A Design for Wireless Sensor Network Visualization Tools Based on Network Management Principles

Yuki Urata†, Takuya Yoshihiro‡, and Yutaka Kawahashi*

†Graduate School of Systems Engineering, Wakayama University, Japan
‡Faculty of Systems Engineering, Wakayama University, Japan
* Center for Information Science, Wakayama University, Japan
{s171009, tac, yutaka}@center.wakayama-u.ac.jp

Abstract - Wireless Sensor Networks (WSNs) are considered as a key technology of up-coming IoT applications in the world. Several studies have indicated the requirements on visualization tools to support administrators for practical operation and management of WSNs. Although they proposed various visualization system design as well as diagnosis methods, no firm principle has been suggested on designing administrative systems to monitor WSNs. We in this paper provide a new suggestion on designing WSNs management system to introduce the principle of the network management grown in long time with the experiment of the Internet. Note that we cannot apply query-based information acquisition such as SNMP since duty-cycled sensor devices cannot reply in real-time. So, under the assumption of deploying the passive management, in which the values required in WSNs management is continuously collected to the sink by piggy-backing them on data packets, we provide a new design of visualization tools that suffices the practical requirements on managing WSNs based on the three network management aspects: structure, failure, and performance managements. We developed a prototype of the proposed visualization system, and conducted an evaluation experiment. The results demonstrate that the proposed system design surely supports network operators to meets all requirements on WSNs management.

Keywords: Wireless Sensor Networks, Operation, Management, Visualization

1 INTRODUCTION

Wireless Sensor Networks (WSNs) are expected as a practically important technology to develop the up-coming IoT (Internet of Things) applications that will improve the quality of our lives. Many applications of WSNs including agriculture, medical cares, factory automations, environmental monitoring, etc. have been considered in many research work to improve each area of human activities. However, to realize practical WSNs, realizing reliable communications within very low power consumption are essentially important. For this challenge, various communication protocols as well as sensor-node hardware that consume extremely low power have been developed so far.

As a well-known communication standard for sensor networks, IEEE802.15.4[1] has been promoted, and many commodity devices have been developed. However, IEEE802.15.4 in fact consumes considerable power because it is designed to cover wide variety of practical scenarios, and also to include many functions to enhance flexibility of communications. Thus, to achieve a minimum power consumption level to collect sensed values to sink nodes, lots of low-power MAC protocols for WSNs have been proposed [2]-[5].

In these MAC protocols, each node takes as much sleep time as possible to save its power consumption, by efficiently synchronizing the communication timing between a sender and a receiver nodes within a limited awaking time. From the viewpoint of synchronization mechanism, they are classified into two types of MAC protocols, sender-initiated and receiver-initiated protocols. Sender-initiated MAC protocols such as B-MAC[2] and X-MAC[3] synchronize wake-up timing using actions of sender nodes. For example, B-MAC transmits a long preamble that is longer than the wake-up interval to enable receivers being awake state when a transmitter transmits a frame. Receiver-initiated MAC protocols such as RI-MAC[4] and RC-MAC[5] synchronize wake-up timing based on the action of receiver nodes. Typically, like RI-MAC, receivers periodically transmit beacons when they are ready to receive frames, while a sender that has a data frame keeps waking-up and waits for beacons to transmit the frame. Recently, receiver-initiated MAC protocols are regarded as one of the promising approaches for practical low-power WSN systems so that many proposals have appeared in the literature [6][7].

On the other hand, similar to the networks connected to the Internet, we have to manage and operate WSNs in practice to keep WSNs work correctly to gather sensed data values. To this end, We need a system with which we can watch the state of WSNs and find the trouble as soon as it occurs so that we can take effective measures against the troubles to maintain WSNs to work continuously and correctly. There are several tools and methods developed for this purpose. In early days, query-based methods to collect information of WSNs has been tried, which is a similar approach to SNMP [8] in the Internet management. However, in low-power duty-cycle MAC protocols, such queries require significantly long delay due to long sleeping time of each node, and thus it is not a possible approach in WSNs. Alternatively, one of the basic approaches is to collect required values to sink nodes by piggy-backing them on data packets to find anomalies of WSNs. There are several proposals [9] [10] from this approach.

