolerance for noise and manages a sleep controller which switches wireless device on and off for saving energy. Data link layer controls the link management between two nodes, where each node can establish the link with multiple nodes, that is IRDT can construct a mesh network. In asynchronous MAC, a sender node has to wait until a receiver awakes. IRDT can reduce the time of sender's waiting for a receiver because it can wait for multiple receiver nodes, which reduces energy consumption. Specifically, in IRDT, each receiver sends

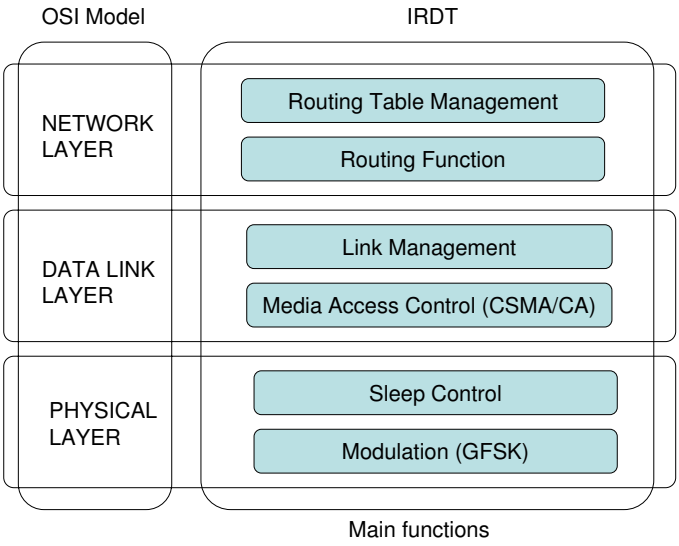


Figure 1: Protocol layer and main function of IRDT

its own ID to inform other nodes that they are ready to receive data packets as shown in Fig. 2. A sender node waits for the receivers' IDs, and when it acquires an ID from an appropriate receiver, it establishes a link with the receiver and sends a data packet. After getting an acknowledge packet for the SREQ (RACK), the sender transmits a data packet and finishes communication following receipt of an acknowledge packet for the data (DACK). In this way, a sender node can communicate with one or more receivers flexibly, which can improve the communication reliability by constructing mesh network and can save considerable energy by shortening the sender's active time waiting for an receiver. Therefore, in network layer, routing protocol is designed to use multiple receiver nodes flexibly and effectively. However, the management of the routing table in IRDT is not designed to deal with the emergency. Thus, when route-changes caused by the wireless channel fluctuations or node failures occur, a system based on the original IRDT may not work properly long after the route-changes. There are a number of applications which collect various data such as environmental information. In these application, it is important to maintain the function of the system in general.

The other critical problem than energy saving in wireless sensor networks is route fluctuation caused by instable radio condition and failure or energy depletion of nodes. Once significant route-changes occur, data sent from sensor nodes can't be collected correctly and the whole system of performance eventually degrades. In particular, in the situation where correct information

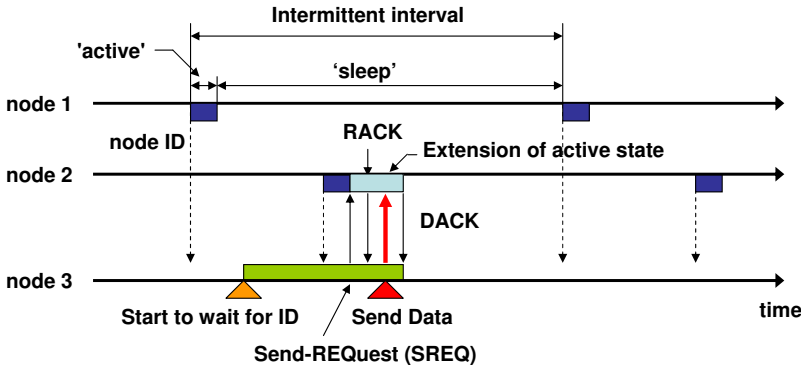


Figure 2: Link management in IRDT

of the route is absolutely necessary such as the failure of the destination node, the quick response to the route-changes is a critical problem. As mentioned above, IRDT cannot deal with urgent route-changes because of its network layer. In this thesis, we note similarities between periodic transmission of an ID in IRDT and soft-state management, therefore in order to improve the robustness of IRDT, we introduce the soft-state management of routing information to IRDT. Here we define the 'robustness' as the property that can maintain a packet collection ratio of a sensor network system even though critical route changes occur. Soft state is one of the method for management of node's state, in which a node sends periodic refresh message to other node to maintain other's state. Other node maintains its own state as long as such refresh messages arrive, but when other node can't receive the refresh message within a given time period, it changes to default state. This soft-state mechanism is generally noted to have robustness [14]. Then we evaluate the improvement of the robustness of IRDT with soft-state management through a comparison with IRDT with hard state by computer simulation. We also compare with sender-driven asynchronous intermittent MAC with soft-state management and show receiver-driven protocol have compatibility to soft-state management.

The rest of this thesis is organized as follows. Next Section 2, we show some related work and in Section 3, we describe the overview of IRDT. In Section 4, we discuss soft-state management in IRDT. We present the simulation results in Section 5 and our conclusions in Section 6.

2 Related Work

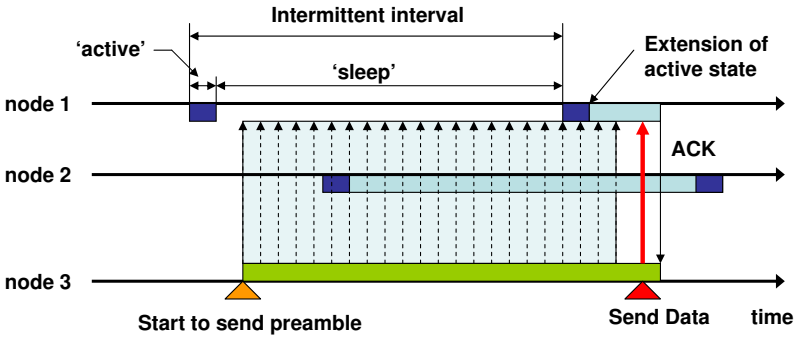
2.1 MAC protocols for intermittent asynchronous transmission

In this section, we present some MAC protocols for intermittent asynchronous transmission and show the essential difference between 'sender-driven' and 'receiver-driven'.

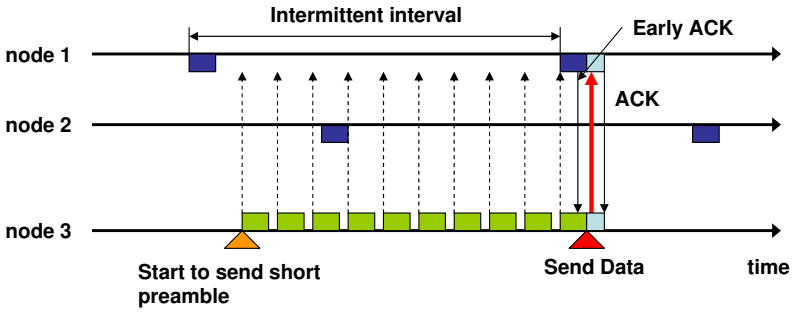
There are various approaches for media access control for intermittent asynchronous transmission. **B-MAC** [4] is a basis of LPL protocols and Fig. 3(a) shows its basic operation. In LPL, receivable nodes intermittently check the channel condition. If the channel is idle, they sleep again, and if busy, they start to be ready to wait for data receptions. After receiving data packets intended for them, they return acknowledge packets. In Fig. 3(a), where node 3 wants to send data to node 1, therefore, in order to make the channel busy, node 3 continuously sends a preamble packet for a longer time than the intermittent interval. After sending the preamble packet, node 3 sends a data packet and wait an acknowledge packet. There are many problems in this LPL protocol, i.e., when the intermittent interval is comparatively long to lower the duty cycle, each sender node occupies the channel by transmitting a preamble packet for a longer time than the interval, which obstructs neighbor nodes' transmission of packets. Moreover, neighbor nodes which are not a receiver for the sender waste energy due to unrelated sender's preamble, which is called overhearing problem. Other problem is that each sender node has only a specific node with which communication is possible.

X-MAC [8] was designed to solve overhearing problem of B-MAC. To prevent the channel occupation of the sender's preamble packet in B-MAC, X-MAC alternatively transmits a short preamble packet continuously which includes receiver node's ID. The operation of X-MAC is shown in Fig. 3(b). Receiver node replies the early acknowledge (early ACK) packet if the ID included in the short preamble is oneself. The sender node transmits a data packet after this early ACK is received, and waits for the acknowledge packet for the data. Receivers that detect unrelated short preamble can sleep soon after this reception finished. Thus, the overhearing problem generated by keeping transmitting the preamble during one intermittent interval in B-MAC can be solved.

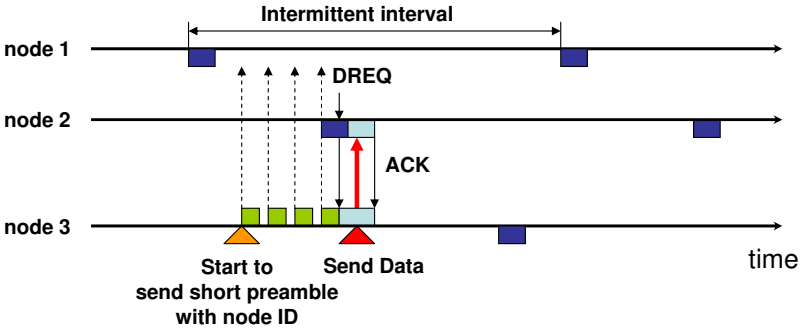
Attribute-based X-MAC (AX-MAC) protocol was proposed in Ref. [15] where sender nodes can use multiple receiver nodes. AX-MAC is one of the LPL, where receiver nodes periodically wake-up and check whether the short preamble packets for themselves are transmitted as shown



(a) Link management in B-MAC



(b) Link management in X-MAC



(c) Link management in AX-MAC

Figure 3: Sender-driven asynchronous intermittent transmission protocol

in Fig. 3(c). When a receiver gets a short preamble packet from an appropriate sender, it returns a data request (DREQ) packet. A node which has a data packet to send first starts to transmit short preamble packets continuously and wait for a DREQ packet from a receiver. After sender node gets a DREQ, it sends a data packet waits for an acknowledge packet for the data. Short preambles are intermittently transmitted, and node ID is included in this preamble and a receiver node that gets this preamble decides whether to relay sender's data packet. Comparing Fig. 3(c) with Fig. 2, these procedures of the link management are fairly similar to each other, but the first packet to start to communicate is sent by sender node in AX-MAC and sent by receiver node in IRDT. An essential difference between AX-MAC and IRDT is that nodes in intermittent operation transmit packets or listen the channel, and this can be said that an essential difference between the sender-driven method and the receiver-driven method. Our previous research showed the impact of this difference on the performance. In this thesis, we compare the IRDT with AX-MAC when these protocols use the soft-state management. Through that comparison, we evaluate the difference when the sender-driven method and the receiver-driven method respectively use the soft-state management.

Ref. [11] proposed two generic asynchronous intermittent MAC, **Transmitter Initiated CyclEd Receiver** (TICER) and **Receiver Initiated CyclEd Receiver** (RICER), and evaluated their latency and power consumption. The procedure of data sending and receiving in RICER takes the similar procedure of IRDT. However, two channels are used in RICER to communicate and moreover, a sender uses just one receiver in RICER. IRDT uses a single channel, which makes implementation more easily, and construct mesh network for a highly reliable system.

2.2 Soft-state protocols and their robustness analysis

In this section, we show some soft-state protocols and describe soft state and hard state.

RSVP [16] is a protocol for the QoS guarantee and the network resource is reserved by receiver. Receivers send Resv messages to their senders periodically for the reservation of the network resource and when the Resv message doesn't reach during the fixed time, sender's state is initialized into default state.