These methods carefully model the relationship among possible events to specify the essential causes of troubles. However, since the events that are considered in them are limited to

the ones that are supposed in advance, new phenomena cannot be treated. As the Internet management process poses, the troubles in network management have too large variety, so that those proposals cannot cover all possible cases. Also, finding a small sign of trouble before it occurs is an important issue for prevention of troubles in managing networks. Thus, in addition to the predefined diagnose systems, we need a system that visualizes the state of WSNs to help finding troubles or their prior signs in managing and operating WSNs.

In the literature, several systems that visualize WSNs exist [11]. However, they are not designed from the viewpoint of managing networks to utilize the experiment of Internet management, nor evaluated from the viewpoint.

In this paper, we propose a new design of systems that visualize the state of WSNs and help managing them. Our system design is based on the principle of Internet management, i.e., we designed to perform (a) structure management, (b) failure management, and (c) performance management, in order to keep stable operation of WSNs. We implemented and evaluated our system using simulation traces of a WSN to confirm that the system enables users to find important signs from the viewpoint of management principles (a)-(c).

This paper is organized as follows. In Sec. 2, we present related work to manage WSNs. In Sec. 3, we describe the design policy of WSN management systems. Especially, we discuss on the principal of Internet management and requirements for WSN management systems. In Sec. 4, we present the proposed system, and we evaluate the system in Sec.5. Finally in Sec.6, we conclude the work.

2 RELATED WORK

There are several related studies and systems that visualize and help managing WSNs. In this section, we describe these previous contributions.

It has been pointed out from the early stage of WSN studies that the behavior as well as the root causes of troubles in WSNs are hard to look out. Thus, several studies have been proposed to diagnose WSNs in face of troubles to find out what is going on in WSNs. Note that the cause of troubles can be inside the sensor node, i.e., software bugs that happen only in multiple-node-related scenarios are hard to eliminate in the development phase, and require a tool to help debugging. Consequently, both realfield operation and development testing can be the target of those diagnosis methods and tools.

As specific proposals, Ramanathan et al. proposed a diagnosis method called Sympathy [9] that construct a decision diagram from passively collected data at sink nodes to find out the root cause of the trouble. Liu et al. proposed a method called PAD [10] that uses causal diagram to find out the root cause. Liu et al. also proposed a method TinyD2 in which multiple nodes cooperatively work to find out the root cause of trouble [12]. Khan et al. proposed a debugging method DustMiner [13] that utilizes a set of event logs in WSNs to find the sequences of events that lead to reveal bugs and troubles. However, those methods make diagnosis from predefined causes and troubles, so that they are not possible to treat new phenomena and troubles. Unfortunately, in the real Internet management, system administrators often meet new kind of troubles so that flexible responses are required.

As another approach, several methods learn the normal state of WSNs and detect abnormal states, i.e., anomalies, based on the distance between the current state and the normal state. For example of this class of methods, Li et al. proposed a method VN2 that learns the normal state using 43 metrics of WSNs from event history logs and apply Non-negative Matrix Factorization (NMF) model to detect exceptional state of WSNs [14]. Miao et al. proposed a method called Agnostic Diagnosis (AD) that computes correlations between 22 metrics to detect anomalies from the history logs. Those systems are useful to reduce labor and time of administrators to detect anomalies, but in managing WSNs [15], the detail and the root causes must be carefully examined anyway using some visualization tools to clarify the occuring phenomena and take a proper countermeasure.

As a visualization tools for WSNs, there are several studies and systems such as Octopus [16], SRNET[17], etc. Also, several visualization tools have been provided as bundled software in WSNs devices or implementations; they are concisely introduced in a survey paper [11]. However, they all are not designed based on the principle of network management, nor evaluated from that viewpoint. Thus, the efficacy of those tools in WSN management is not sufficiently clarified. This paper is the first study that presents the WSN administration tool designed based on the principle of the Internet Management.