SIP [17] uses soft state for the session control. Each node periodically sends the location information to the location management server for the establishment of sessions. When the location information is not registered during the fixed time, it becomes invalid.

PIM architecture is proposed in Ref. [18], which constructs distributed multicast tree efficiently in low-dense network. PIM needs robustness against severe change of route information, therefore it employs soft-state management. In PIM, each router sends refresh message to its parent router periodically.

SAP [19] is used to make multicast of session information. In SAP, client nodes periodically transmit announcement packet to well-known address and port to maintain the session state.

As above stated, soft state is used in diverse ways, but they don't show the quantitative robustness but the qualitative robustness. John C.S. Lui et al. stressed the need for the quantitative robustness in Ref. [14]. They modeled a hard state protocol and a soft-state protocol and compared them quantitatively. They concluded that when network conditions can anticipate, hard state can attain better performance, but when these conditions are unpredictable, soft-state protocol can suppress drastic increases of the communication cost and the blocking rate in case the DoS attack and the link trouble occur.

The vagueness of the concept of soft state is also recommended in [20]. Authors proposed a formal model for soft-state communication based on a probabilistic delivery model and evaluated the tradeoff between performance overhead and robustness.

The distributed data management server application is assumed for the evaluation of the robustness of soft-state management in Ref. [21]. The message processing rates of the soft-state data registering and the hard state data registering is evaluated, and the robustness of soft-state management is shown. The consistency was defined as $E[c(t)]$ that meant how much equal after information that two systems have respectively after infinite time passed.

In the following sections, we show the overview of IRDT and describe the way that IRDT maintains its routing information with soft state and hard state.

3 Overview of IRDT

As discussed previously, IRDT is developed for highly reliable and low-power-consumption system. Note that IRDT is developed and actually used for meter products [10]. Furthermore, we are proposing this technique to IEEE 802.15 Task Group 4 as a part of standard protocol for smart meter system [22]. In this section, we present the overview of IRDT especially on its MAC layer and routing layer.

3.1 MAC layer

In terms of MAC layer of IRDT, its procedure of connection is explained in Section 1 and carrier sensing multiple access with collision avoidance (CSMA/CA) is used in sending any packets. Here, the decision of a sender regarding whether or not to send an SREQ packet is made on the basis of a routing table and a routing function provided from routing layer. Figure 2 shows the example of communication among three nodes. In this figure, node 3 gets an ID packet from node 2 and decides to send an SREQ packet to node 2 according to its own routing function.

3.2 Routing layer

The routing layer in IRDT offers the following two functions:

- Management of the routing table
- Management of the routing function

The routing table is used in the routing function and the routing function decides transmission of an SREQ packet. The routing table contains the values which figure that the hop count from node's own destination is larger, smaller, equal, or unclear as compared with the hop counts from the destination to every node respectively. Therefore, in order to make the routing table, each node has to know the hop counts from all nodes in the network. In IRDT, this information is registered in the hop count table. To begin with, we describe the hop count table of each node. In following description, we use simple network model as shown in Fig. 4 and a node that has an ID k is denoted by node k and the hop count from node m to node n is denoted by H_{mn}.

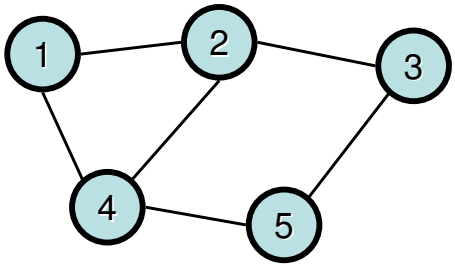


Figure 4: Simple network model for explaining routing layer

Original owner : node 2	
Node ID	Hop
1	1
2	0
3	1
4	1
5	2

(a) Own hop count table

Original owner : node 1	
Node ID	Hop
1	0
2	1
3	2
4	1
5	2

Original owner : node 3	
Node ID	Hop
1	2
2	1
3	0
4	2
5	1

Original owner : node 4	
Node ID	Hop
1	1
2	1
3	2
4	0
5	1

(b) Hop count tables received from neighbor nodes

Figure 5: Hop count tables of node 2 in Fig. 4

Hop count table

Each node has a hop count table in which the numbers of minimum hops from all nodes are registered as shown in Fig. 5(a). In default IRDT, all nodes wake up and wait for ID packets for six seconds every one hour. When a node n receives an ID packet from node m, node n registers on its hop count table that H_{mn} is one, that is to say, node n registers node m as a neighbor node. Each node also has its neighbor nodes' hop count tables in order to calculate the minimum hop counts of nodes whose distance is more than two hops from itself. For example, Fig. 5(b) shows the hop count tables of node 2 of Fig. 4.

Here, The exchanges of the hop count tables between neighbor nodes happen in the following two cases.

(1) Table exchange occurs when the hop count table is modified after sampling an node ID

finishes.

(2) Table exchange occurs when the hop count table is modified after exchanging tables with a neighbor node.

In the above case (2), it seems to occur the flooding of the table exchanges, therefore nodes exchange tables after the elapse of a random period of time.

Routing table

As stated previously, the routing table has the values that indicate the difference of the hop counts between the owner node of the routing table and an arbitrary node. Now, we explain how each node creates the routing table.

A routing table (denoted R) is represented by the $N \times N$ matrix where N means the number of nodes in the network as shown in Fig. 6. Here, we define that the element of R is r_{ij}, denoting the element of row i and column j. In addition, i is the ID of the destination node and j is the ID of the receiver node. Here, r_{ij} is the integer value which means that the relay to the destination whose ID is i by way of the receiver whose ID is j is any one of 'forward relay', 'sideward relay' or 'backward relay' which are described below. In a routing function, 'forward relay' is used for the minimum hop routing. 'Sideward relay' and 'backward relay' is, for instance, used for avoiding the congestion.

Given sender node n whose destination is node i, when node n receives an ID from node j, node n compares H_{ni} in its hop count table with H_{ji} in the hop count table previously received from node j. If the Boolean expression, $H_{ni} - H_{ji} = 1$, evaluates to true, the relay to node i by way of node j is called 'forward relay' and node n sets r_{ij} to one. If $H_{ni} - H_{ji} = 0$ is true, the relay is called 'sideward relay' and r_{ij} of node n is two and if $H_{ni} - H_{ji} = -1$ is true, this relay is 'backward relay' and r_{ij} is set to three. In case node j is not a neighbor of node n, node n set r_{ij} to zero and the relay by way of node j is denoted by 'non-neighbor relay'. In addition, we define that 'forward node' is a node for the forward relay and 'sideward node' and 'backward node' are in a similar way. The example of the routing table of node 2 in the network composed of five nodes shown in Fig. 4 is illustrated in Fig. 6. The elements of the routing table are calculated based on the hop count tables shown in Fig. 5.

		Destination's ID				
		1	2	3	4	5
Receiver's ID	1	1	0	3	2	3
	2	0	0	0	0	0
	3	3	0	1	3	1
	4	2	0	3	1	1
	5	0	0	0	0	0

Figure 6: Routing table of node 2 in Fig. 4

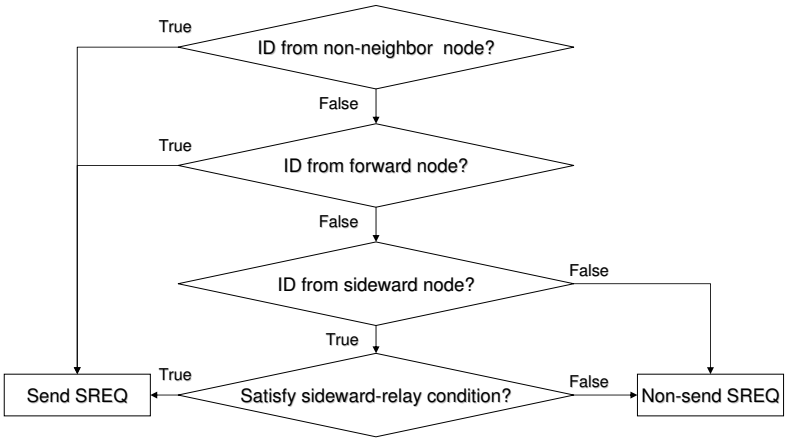


Figure 7: Routing function

Routing function

The routing function made in the routing layer is a logic function, and it is offered from the routing layer to the MAC layer. In MAC layer, the sender decides whether or not to return the SREQ packet according to this function. The above-mentioned routing table is used in this routing function. When a node which has a data packet receives an ID packet, MAC layer only check its own routing function and the destination of the data and the received ID. The example of the routing function is shown in Fig. 7. In this figure, the minimum hop routing is assumed to do, however sideward relay is also used when the condition of sideward relay is satisfied. Easy examples of the sideward-relay condition is that true is returned at the probability of 25%.

Here, we assume that an ID packet from non-neighbor node m is arrived at node n whose

destination is node i. Because node n doesn't have node m's hop count table, node n has no way of telling that node m is nearer to the destination of node n. Therefore, in IRDT, when a node that receives an ID from a non-neighbor node, it always replies to send an SREQ packet. What is important then is that node n transmits an SREQ packet to node m including H_{ni} in the SREQ packet. Thus, after node m received an SREQ from node n, node m can compares H_{mi} and H_{ni}. If the expression $H_{mi} < H_{ni}$ is satisfied, node m returns an EACK packet, but if it does not, node m returns a negative acknowledgement (NACK) packet. If node n receives a NACK packet, node n starts to wait for an ID packet again.

4 Soft-State Connectivity Management in IRDT

In this section, we present two kinds of soft-state management for improving robustness of IRDT network against the wireless radio fluctuations and the node failures that cause the route-changes. We propose a soft-state management of the neighbor relationship in the hop count table and also propose a soft-state management of the hop count tables received from neighbor nodes. The following sections explain of each in detail.

4.1 Soft-state management of the neighbor relationship between neighbor nodes

Each node in IRDT has a hop count table in which the hop counts from all nodes in the network are registered as described in section 3.2. The neighbor relationship in the hop count table can be maintained by sampling an ID packet because ID packets are periodically transmitted. In the original IRDT, each node tries to sample ID packets for a short time period every a few hours. Then we call this period *sampling period* denoted by T_s and call this interval *sampling interval* denoted by T_i. A few hours of T_i is too long to response to the route-changes and this seems to be pretty hard state-like and it can be said that the state becomes softer as T_i decreases.

Here, we propose the soft-state management of the neighbor relationship in the hop count table. First of all, we define the default state of the neighbor relationship as 'disconnected'. In the hop count table, when a node n cannot get the ID packet from node m within T_i, node n set H_{nm} to infinity to represent the 'disconnected' state. If node n can obtain the ID packet from node m, the state of the neighbor relationship with node m is 'connected' and node n sets H_{nm} to one as a neighbor node. The example is shown in Fig. 8 where node 2 in Fig. 4 cannot get the ID packet from node 4 within the last T_i. Node 2 decides the state of the neighbor relationship with node 4 is 'disconnected' and sets the hop count from node 4 to infinity in its hop count table.

For this management of the neighbor relationship, we add a time stamp to each item in the hop count table as shown in Fig. 8, where the items in the column of 'last update' are the time stamps. For maintaining neighbor relationship, each node waits for an ID packet for T_s every T_i. In the sampling period, when a node gets an ID packet, the node sets the hop count from the sender of the ID to one and updates the time stamp to the current time. Note that when a node waiting for an ID packet in order to transmit a data packet receives an ID packet, the node also sets the hop count to one and updates the time stamp. After the sampling, a node recalculates the hop count which

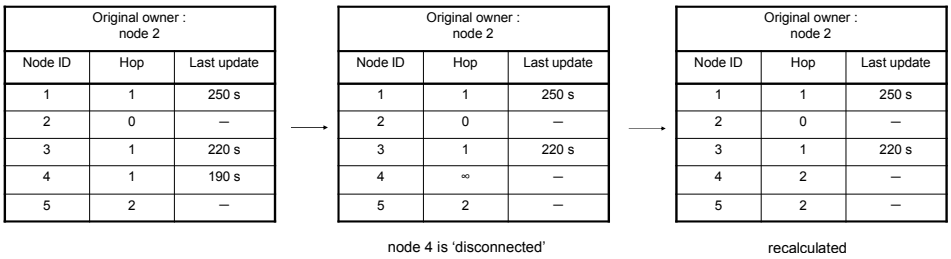


Figure 8: Neighbor relationship management: the last update must be after 200 s, thus the hop count from node 4 is recalculated

corresponds to infinity by using the hop count tables received from the neighbor nodes. As we discuss later, the hop count table received from the 'disconnected' node is discarded. Therefore, in the recalculated hop count table, the hop count from a 'disconnected' node is more than one as shown in Fig. 8.

Here, the operation when a node tries to sample ID packets takes place as follows:

In case a node has no data packet:

When a node has no data packet, the node is in the intermittent operation, namely, it repeats sleep and ID transmissions. In this case, the node samples ID packets instead of the sleep and the node continues the periodical transmission of an ID packets, which becomes possible to maintain the neighbor relationship even when two or more nodes are sampling at the same time.

In case a node has data packet to send:

In this case, a node waits for an ID packet for a data transmission, therefore, when the node receives an ID packet, it sets the hop from the sender of ID to one and update the time stamp to the current time. In addition, when the node receives an ID packet, it returns an SREQ packet if this ID meets with the routing function, which makes it possible to send an data packet even when the node is sampling.

4.2 Soft-state management of neighbor nodes' hop count table

In IRDT, the hop count tables of the neighbor nodes are necessary for each node to make own hop count table and own routing table. In original IRDT, each node exchanges the hop count tables

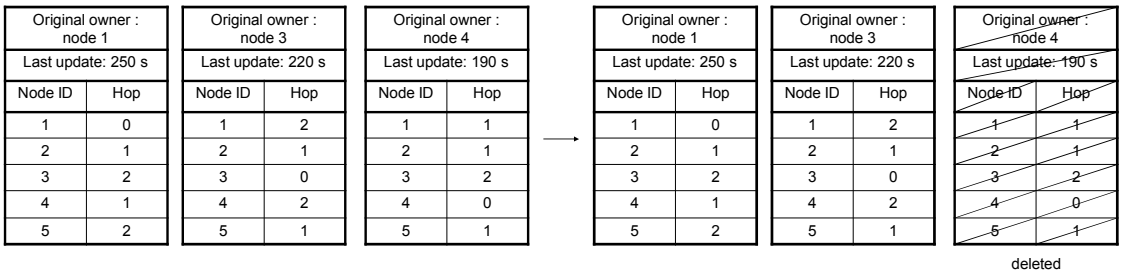


Figure 9: Neighbor hop count table management: the last update must be after 200 s

with the neighbor nodes when own hop count table has changed. Moreover, a node keeps holding an neighbor's table once the node receives a table from its neighbor.

Here, we introduce the soft-state management of the hop count table. We define the default state of the hop count table received from a neighbor node (denoted by neighbor hop count table) as 'unused'. When a node n cannot get the table from node m within T_i, node n deletes the neighbor hop count table from node m. If node n can obtain the hop count table from node m, the state of the neighbor hop count table from node m is 'used'. The example is shown in Fig. 9 where node 2 in Fig. 4 cannot get the hop count table from node 4 within the last T_i. Node 2 decides the state of the neighbor hop count table from node 4 is 'unused' and deletes the table from node 4.

For this management of the neighbor hop count table, we add a time stamp to the neighbor hop count table as well as the neighbor relationship management, as shown in Fig. 9. For maintaining neighbor hop count table, we use the T_s and T_i. Therefore, the management of the neighbor relationship and management of the neighbor hop count table are done at once. This is because if these managements are separately controlled, the recalculating of own hop count table bring unexpected results. For example, in Fig. 8, when node 2 decide the state of the neighbor relationship with node 4 is 'disconnected', if node 2 still has the neighbor hop count from node 4, the recalculation of the hop count from node 4 result in one. After the sampling, a node recalculates own hop count table by using the neighbor hop count tables that the node still has.

Now, we describe the procedure of exchanging the hop count tables. As a premise, hop count tables are exchanged when a node receives an ID packet while the node is sampling. Note that only a time stamp is updated when the node determines that the node doesn't need updating the neighbor hop count table from the ID-sender. Here, the sequence number (denoted by table se-

quence number or TSN) is added to the hop count table and the sequence number is used for the determination whether or not to update the neighbor hop count table. This sequence number is incremented when the hop count table is updated and the latest table sequence number is included in an own ID packet. A node that receives an ID packet compares the sequence number included in the ID packet with the sequence number of the neighbor hop count table previously received from the sender of the ID packet. If the TSN included in the ID packet is larger than the TSN of the neighbor hop count table, the nodes notice that the neighbor hop count table from the ID-sender is old and it has to get the latest neighbor hop count table.

Specifically, tables are exchanged according to the following procedures.

(1) When a node receives an ID packet while it is in sampling period, it checks the TSN contained in the ID packet. The node also examines that the TSN of the neighbor hop count table received from the ID-sender.

   (a) If the node has the latest neighbor hop count table, it updates the time stamp of the neighbor hop count table received from the ID-sender and returns a table non-exchange (TBNX) packet. The node that receives a TBNX packet addressed for the node itself also updates the time stamp of the neighbor hop count table from the TBNX-sender.

   (b) If the node has an old neighbor hop count table, the node and the ID-sender exchange tables each other as follows.

(2) The node transmits a table exchange (TBEX) packet that demands to exchange tables. TBEX packet includes two TSN values. One is the table sequence number of the neighbor hop count table previously received from the ID-sender (denoted by TSN1) and the other is own table sequence number (denoted by TSN2).

(3) When the ID-sender receives an TBEX packet, it compares the TSN1 contained in the TBEX packet with the TSN of own latest table and transmits only the difference as a table packet between the previous own hop count table whose TSN corresponds to TSN1 and the present own hop count table. For this purpose, all nodes maintain a history of own hop count table. Furthermore, the ID-sender checks the TSN1 included in the TBEX packet. If the ID-sender

also has need to acquire the TBEX-sender's latest hop count table, the TSN of the hop count table received from the TBEX-sender previously is added in the table packet.

(4) After receiving the table packet, the node sends a table packet if it is necessary. If there is no need for transmitting a table packet, the node comes to wait for an ID packet again.

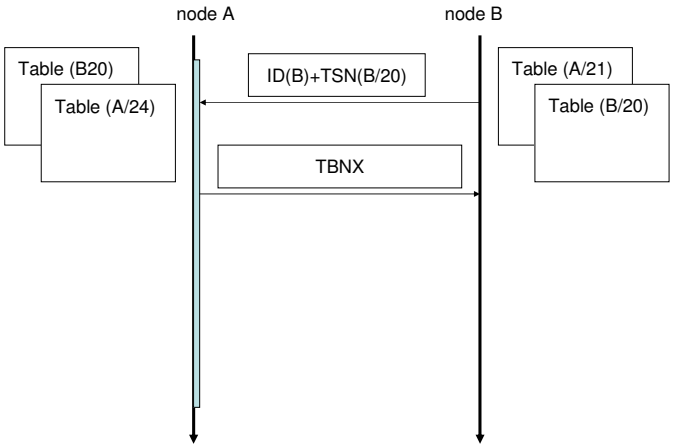
The specific examples are shown in Fig. 10 about the above-mentioned exchange of the hop count tables. In these figures, node A is in sampling period and node B transmits an ID packet.

Fig. 10(a) shows the case only node A has the latest hop count tables created by node A and node B respectively. In this case, node A checks an ID packet from node B and notices that the neighbor hop count table previously received from node B is still latest. Therefore, node A return a TBNX packet.

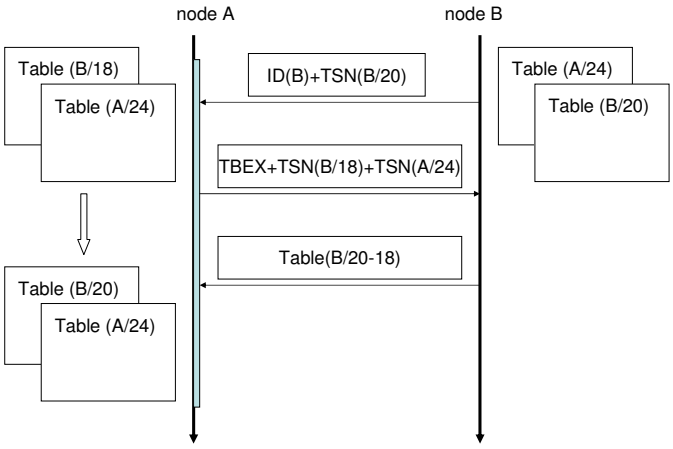
In Fig. 10(b), only node B has the latest tables of node A and node B respectively. So, node A returns a TBEX packet including the TSN of own hop count table and the TSN of the neighbor hop count table from node B. In this figure, the latter TSN is 18 and the TSN of node B's latest hop count table is 20, therefore node B transmits the difference between the hop count table whose TSN is 18 and the latest hop count table (denoted by Table(B/20-18) in Fig. 10(b)).

Fig. 10(c) illustrates the case where neither node A nor node B has one another's latest hop count table. Accordingly, in addition to the procedure as shown in Fig. 10(b), node A also transmits a table packet to node B.

Here, we discuss the TBEX packet collisions. When an ID packet reaches two or more nodes under sampling, TBEX packets or TBNX packets are returned simultaneously from the nodes that received the ID packet. For avoiding this collision, random value is added to every T_i.

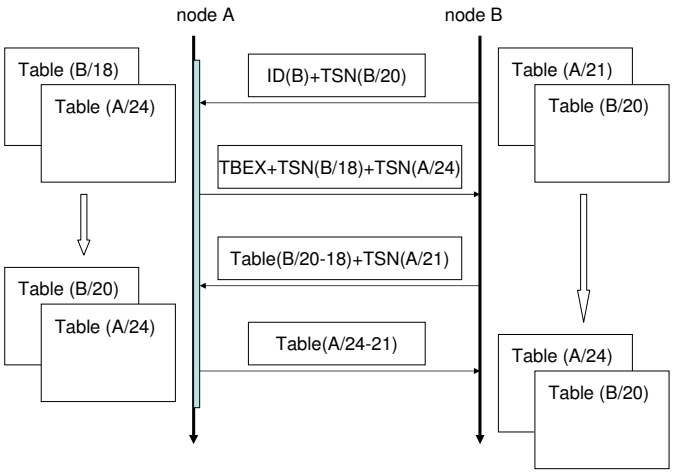


(a) Case: Node A has the latest table of node B



(b) Case: Node B has the latest table of node A

Figure 10: Table exchange sequence



(c) Case: Neither node has each other's latest table

Figure 10: Table exchange sequence

5 Simulation Results

We show the basic performance and robustness of IRDT soft-state managements of the neighbor relationship and the neighbor hop count tables by computer simulation. First, we compare the IRDT and AX-MAC when both protocols use the soft-state management in order to compare the impact on applying the soft-state management to the receiver-driven method and the sender-driven method. We evaluate and compare the packet collection ratio, the mean delay, the average energy consumption, and the overhead of the energy consumption for the soft-state management. In AX-MAC, each node continues to send short preamble packets for T_s every T_i which include own ID number and the TSN of own table. A receiver that catches a short preamble packet returns a TBEX packet and exchanges hop count tables if necessary as is the case with IRDT. Next, we evaluate robustness of IRDT against the wireless channel fluctuations and node failures. Finally, we discuss the compatibility between IRDT and soft-state management and discuss robustness of IRDT.