3 REQUIREMENTS AND THE DESIGN

3.1 Requirements for WSN Management Systems

In this study, we suppose wireless sensor networks (WSNs) that deploy low-energy MAC protocols such as X-MAC and RI-MAC, so that any query-based management protocol such as SNMP cannot be used to manage them. The basic way to manage this kind of WSNs is to apply passive strategy as presented in [9][10], where administrative data values are collected to sinks piggy-backed on data packets, and network states/problems are visualized/inferred to manage networks properly.

In managing WSNs, we are required to carefully watch the networks to keep them correctly working for collecting sensed data values. For this purpose, not only detecting the problems occurring in the networks, but also finding implicit signs of future problems is important, to prevent degradation of communication performance or extra power consumption due to redundant network behaviors. One of our primary goals is to make network administrators possible to find these signs surely and speedily by visualizing the values collected at sinks. Note that it is difficult to manage multi-hop WSNs in real time as long as it is managed in the passive way, i.e., managed through the values collected at sink nodes. In other words, we have to allow a certain level of delay on finding failures or troubles in the network management tasks. If we have a requirement on the delay performance, they should be covered by the deployed MAC and routing protocols. Accordingly, we in this paper focus on just inquiring the causes

of the failures without caring the time delay that takes to find them.

On the other side, in the Internet, network management has been a critically important issue to provide stable and secure services to end users since the Internet now has a role of social Infrastructure. In this context, the area of network management has naturally been grown in the past decade and formed a rough but firm consensus on how to manage networks. Such consensus is issued as some documents, e.g., reference [18] describes the required knowledge and administrative operations that should be performed in managing networks connected to the Internet. The document says that the network management consists of 5 specific aspects of management domains shown in the following.

Structure Management
Structure management is the task that maintains physical and logical elements in networks. Network structure is the basis of all management task, so it is siginificantly important to grasp the latest state of the network structure. In the Internet management, the network structure consists of all the physical elements such as network boxes, the logical elements such as virtual functions and configurations, and also their connections. In contrast, the structure of current WSNs is quite simple in which sensor devices, their configurations, neighbor relationship, and the paths from each node to sinks are included.

Failure Management
Failure management is the task (1) that defines the event regarded as failure, (2) considers the detecting strategy, countermeasures and preventive measures for each failure, (3) and executes them. We have two types of the failure detection approaches: active detection based on queries sent to each device, and passive detection based on the reports coming from each device. In case of this study, since we suppose WSNs which deploy low-power MAC protocols, we have no choice but taking passive detection approaches. Also, in WSNs, we mainly considers node and link failures.

Performance Management
Performance management is the task that maintains WSNs to keep a constant level of communication performance. Generally, the performance of networks include such as throughput, packet loss ratio, latency, jitter, retransmission count, congestion frequency, CPU usage, etc. In WSNs in this study, we expect each packet to reach a sink reliably within a certain latency. Thus, especially packet loss ratio and latency are the important measure of the performance.

Resource Management
Resource management is the task that maintains the system resources required in the operation of systems. The resource includes every elements that consists of the system such as hardware, software, cables, etc. Note that resource management includes CPU, memory, and network capacity management that prevents shortage of those dynamically used resources.

Security Management
Security management is the task that protects the system and the contents from the threat outside of the network. In the Internet, security is a very important issue since there are several threat such as viruses and attacks from outside the network. In WSNs in contrast, the main concern is the information leaking, which can be prevented by deploying some encryption facility.

In this study, we only consider three management aspects, i.e., structure, failure, and performance management, and omit considering resource and security management. Although resource management includes CPU, memory, and communication capacity management, they are not the matter in WSNs; WSNs use small amount of CPU and memory resources so that they are not important to manage in most cases. Communication capacity of links is an important issue in WSNs, but in wireless networks, communication capacity management is tightly connected to the performance management, since communication performance gradually degrades as communication amount approaches the capacity. As above, in WSNs, three management issues, i.e., structure, failure, and performance management, are essentially important.