First, we describe our simulation model. We use the network model which shapes a square with 300 m of side length where 30 sensor nodes represented by circles are randomly deployed and 1 sink node represented by a square is set on the left bottom of the network as shown in Fig. 11. The parameters are set as shown in Table 1 and we assume that data packets are generated by each sensor node according to Poisson process. We assume that the number of histories of own hop count table in each node is sufficiently large. Note that in our reception model, when collisions with other packets occur while a packet is being received, the packets are always discarded. All nodes use CSMA/CA when sending any kind of packet, however, they cannot prevent hidden node problems unfortunately.

5.1 Performance evaluation

Here, the wireless channel fluctuations and node failures are not considered in order to focus on the basic performance of IRDT and AX-MAC with the soft-state management. The packet collection ratio is calculated by dividing the number of the packets received at the sink node by the number of all generated packets. The mean delay is the mean time elapsing from a generation of a data packet to an arrival of the data packet at the sink node. The average energy consumption is obtained to divide the sum of the energy consumption of all nodes by the number of nodes and the

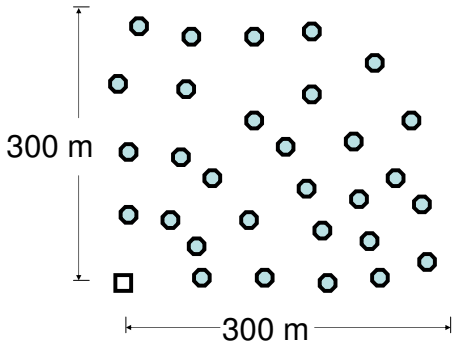


Figure 11: Network model for evaluating the basic performance

Table 1: Parameter settings

Parameter	Value
Transmission speed	100 kbps
Communication range	100 m
Data packet generation rate	0.003 packets/s
Current consumption (TX)	20 mA
Current consumption (RX)	25 mA
Current consumption (SLEEP)	0 mA
Packet size (ID, SREQ, DREQ, TBEX, TBNX)	24 byte
Packet size (RACK, NACK, DACK)	22 byte
Packet size (DATA)	128 byte

overhead of energy consumption for soft-state management is the proportion of the sum of energy consumed for ID-samplings and table exchanges to whole energy of nodes. In the simulation, each node communicates with nodes within 100 m and the intermittent interval is set to 0.1 s or 1.0 s. The simulation commences after initializing phase in which each node exchanges hop count table with neighbor nodes sufficiently and finishes at 8000 second in the simulator. The sideward-relay conditions of IRDT and AX-MAC is as follows:

The sideward-relay condition of IRDT

A sender node uses sideward-relay condition after it tried all of the forward relays since the

latest data was generated by the sender. The sender returns an SREQ packet for an ID packet from a sideward node at the probability of 25%. This condition is based on the minimum hop routing but nodes can also use detour paths flexibly.

The sideward-relay condition of AX-MAC

A receiver node transmits a DREQ packet at the probability of 25% when the receiver receives a short preamble packet from a sideward node.

5.1.1 The effect of the sampling-interval changes

In general, comparing the sampling interval of the soft-state management and that of the hard-state management, the soft-state management uses the shorter interval in order to respond flexibly to changes in the network, namely in order to acquire robustness. However, it seems that the shorter interval leads larger overhead of the energy consumption. We identify the shorter interval as 'soft state' and the longer one as 'hard state'. Then, we change the sampling interval in our simulation and evaluate the performance. Note that the sampling period is fixed to two times of the intermittent interval.

Figure 12(a) shows the packet collection ratio when the sampling interval is changed from 30 s to 600 s every 30 seconds, where the error bar corresponds to the 95% confidence interval. IRDT and AX-MAC can attain a higher collection ratio at 1.0 s and 0.1 s of the intermittent interval respectively. This difference is accounted for the difference between the sender-driven method and the receiver-driven method. In the asynchronous intermittent MAC protocol, receiver nodes repeat the intermittent operation because the receivers don't always know whether their neighbor nodes have a data packet.

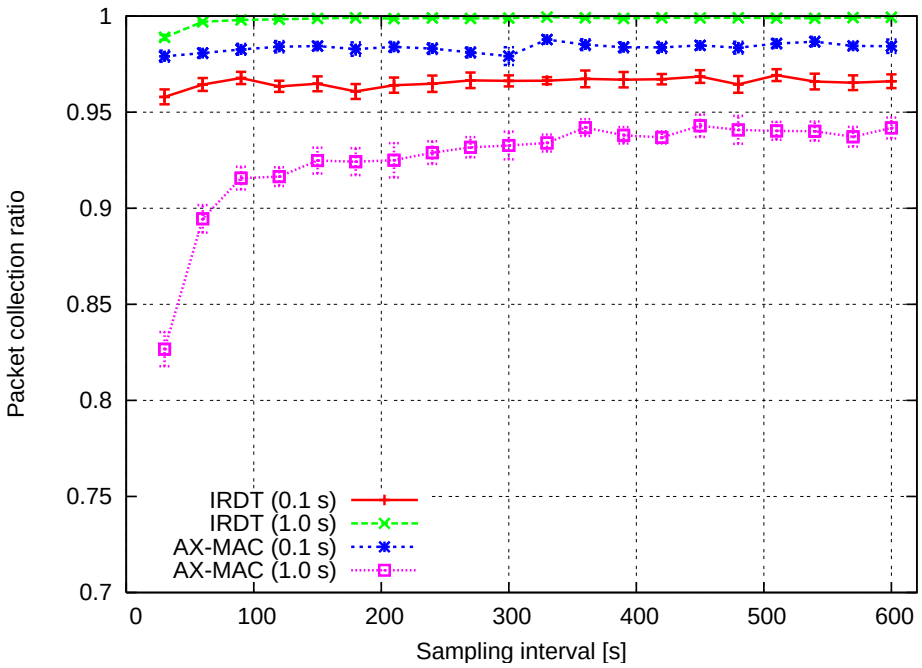
Then in a receiver-driven method, receiver nodes have to transmit a packet periodically in order to inform that they are ready to receive a data packet, which obstructs much more the transmissions of their neighbor nodes as the intermittent interval is shorter. As for the longer intermittent interval in IRDT, the interference by the packets scarcely happens and a higher data collection ratio can be attained. For the application where the packet generation rate is high, because the maximum number of packets that each node can receive per unit time corresponds to the number of ID packets each node sends per unit time, nodes have to set their intermittent interval to shorter value. We target the sensor networks where data packets are less generated, so above mentioned case is

out of scope in this thesis. Meanwhile, in a sender-driven method, receivers wake up and check the channel condition in order to obtain a sender's packet which informs them to be ready to send a data packet. By contrast to the receiver-driven method, a short intermittent interval is more preferable to get a better data packet collection ratio. However, a long intermittent interval causes a long preamble transmission to announce the sender of the preamble has a data to send, which obstructs the transmissions of their neighbor nodes. Moreover, receivers cannot do anything when the preamble transmission from two or more their neighbor nodes occur simultaneously, which is the critical problem at the sink node. In the worst case, neighbor nodes of the sink node go on sending preambles until they use the sideward relay or their sending timer, if they have, is expired.

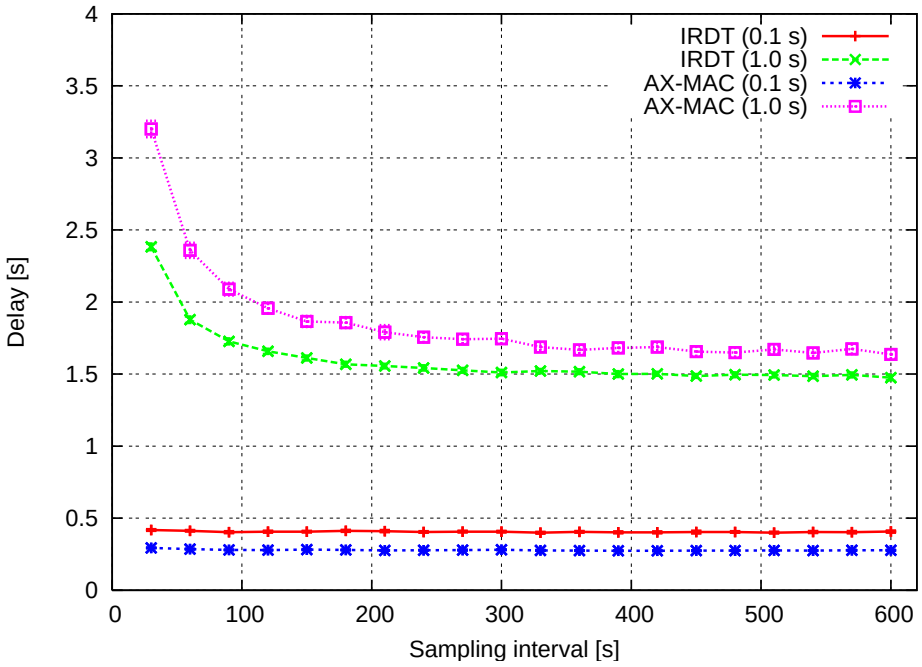
In regard of the traffic overhead for soft-state management, AX-MAC with 1.0 s of the intermittent interval degrades its packet collection ratio notably when the sampling interval is shorter than 100 s. Because even the sink node transmits short preambles every T_i in AX-MAC, the sink node can't receive a data packet all that time. Especially, IRDT with 1.0 s of the intermittent interval can always get the highest packet collection ratio, more than 98% irrespective of the sampling interval.

The mean delay is shown in Fig. 12(b) which is the mean time since data packets are generated until the packets arrived at the sink node. When the intermittent interval is 1.0 s, IRDT can deliver a data packet average 10% faster than AX-MAC, but when 0.1 s of the intermittent interval, the mean delay of AX-MAC is 30% reduced by that of IRDT. The main reason of the delay in AX-MAC is that AX-MAC uses sideward relays more frequently than IRDT due to its sideward-relay condition. Because senders in IRDT make a decision of whether to send a data packet in accordance with its routing function, the sideward-relay condition in the routing function can use sender's information, such as forward relays have been already tried. In our simulation, IRDT uses the sideward relay after all forward relays have been tried but fail, which means that the sender cannot get an EACK packet and a DACK packet from all forward nodes. Meanwhile, in AX-MAC, a receiver node judges a transmission of a DREQ packet from its own routing function, therefore in routing function, sender nodes' information cannot be taken in use. As a result, a DREQ packet for the sideward relay is transmitted with a fixed probability. This is how the delay in AX-MAC increases. Meanwhile, when the intermittent interval is 0.1 s, the obstructions caused by ID transmissions make data transmissions more delayed in IRDT than in AX-MAC.