3.2 The System Design

For structure, failure, and performance management, the system must collect the required administrative information from WSNs and visualize them appropreately to help operators manage WSNs properly. Collecting information is done such that each node measures several administrative values and include them in the packet that the node generates. By piggy-backing the values required for management on packets, passive management using the values gathered on the sink nodes is enabled. Note that the administrative values are not added at each node in the collection paths, but added only at the originated node of the packet. Therefore, the overhead incurred from the administrative values does not change depending on the size of WSNs.

User interface is designed in order for operators to easily find events related with the three management aspects. For structure management, we prepare the *delivery tree view* to see the network state intuitively. Every node is placed at the right position and the set of next-hop links at an arbitrary time point are shown to form the delivery tree. We can overlap two delivery trees at different time points, which enables us to see the transition of the delivery tree.

In failure management of WSNs, we find node or link failure. When the packets from a node do not arrive at sinks, we can regard that the node fails, and we can detect link failure in the similar way. Thus, we prepare the *alert table* in which possible node failure events as well as other alerts are listed up. When operators found possible node failure in the *alert table*, they usually have to check whether the failure really occurs or not. To do this, we prepare the *administrative values table* in which all the data values collected in sinks are seen

per source node, and also prepare the *line graph view* that visualize the data values per node and per item to see the values intuitively.

Finally, for performance management, we mainly watch residual power, packet loss ratio, and delivery delay at each node. By listing the events in the *alert table* when these values exceed a threshold, the operators easily find the performance degradation. After the operator is notified of the performance anomalies, they usually explore for the level of the degradation and specify what is the root cause of this trouble. We can use the *administrative values table* and the *line graph view* again for this purpose.

With the basic design policy described above, our system has the following characteristics that differentiate our system from the others.

(1) Visualizing Delivery Tree Transition
The function of overlaid display of multiple delivery trees obtained from different time points is unique to our system, which enables operators to grasp the transition of delivery paths easily and intuitively. Since this view provides an intuitive overview information, this view plays a role of the 'base' page from which we can explore several detailed data values.

(2) Alert Table to be Aware of Administrative Events
We prepare the *alert table* that makes administrators keep aware of important administrative events occurred in WSNs. By simply applying thresholds to several carefully-selected administrative data items that represent network performance, administrators can watch WSNs via *alert table* to find important events.

(3) Intuitive Operation
From the base *delivery-tree view*, we can intuitively transit to other views by clicking entities nodes and buttons. In our user interface design, the administrators are possible to access the related data values to explore for the state of WSNs.

4 THE PROPOSED SYSTEM

4.1 System Structure

The proposed system visualize the network state from the administrative values collected to sinks. We show the system structure in Fig. 1. A sink node collects the sensed values generated periodically at every node. Note that there may be multiple sink nodes in a WSN, but the server collects values from all sink nodes. The server provides the function of web servers so that the administrators access to the server via Web browsers to visualize the state of WSNs.

4.2 Attaching Administrative Values to Packets

In our framework, we piggy-back the administrative values on packets to collect them to the sinks in WSNs. As the administrative values, we used the following 18 items that are typically used in administrating networks. Note that every item is measured at each node by itself.

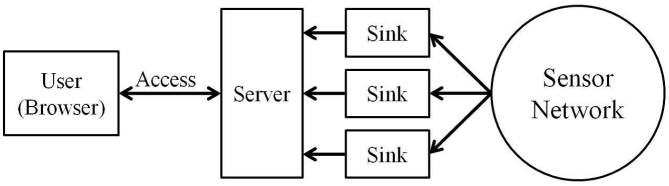

Figure 1: System Structure

(a) Reception Time at Sink

(b) Generated Time of Sensed Values

(c) Sink Node ID

(d) Source Node ID

(e) Sequence ID

(f) Parent Node ID (Next-hop of Source Node)

(g) Number of Transmitted Frames per Unit Time

(h) Number of Received Frames per Unit Time

(i) Number of Transmitted ACKs per Unit Time

(j) Number of Received ACKs per Unit Time

(k) Number of Transmitted Control Messages per Unit Time

(l) Number of Received Control Messages per Unit Time

(m) Accumulated Awaking Time per Unit Time

(n) Accumulated Sleep Time per Unit Time

(o) Residual Power

(p) Sensor Coordinate (If device is with GPS)

Most items above are well-defined but we would add an explanation for several items. Note that these administrative values are added to data packets only when a sensor value is measured and the corresponding data packets are generated, and not added at the relay nodes.

Item (e) is a value uniquely assigned to each packet by a node, which is used to check the loss of data packets. Items (g)-(l) are the values measured per unit time, where typically counting is done after previous sensed-value generation since we assume sensing is done periodically at each node. Note that the number of transmissions (g)(i)(k) include not only the frame generated at the node but also the frame that the node relays to sinks. Item (o) represents residual power at each node when the packet is generated.

Sensor location (p) may be collected at sink, but we have to consider that many sensor-node devices do not have GPS due to large power consumption, and also due to relatively large errors in computing positions.

Next, we describe the reasons for our choosing these values, and the necessity. Items (a)-(e) are the most basic values for understanding the flow of each packets, and are essential in network management. Item (f) is necessary to grasp delivery tree of the packet. Items (g)-(j) are also absolutely necessary to grasp communication state in each node in every unit

time, such as how much communication is successful. Items (k), (l) are important index to confirm that operation of routing protocols is correct. If abnormality of protocols are suspected, a success rate of transmission and reception of these control messages will be important information for making a decision. Items (m), (n), (o) are necessary for grasping state and detecting troubles about power of sensor nodes. In WSN that the remaining battery is essential measure, these items are high importance, because these are closely related to power consumption. Item (p) is required, when we collect positional information by GPS attached to sensor nodes. As explained above, minimal information that is required to grasp state of WSN are contained in the 16 administrative values which are treated by this system.

From the items (a)-(p) above, we compute several values useful for managing WSNs shown as follows.

(q) Packet Loss Ratio of the Next-hop Link per Unit Time

(r) Delivery Delay

(s) Elapsed Time after Last Frame Reception from Each Node at Sink

Item (q) represents a quality of next-hop link that is computed from the transmission count (g) and the ACK reception count (j) using the formula (q)= $\{((g)-(j))/(g)\}\times 100[\%]$. Item (r) is the average time of packets taken to travel to sinks from the packet is generated. This value is computed as an average of (r)=(a)−(b) for each packet. Item (s) implies a potential anomaly if it is far larger than the sensing time interval. This value is computed from item (a): if we let a_n be the arrival time of a packet with sequence number n, (s)= $a_n - a_{n-1}$.

Note that those three values (o), (q)-(s) are especially important in managing WSNs since worse values immediately imply performance degradation of WSNs. Thus, in our system, we set a threshold value for each of (o), (q), (r), and (s) so that the system can notify administrators of the abnormal state through the *alert table*.

4.3 User Interface

4.3.1 Transition of Views

The transition of user's view in our system is shown in Fig. 2. In the top view, the delivery tree is displayed to show the overview of the current state of the WSN. The administrative values table and the alert table are aside of delivery tree to show the specific data values and important alerts to notice. In the delivery tree view, the delivery tree of a specific time point is displayed with several important information items. In the administrative values table, list of administrative values received at sinks are displayed to check the specific values and states of WSNs. In the alert table, to help being aware of important events and states of WSNs, list of automatically detected events are displayed.

At the top of each item in administrative values table, we placed a button that pop-up a new window and display the

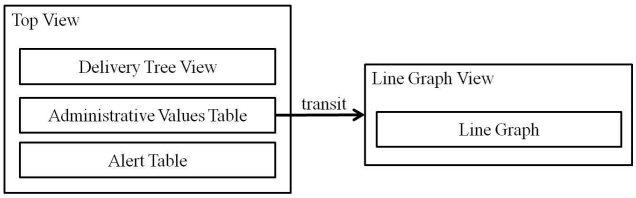

Figure 2: User Interface

line graph of the specified data item. Line graphs enable administrators to understand the tendency of value transition intuitively. In combination of those four components, administrators can watch the state of WSNs and explore the detailed behavior to find what is going on under troubles.