An increase in the power consumption of IRDT and AX-MAC can be seen when the sampling

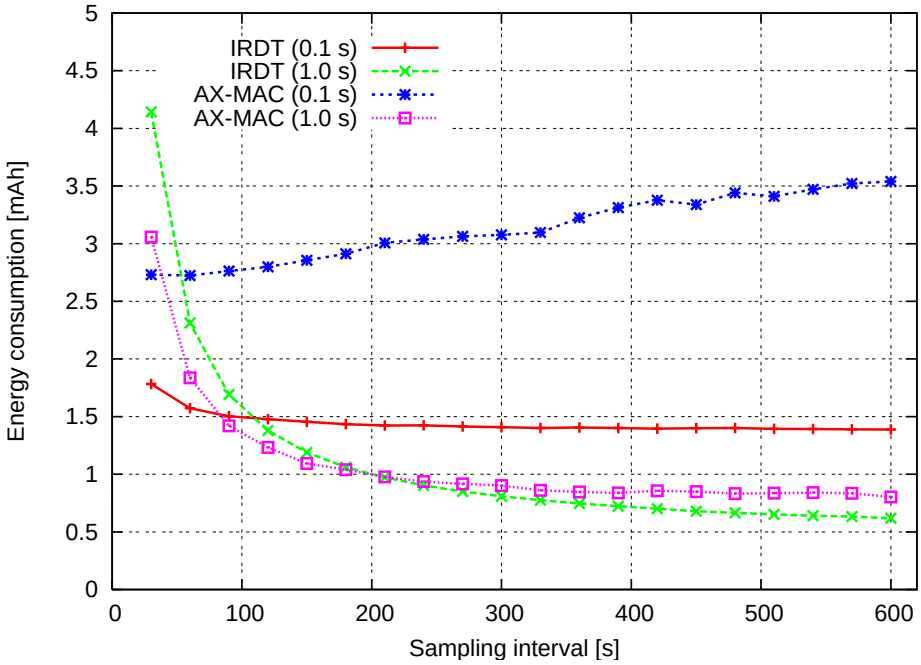


(a) Packet collection ratio

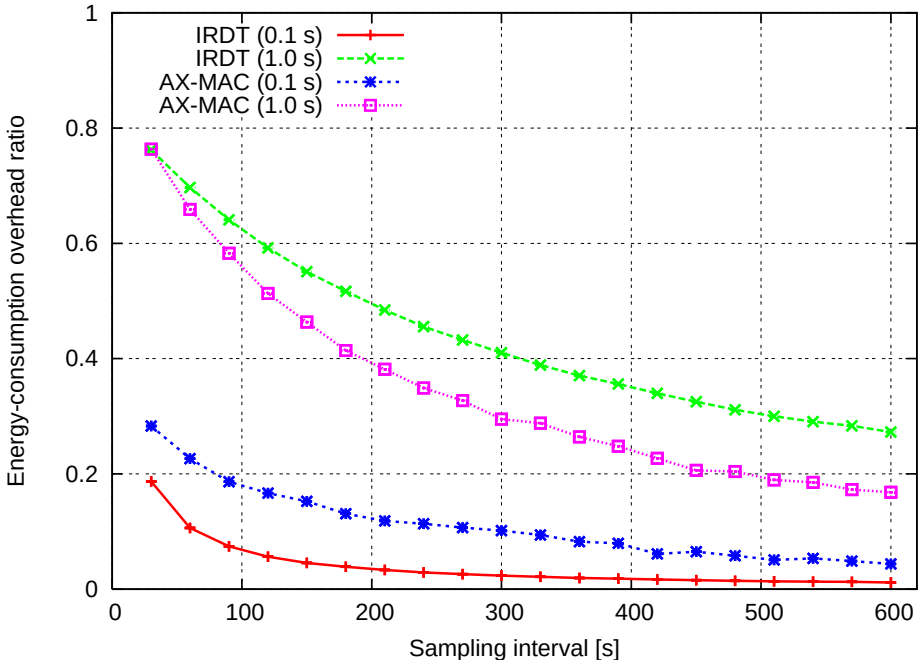


(b) Mean delay

Figure 12: The basic performance of IRDT and AX-MAC: changing the sampling interval



(c) Average energy consumption



(d) Overhead of energy consumption for soft-state management

Figure 12: The basic performance of IRDT and AX-MAC: changing the sampling interval

interval is short as shown in Fig. 12(c). Only the average energy consumption of AX-MAC at 0.1 s of the intermittent interval rises as the sampling interval increases because of the inaccuracy of nodes' routing table. The inaccuracy of the routing table which lengthens the time for transmitting short preamble packets since the number of the forward node decreases, which has also a little effect on the energy consumption of AX-MAC with the 1.0 s intermittent interval. When the intermittent interval is 0.1 s, because the duty cycle of receivers is higher than in the 1.0 s intermittent interval, nodes consumes more energy. Furthermore, the overhearing becomes a very serious problem in AX-MAC then. The average energy consumption of AX-MAC is at least 50% higher at 0.1 s of the intermittent interval. In the case of the comparatively long intermittent interval such as 1.0 s, the energy consumption is more affected not by the overhearing but by the sampling interval T_i. We set the sampling period T_s to two times of the intermittent interval, so, when T_i is 30 s, each node wakes for the sampling for 2.0 s every 30 s at the intermittent interval of 1.0 s. This consumes more energy at both of IRDT and AX-MAC.

Figure 12(d) shows the overhead of the energy consumption for the sampling, the exchanging tables, and the overhearing for the sampling. The overhead in AX-MAC is higher than that of IRDT when the intermittent interval is 0.1 s because of the overhearing. However, little influence of the overhead at the 1.0 s intermittent interval reverses that. At this time, the overhearing by the transmissions of short preambles for data transmissions happens more frequently than short preambles for the sampling.

5.1.2 The effect of the sampling-period changes

Because neither the changes of the wireless channel conditions or the failures of nodes occur in order to evaluating basic performance, each node has a hop count table and registers the nodes within 100 m as a neighbor. If a node cannot receive an ID packet from its neighbor node within T_i, it changes own hop count table. The followings are the situations in which a node cannot receive an ID packet from neighbors normally in IRDT.

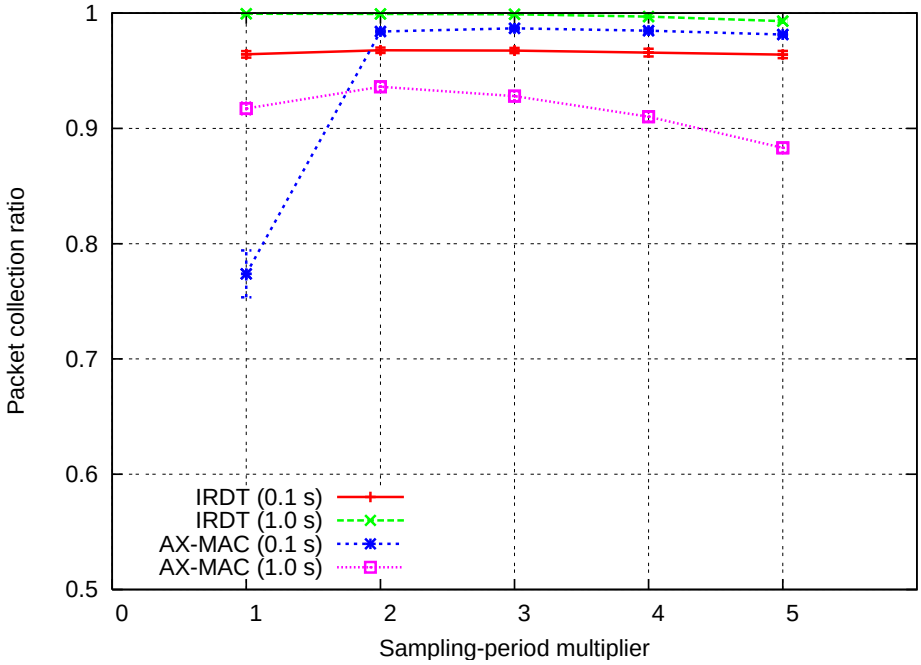
- The case a neighbor node is receiving a packet at the time of the transmission of an ID packet.
- The case a neighbor node senses the channel conditions according to CSMA/CA and passes on the ID transmission because of the channel noise.

- The case a contention occurs when a sampling node receives an ID packet.

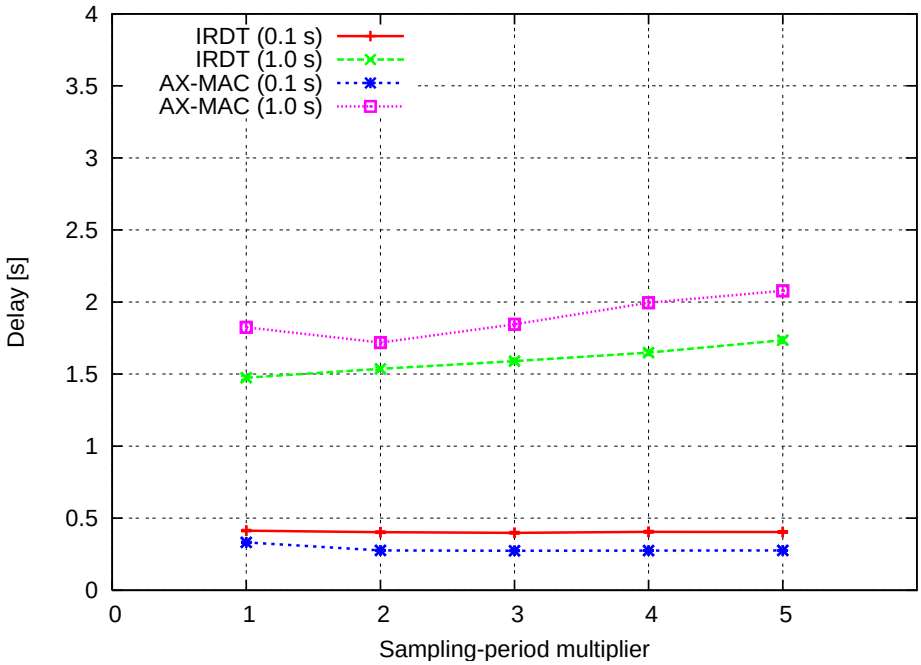
These mistakes of the sampling are solved by using the long sampling period. In the simulation results shown in Fig. 12, we set the sampling period to two times of the intermittent interval. In this case, the packet collection ratio is hardly influenced by the sampling period when the channel fluctuations and node failures are not considered. However, the mean delay and the energy consumption of AX-MAC are affected due to the inaccuracy of the routing table. Here, we show the performance of IRDT and AX-MAC when this sampling period is changed from one time of the intermittent interval to five times of the intermittent interval in Fig. 13, where the sampling interval is set to 300 s. 300 s of the sampling interval is set in order to clarify the impact of only the sampling period. In the following figures, each value of the x-axis (denoted by X) refers to that the sampling period is X times of the intermittent interval.

The packet collection ratio of IRDT is constant at each intermittent interval respectively, but AX-MAC is sensitive a little to the sampling period as shown in Fig. 13(a). The collection ratio at 0.1 s of the intermittent interval in AX-MAC is remarkably wrong, which is less than 80%. This is because, in AX-MAC, the short preamble packets transmitted by the sink node for the sampling are interfered by other nodes' transmission of a packet. In this case, the neighbor nodes of the sink node lose the connectivity to the sink node. Additionally, this neighbor nodes tables gradually expand to the whole network. In contrast, IRDT can maintain the connectivity because each node is waiting for the reception of ID packets when sampling an ID packet, which is not affected by data transmissions. The collection ratio at the 1.0 s of the intermittent interval decreased as the sampling period is twice or more long. The increase of exchanging table packets during the sampling causes the decrease.

As for the mean delay, the inaccuracy of the routing table makes delay longer particularly when the sampling period is 1.0 s as shown in Fig. 13(b). The average energy consumption shown in Fig. 13(c) rises as the sampling period increase except for the AX-MAC with the 0.1 s intermittent interval because the inaccuracy of the routing table increases sideward relays and causes more overhearing. Therefore, the overhead of the energy consumption for the soft-state management in AX-MAC with the 0.1 s sampling period is lower than that with the 0.2 s sampling period as shown in Fig. 13(d).