4.3.2 Delivery Tree View

Delivery tree view displays the delivery tree at the specific time point. Normally it would display the latest tree, but user can specify arbitrary time to see the tree at that time point. Delivery tree basically consists of a set of nodes which are placed at the right position and the set of next-hops to which each node forwards packets destined to sinks. In managing WSNs, we heve to know the place of each node, so we assume that the coordinate of each node is known by some mean, for example, GPS equipped to each sensor node, or static map that include the coordinate of each node. To show the transition of the delivery tree in time, our view has a function to overlay the past or the future delivery tree over the tree of the current time point. See Fig. 3 for this overlaid view. The current delivery tree is shown with solid blue lines whereas the past tree with dotted pink lines. The past tree consists of the previous next-hop links that are different from the current one and are reported not before the time $c - t$ where c is the current time and t is a predetermined threshold. Overlaying the future delivery tree is done in the similar way. By labeling links with the time reported (i.e., the generation time of reported packets), administrators can understand the transition of trees with exact reported time.

4.3.3 Administrative Values Table

The administrative values table shows all the administrative values collected at sinks per node. We show an example of the administrative values table in Fig.4. By clicking a node in the delivery tree, the administrative values table is updated to show the values sent from the clicked node around the time specified in the delivery tree view. With the administrative values table, administrators can refer the required administrative values intuitively and efficiently from large amount of data. Also, the button on top of each column invokes the line-graph window to see the data values intuitively.

4.3.4 Alert Table

The alert table has a role to keep administrators being aware of the important signs of WSN state changes. Especially, performance changes are essentially important to notify since they are invisible in the delivery tree so that not easy to notice

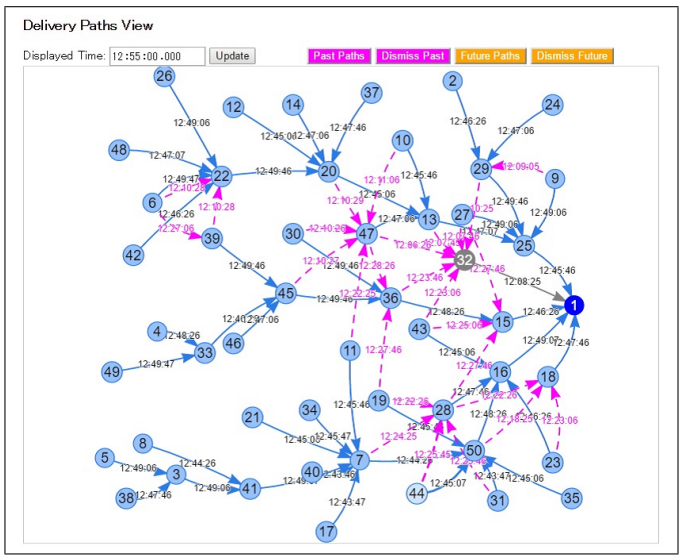

Figure 3: Delivery Tree View (When Two Time-points are Overlayed)

for administrators. To thid end, as mentioned in Sec.4.3.2, we display alert messages on communication performance by applying a threshold for each data items (o), (q), (r), and (s). An example of the alert table is shown in Fig. 5.

4.3.5 Line Graph View

The line graph shows the values listed in the administrative values table in the line graph fashion to enable administrators to grasp the trend of values intuitively. An example of the line graph view is shown in Fig. 6. As shown in this example, administrative values are plotted in time series where the horizontal axis is the reported time of the value.

5 EVALUATION

5.1 Implementation

The proposed system is implemented as a web application implemented using javaScript and AJAX. We use a library vis.js[19] for graph visualization, and highcharts[20] for drawing line graphs. In our system implementation, our application runs on Apache ver.2.4.6[21].