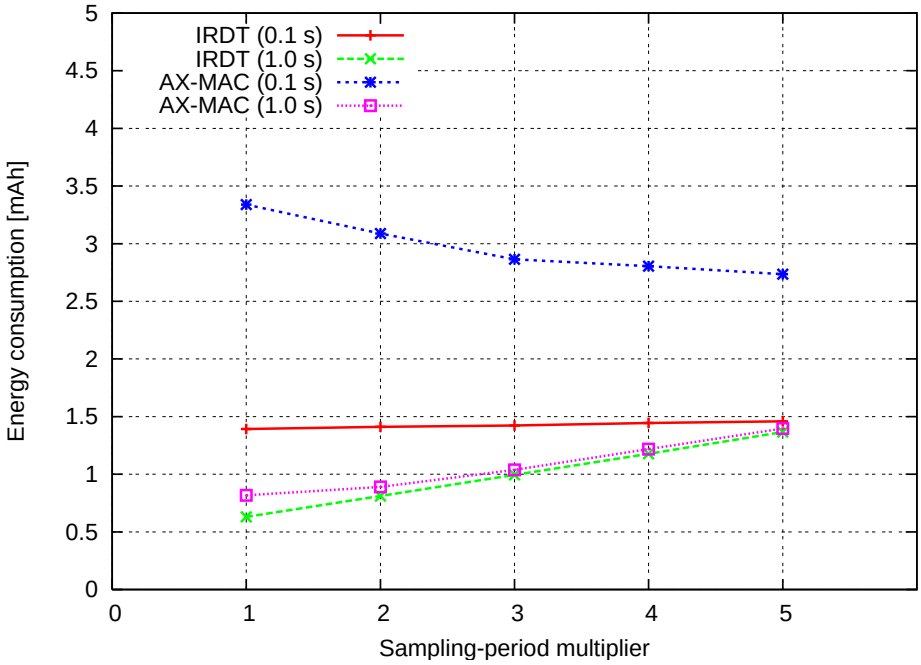


(a) Packet collection ratio

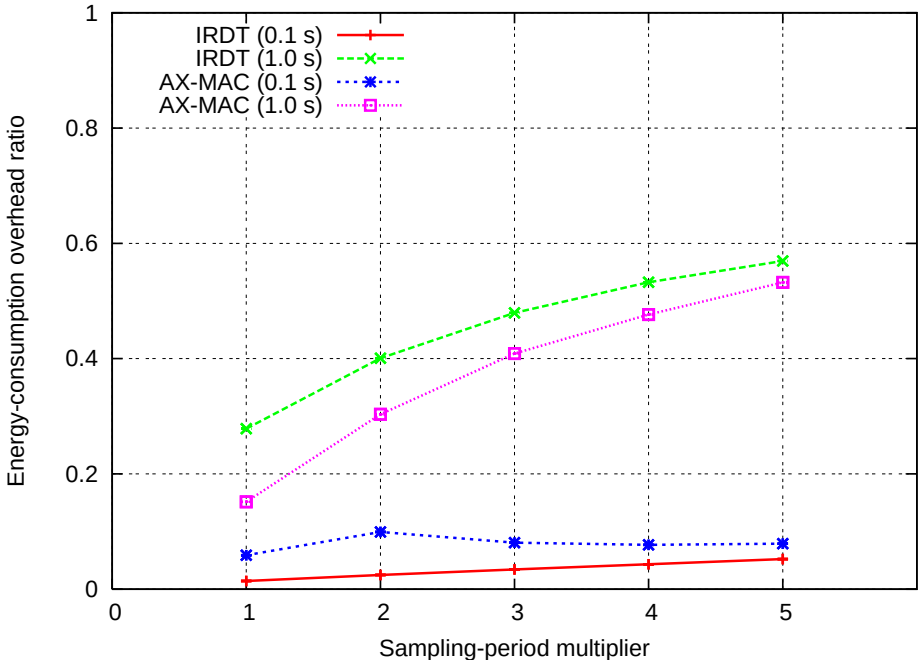


(b) Mean delay

Figure 13: The basic performance of IRDT and AX-MAC: changing the sampling period



(c) Average energy consumption



(d) Overhead of energy consumption for soft-state management

Figure 13: The basic performance of IRDT and AX-MAC: changing the sampling period

5.2 Robustness evaluation

In this subsection, we evaluate robustness of IRDT with the soft-state management of the neighbor relationship and the neighbor hop count tables against the pathway defect by the wireless channel fluctuations and the failures of nodes. In order to evaluate robustness, we use the value of the packet collection ratio when the channel condition is changed and use the behavior of packet collection ratio before and after these failures in a sensor network.

Pathway defect by the wireless channel's change

We use a Gilbert-Elliot channel model [23] to model a wireless channel fluctuation. In this model, two state discrete time Markov Chain, there are a good state and a bad state. We set the bit error rate in a good state to 0 and in a bad state to 1 respectively. All the links between all pairs of two nodes have this state respectively. Because the channel fluctuation occurs on a number of scales, we evaluate the data packet collection ratio when the cycle of the channel fluctuation is from 10^{-2} s to 10^{3} s. We set the sampling interval to three values, 30 s, 300 s, and 3000 s, and the intermittent interval and the sampling period are 1.0 s and 2.0 s respectively.

The red line in Fig. 14 indicates that the state becomes soft, a system can be more robust against the cycle of the wireless channel fluctuations when the wireless channel fluctuation cycle is larger than 10^{1} s. As for the 10^{-2} s of the fluctuation cycle, this time scale corresponds to the time for a node-to-node data transmission. The cycle from 10^{-1} s to 10^{0} s is related with the intermittent interval. Therefore, on these time scales, the packet collection ratio should be improved by MAC layer, not by routing layer. At this point, unfortunately the degradation of the collection ratio on the time scale of 10^{1} s cannot be overcome, however, this can be achieved by transport layer rather than by routing layer.

Node failures

Now, we examine the two situations to evaluate robustness against node failures as following.

- Multiple nodes failures at the same time in the network where only one sink node exists
- A sink node failure in the network that has multiple sink nodes

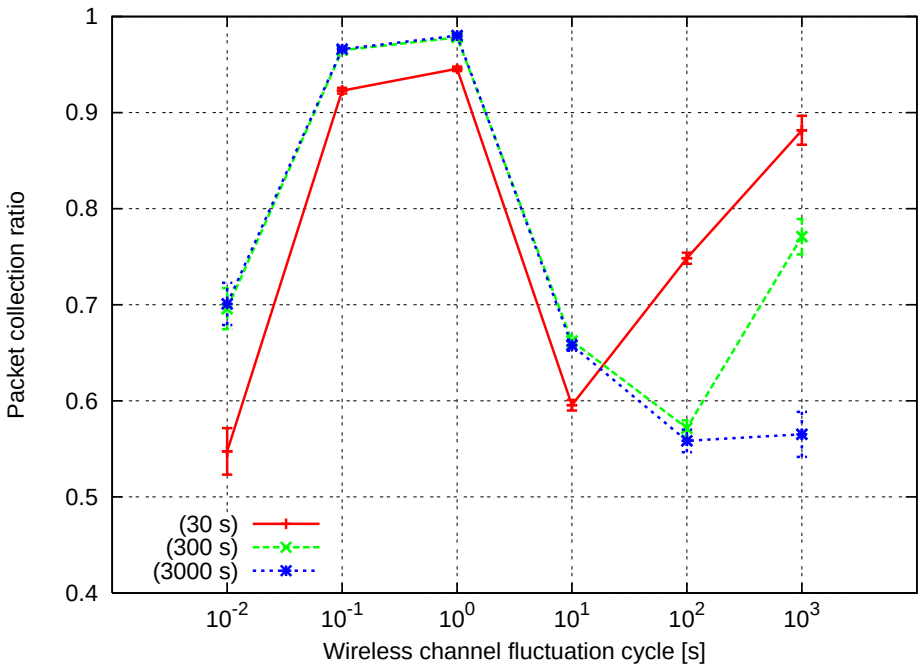


Figure 14: Result for different cycles of wireless channel fluctuations

We investigate the behaviors of the packet collection ratio every 200 second in each situation. In order to discuss only the influence of node failures, the wireless channel fluctuation doesn't occur in these situations. First, we focus on the multiple sensor nodes failures. We use the topology shown in Fig. 15 and at the 2000 second in the simulation, the red colored nodes simultaneously stop its operation by failure. The intermittent interval in IRDT is set to 1.0 s and that in AX-MAC is set to 0.1 s to obtain a high packet collection ratio.

The packet collection ratio of IRDT and AX-MAC is illustrated in Fig. 16, in which multiple nodes failure occur at 2000 second. In IRDT, after the failures, the packet collection ratio falls by less than 1% and it recovers immediately. This fall is caused when the nodes which have a data packet to send break down. The packet collection ratio after multiple nodes failed doesn't decrease very much because each node whose forward node failed can use alternate forward node. Therefore, the influence of the multiple nodes failure is hardly seen in IRDT. In AX-MAC, the packet collection ratio is also not very affected by the failures of nodes. The variance of the packet collection ratio is larger than IRDT due to the likelihood of the collisions of short preamble packets.

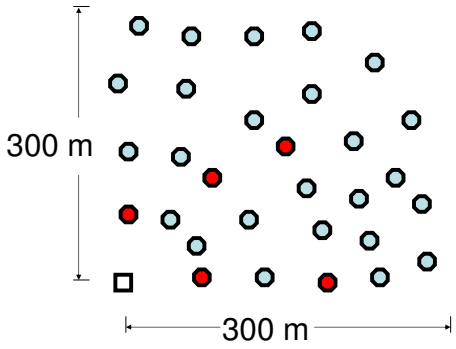


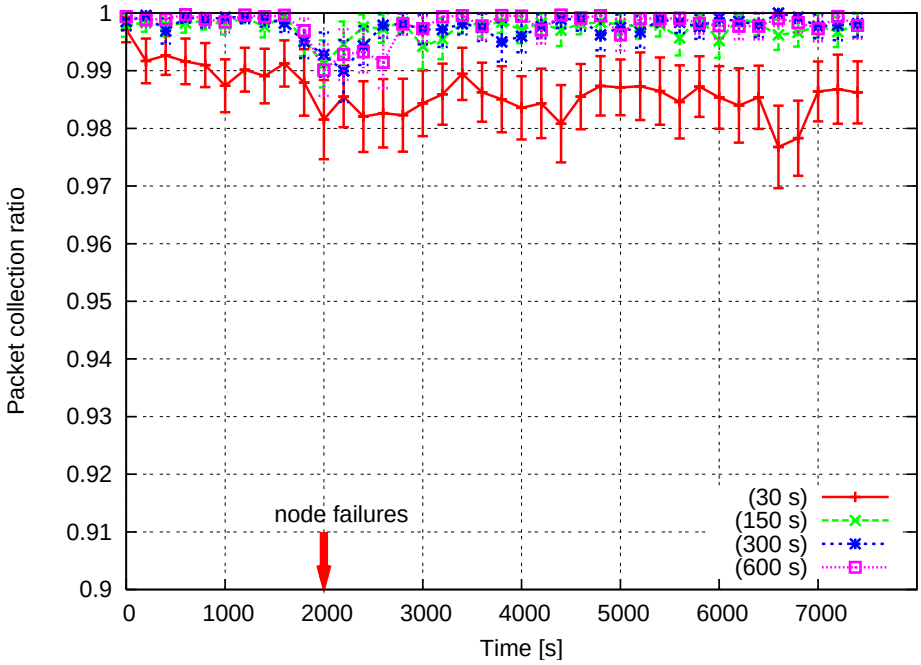
Figure 15: Network model: the red colored sensor nodes fail at 2000 second

We also show the delay of IRDT and AX-MAC with the failures of nodes in Fig. 17. The delay after the failure of multiple nodes soon increases because each node whose forward node breaks down simply wait for other forward node's ID packet. The red line of the Fig. 17(a) is higher than others, which indicates that the exchanges of the hop count tables cause interferences when the intermittent interval is long and the sampling interval is short.

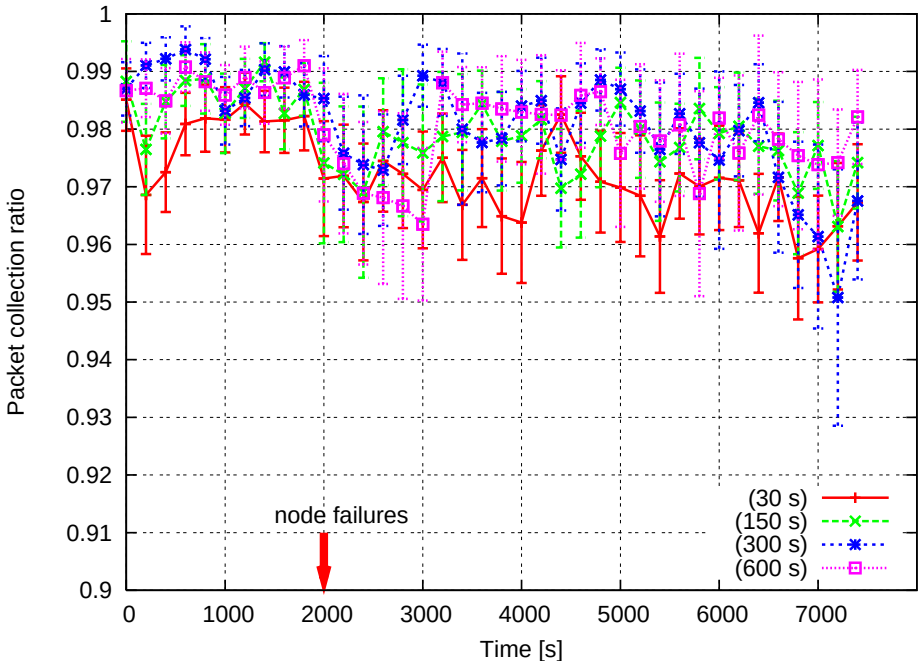
Next, we take a focus on the sink node's failures in a multi-sink sensor network. For the evaluation, we use the topology shown in Fig. 18 where 30 sensor nodes are randomly deployed over 300 m-square and 3 sink nodes are put into the three corners of the square. Each sensor node selects one sink node as a destination from which the hop count is smallest among all sink nodes according to its own hop count table. If two or more nearest sink nodes have the same hop counts, a sensor node selects one of them randomly. At the 2000 second in the simulation, the red colored sink node breaks down as is the case with multiple nodes failure. The intermittent interval in IRDT is also set to 1.0 s and that in AX-MAC is set to 0.1 s and the sampling period is two times of the intermittent interval.

The accuracy of the hop count table of each node is very significant in the case of the sink node's failure. If a node selects the disabled sink node as a destination node, a transmitted data packet wanders over the network and cannot get to any sink node.

In IRDT, the packet collection ratio decreases at the 2000 second but the sampling interval of 30 s can reduce this decrease below 10% (Fig. 19(a)). Other sampling intervals decrease the collection ratio more than 31% and recovery speed is much slower than that of the 30 s sampling interval. The similar result can be seen in AX-MAC as shown in Fig. 19(b). In AX-MAC, when

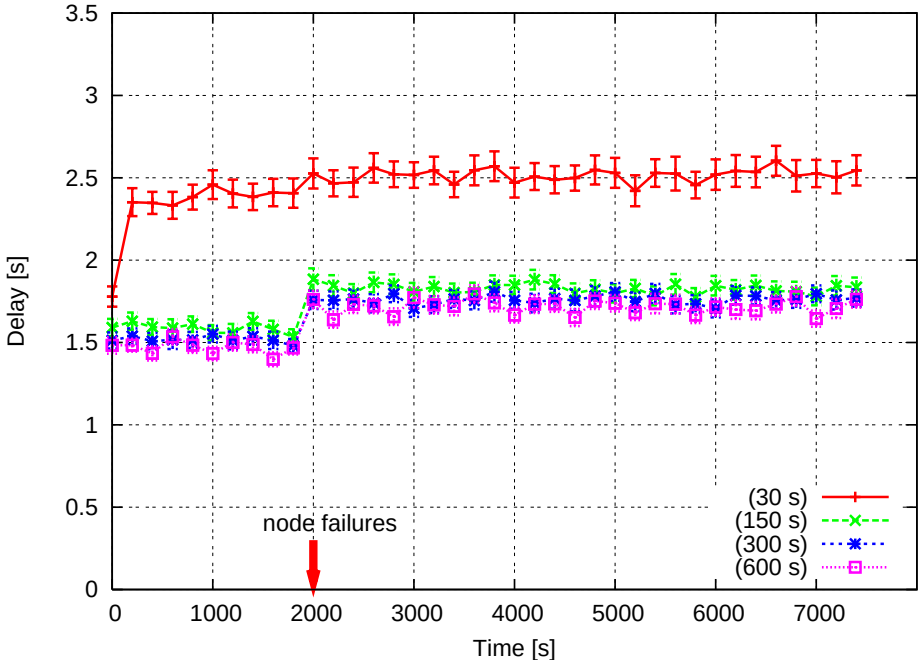


(a) IRDT

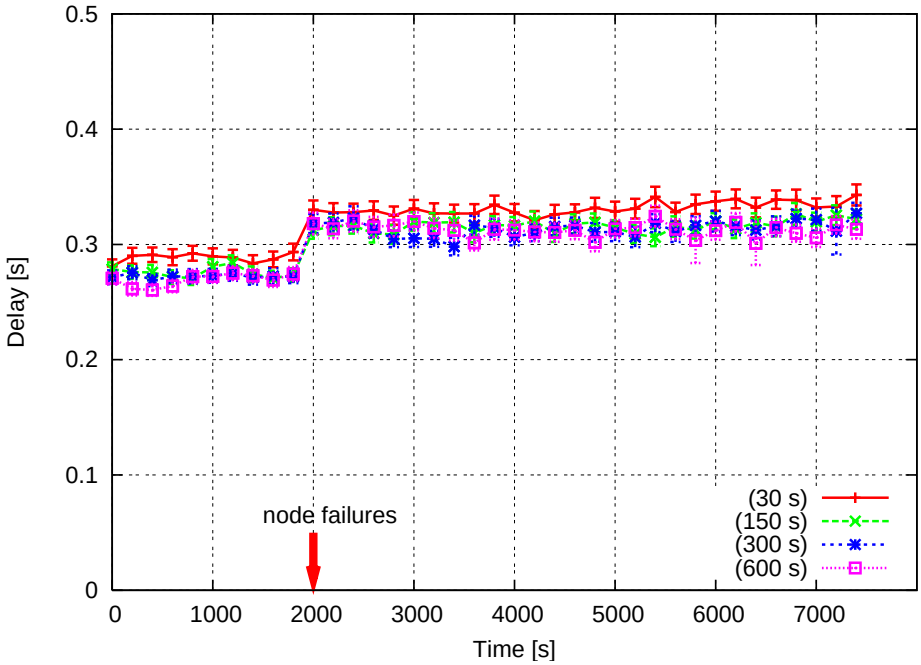


(b) AX-MAC

Figure 16: Robustness against the multiple nodes failures



(a) IRDT



(b) AX-MAC

Figure 17: Delay change against the multiple nodes failures

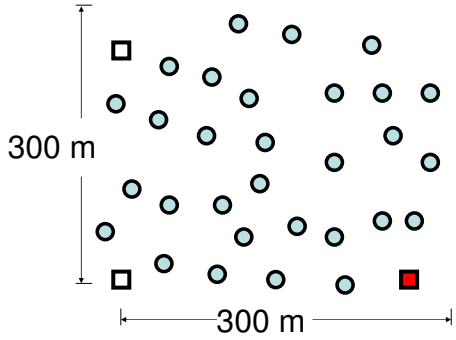


Figure 18: Network model: the red colored sink node fails at 2000 second

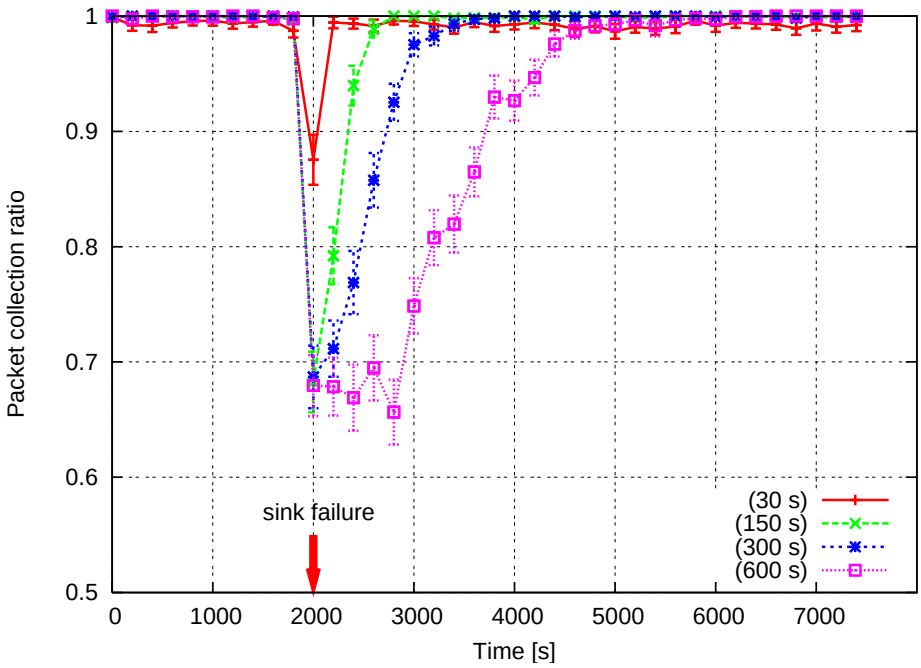
Table 2: Elapsed time for the 99 % recovery of the collection ratio after the failure

Sampling interval	Elapsed time (IRDT)	Elapsed time (AX-MAC)
30 s	200 s	200 s
150 s	600 s	800 s
300 s	1200 s	1400 s
600 s	2600 s	3200 s

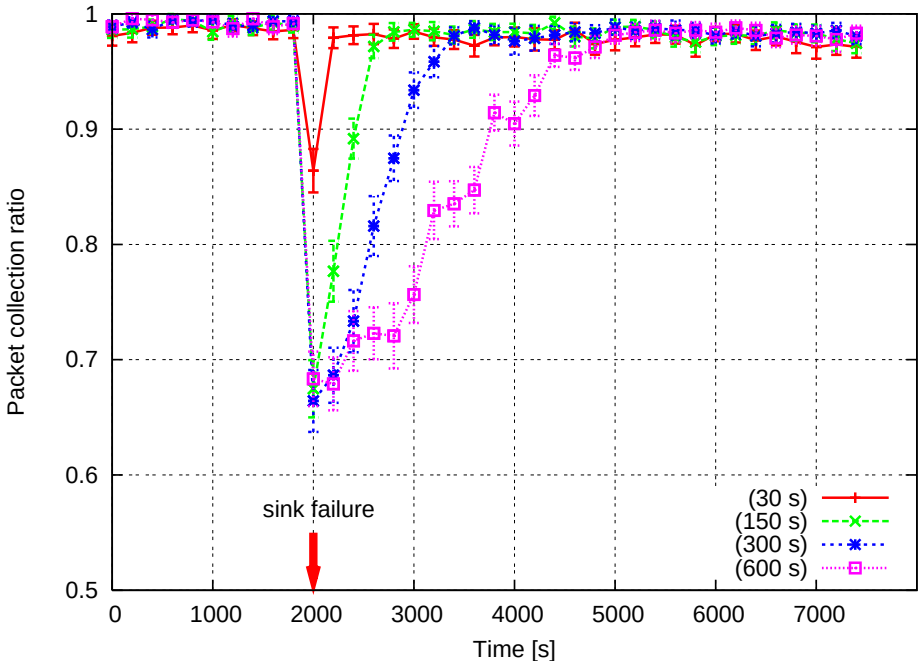
the sampling interval is 30 s, the decreaes of the packet collection ratio is 12%, and that of other sampling intervals is larger than 31%. Here, the recovery speed is defined that the time that elapses since the failure occurs until the collection ratio recovers to 99% of the ratio immediately before the failure. Table 2 shows the recovery speed of IRDT are faster than AX-MAC. The delay becomes longer after the failure and this change is also more quickly as the sampling interval is shorter as shown in Fig. 20.

5.3 Discussion

First, we discuss a compatibility between IRDT and soft-state management. IRDT uses a receiver-driven MAC protocol which has a great advantage in a long intermittent interval relative to AX-MAC that is sender-driven method as discussed in Section 5.1. A long intermittent interval is advantageous to save energy, however, it needs a longer sampling period at least more than one intermittent interval. When the sampling period is longer, the energy consumption for the soft-

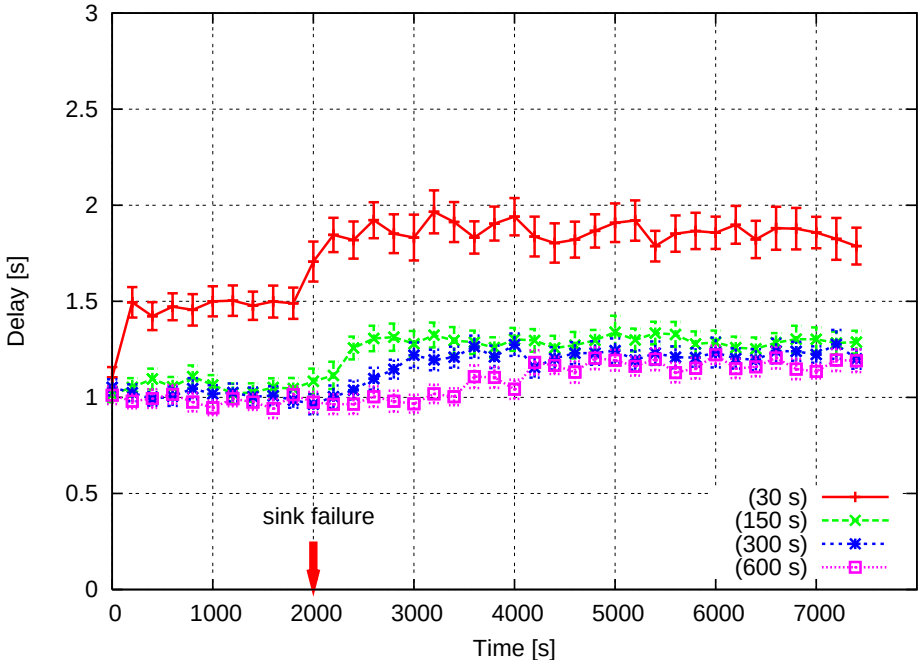


(a) IRDT

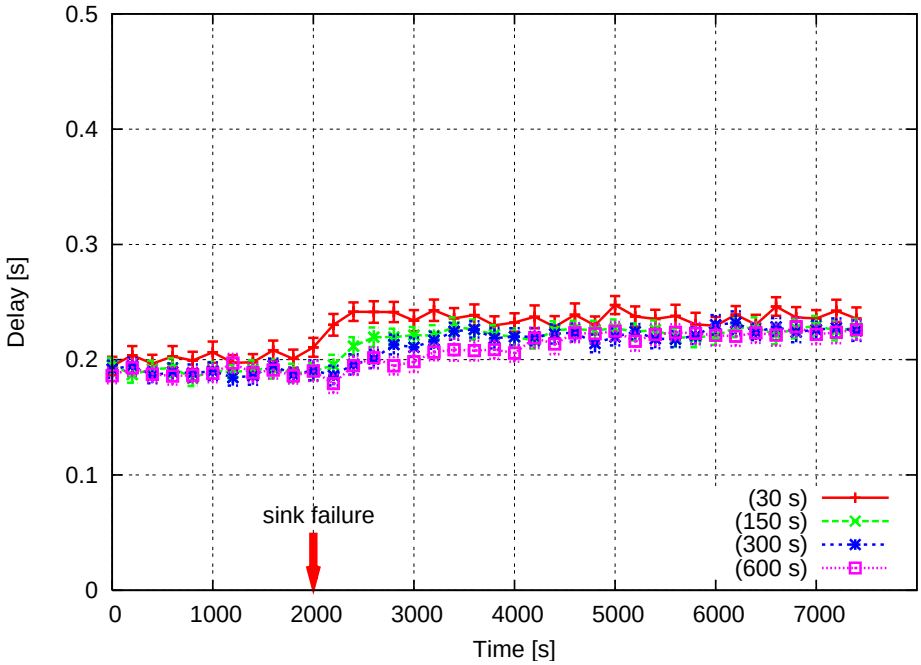


(b) AX-MAC

Figure 19: Robustness against the sink node's failure



(a) IRDT



(b) AX-MAC

Figure 20: Delay change against the sink node's failure

state management rises rapidly as the sampling interval is shorter as shown in Fig. 12(d). On the other hand, Fig. 14 shows that IRDT can shorten the sampling period more than AX-MAC, which can reduce the energy consumption. In terms of the energy consumption, IRDT does not necessarily outperform AX-MAC when they use soft-state management, and vice versa because the sampling interval and the sampling period should be adaptively to environmental conditions. In the light of the packet collection ratio, IRDT with the 1.0 s intermittent interval can achieve higher performance than AX-MAC at any sampling interval and any sampling period. If the wireless channel fluctuations and node failures less occur, IRDT with a longer intermittent interval can use a shorter sampling period, which doesn't drop the packet collection ratio and is able to save energy as shown in Fig. 13(a) and Fig. 13(c). This indicates that sampling and table exchange have a relatively small effect on the packet collection ratio, that is the soft-state management is compatible with IRDT. However, the energy consumption has significant increase in IRDT with long intermittent interval which originally intends to save energy. We have to solve this problem hereafter.

Next, we discuss robustness of IRDT. Both IRDT and AX-MAC are originally robust because these protocols construct a mesh network. This can be seen in Fig. 16 where multiple nodes failures don't have much influence on the packet collection ratio. It is expected that the multiple link failures also don't affect on the collection ratio thanks to the alternate pathway. The wireless channel fluctuation in our simulation can be thought as the random link failures. In Fig. 14, we investigate the packet collection ratio under the severe fluctuation where the probability that a channel is in a good state of Gilbert-Elliot model is less 40%. In other words, each link fails and recovers every fluctuation cycle, at least 50% of data packets can be collected. On the critical situation that the sink node breaks down, the packet collection ratio hardly decreases and rapidly recovers when the sampling interval is 30 s. Comparing Fig. 19(a) with Fig. 19(b), the collection ratio of IRDT is higher than that of AX-MAC in particular when the sampling interval is 30 s. It can be said that robustness that IRDT originally has is effective only when the destination exists and robustness against emergent changes is improved by the soft-state management.

6 Conclusion and Future Work

In this thesis, we proposed the soft-state management of the neighbor relationship and the soft-state management of the neighbor hop count table for the receiver-driven asynchronous system IRDT. We also evaluated the basic performance and robustness of IRDT with the soft-state management through comparison with IRDT with hard-state management and comparison with AX-MAC, which is a sender-driven asynchronous system, by using computer simulation. As a result, we verified that IRDT is compatible with the soft-state management in terms of the packet collection ratio, however, more improvements need for saving energy. As for robustness against the wireless channel fading and node failures, the degradation of the packet collection ratio was reduced by 67% and the time for 99% recovery of the packet collection ratio before failures became shortened by 2400 s.

Now, our concern is saving energy when IRDT uses the soft-state management. We believe that the energy consumption can be more suppressed by improving the procedure of the sampling and the procedure for exchanging tables. These are our future works.

Acknowledgements

I would like to express my great gratitude to Professor Masayuki Murata of Osaka University for his generous guidance and insightful comments.

Also, I would like to express my deep appreciation for Professor Masashi Sugano of Osaka Prefecture University. Without his guidance and support, I cannot complete this thesis.

I am deeply grateful to Professors Koso Murakami, Makoto Imase, Teruo Higashino, and Hirotaka Nakano of Osaka University, for their appropriate guidance. I would also like to thank to Associate Professor Naoki Wakamiya, Specially Appointed Associate Professor Kenji Leibnitz, Asistant Professor Shin'ichi Arakawa, and Assistant Professor Yuichi Ohsita. Their helpful suggestions and kindly support on my research is invaluable.

Finally, I express my appreciation to my friends and colleagues of the Advanced Network Architecture Research Group of Osaka University for their support on my research and advices in daily life.

References

[1] X. Du, Y. Xiao, and F. Dai, "Increasing Network Lifetime by Balancing Node Energy Consumption in Heterogeneous Sensor Networks," *Wireless Communications and Mobile Computing*, vol. 8, no. 1, pp. 125–136, 2008.

[2] S. J. Baek and G. de Veciana, "Spatial Energy Balancing through Proactive Multipath Routing in Wireless Multihop Networks," *IEEE/ACM Transaction on Networking*, vol. 15, no. 1, pp. 93–104, 2007.

[3] K. Pister and L. Doherty, "TSMP: Time Synchronized Mesh Protocol," in *Proceedings of the 20th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS)*, 2008.

[4] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (Sensys)*, pp. 95–107, 2004.

[5] "MICA2." available at `http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf`.

[6] Y. Wei, H. John, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 1567–1576, 2002.

[7] I. Rhee, A. Warrier, M. Aia, J. Min, and M. L. Sichitiu, "Z-MAC: A Hybrid MAC for Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, no. 3, pp. 511–524, 2008.

[8] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (Sensys)*, pp. 307–320, 2006.

[9] R. Jurdak, P. Baldi, and C. V. Lopes, "Adaptive Low Power Listening for Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 8, pp. 988–1004, 2007.

[10] T. Hatauchi, Y. Fukuyama, M. Ishii, and T. Shikura, “A Power Efficient Access Method by Polling for Wireless Mesh Networks,” *IEEJ Transactions on Electronics, Information and Systems*, vol. 128, no. 12, pp. 1761–1766, 2008.

[11] E. A. Lin, J. M. Rabaey, and A. Wolisz, “Power-Efficient Rendez-vous Schemes for Dense Wireless Sensor Networks,” in *Proceedings of the IEEE International Conference on Communications*, vol. 7, pp. 3769–3776, 2004.

[12] J. Kim, J. On, S. Kim, and J. Lee, “Performance Evaluation of Synchronous and Asynchronous MAC Protocols for Wireless Sensor Networks,” in *Proceedings of International Conference on Sensor Technologies and Applications*, pp. 500–506, 2008.

[13] D. Kominami, M. Sugano, M. Murata, T. Hatauchi, and Y. Fukuyama, “Performance Evaluation of Intermittent Receiver-driven Data Transmission on Wireless Sensor Networks,” in *Proceedings of the 6th International Symposium on Wireless Communication Systems (ISWCS)*, pp. 141–145, 2009.

[14] J. C. Lui, V. Misra, and D. Rubenstein, “On the Robustness of Soft State Protocols,” in *Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP)*, 2004.

[15] N. Hashimoto, H. Taka, H. Uehara, and T. Ohira, “Attribute-based Relay Control for Reduction of Latency in Duty-cycled Wireless Sensor Networks,” *IEICE technical report, USN2008-44*, pp. 33–38, 2008.

[16] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, “RSVP: A new Resource ReSerVation Protocol,” *IEEE network*, pp. 8–18, 1993.

[17] M. Handley, H. Schulzrinne, E.Schooler, and J. Rosenberg, “SIP: Session Initiation Protocol,” *RFC 2543, Internet Engineering Task Force*, 1999.

[18] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. G. Liu, and L. Wei, “An Architecture for Wide-Area Multicast Routing,” in *Proceedings of the conference on Communications architectures, protocols and applications*, pp. 126–135, 1994.

[19] M. Handley, C. Perkins, and E. Whelan, “Session Announcement Protocol,” *RFC 2974, Internet Engineering Task Force*, 2000.

[20] S. Raman and S. McCanne, “A Model, Analysis, and Protocol Framework for Soft State-based Communication,” in *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, pp. 15–25, 1999.

[21] T. Yamaguchi, H. Tada, and M. Imase, “Evaluation of Robustness of Soft State for Data Management in Distributed Environment,” *IEICE technical report, IN2005-111*, pp. 7–12, 2005.

[22] F. Kojima, H. Harada, T. Hatauchi, M. Tanabe, K. Sakamoto, A. Kashiwagi, T. Banno, and H. Nishiyama, “Low energy MAC for non-beacon enabled PAN.” available at `https://mentor.ieee.org/802.15/dcn/09/15-09-0594-01-004e-low-energy-mac-for-non-beacon-enabled-pan.pdf`.

[23] E.N. Gilbert and et al, “Capacity of a Burst-Noise Channel,” *Bell. System Technical Journal*, vol. 39, no. 9, pp. 1253–1265, 1960.