5.2 The Evaluation Method

We run simulation of WSNs to obtain a trace from a self-designed scenario, and visualize it using the proposed system. For the simulation, we implement a receiver-initiated MAC protocol combined with a low-power routing protocol proposed in [6]. We implement these protocols on Contiki[22], which is an OS for sensor network devices, and simulated it over simulator Cooja included in Contiki OS package. As a simulation scenario, we placed 50 nodes at a random coordinate in a 200[m]×200[m] rectangular field, and we placed a sink node on the central position of the left side of the field. Each node generates a sensed value in 5[min] interval, and

Rcv .Time	Seq .No	Next hop	#Data Rcv	#Data Tmt	#Ack Tmt	#Ack Rcv	#Ctl Tmt	#Ctl Rcv	#Data Rcv (Direct)	#Ack Rcv (Direct)	Data Gen.Time	Awake Time	Sleep Time	Residual Power(J)	Packet Loss(%)	Latency (sec)
11:07:002	0	7	0	0	0	0	8	52	0	0	8:13.	5:20.	2:28.	26993.16	0[%]	2:54
14:26:306	1	7	1	0	0	1	8	53	0	1	13:13.	5:21.	7:46.	26993.04	0[%]	1:13
20:26:427	2	7	2	0	0	2	8	54	0	2	18:13.	5:23.	12:25.	26992.9	0[%]	2:13
29:06:750	4	7	4	0	0	4	16	107	0	4	28:13.	9:59.	17:49.	26986.83	0[%]	53
34:26:330	6	7	5	0	0	5	23	162	0	6	33:13.	15:19.	17:49.	26979.95	0[%]	1:13
39:47:494	6	7	6	0	0	6	25	163	0	6	38:13.	17:19.	20:29.	26977.32	0[%]	1:34
43:46:967	7	7	7	0	0	7	25	164	0	7	43:13.	17:20.	25:47.	26977.2	0[%]	33

Figure 4: Administrative Values Table

Reception time	Generated time	Source Node ID	Next-hop Node ID	Status	Misc.
12:10:29:071	9:29.	20	47	Unstable Links	Packet Loss Ratio:20%
12:29:06:517	24:29.	20	13	Unstable Links	Packet Loss Ratio:11.5%
12:45:06:654	44:29.	20	13	Low Residual Battery Power	Residual Battery:26968.6[J]
12:26:25:914	25:18.	25	1	Unstable Links	Packet Loss Ratio:11.1%
12:30:26:334	30:18.	25	1	Unstable Links	Packet Loss Ratio:10%
12:35:06:376	34:14.	29	25	Unstable Links	Packet Loss Ratio:10%
12:08:25:700	8:5.	32	1	Long Time Without	Interval:51min35sec
12:14:26:909	12:50.	41	7	Unstable Links	Packet Loss Ratio:12.5%

Figure 5: Alert Table

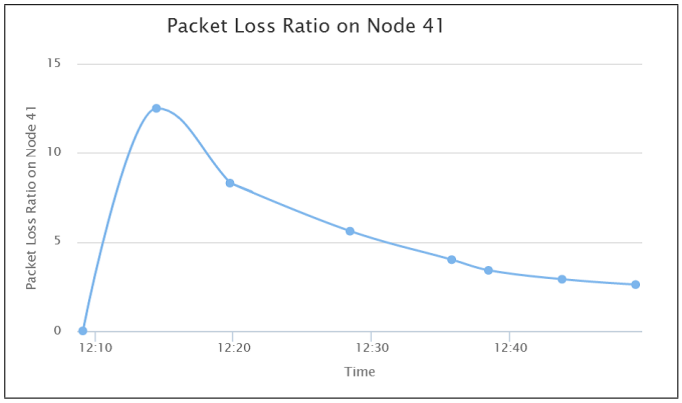

Figure 6: Line Graph View

sends periodical beacons in 40[sec] interval. We run the simulation in 60[min], but we intentionally invoke a node failure about the middle of the simulation time. Other parameters related to the simulation is shown in Table 1. As a result of simulation, we obtained a set of trace data set, i.e., the set of administrative values collected at the sink.

As for the parameters on the proposed system, we used the threshold values to generate alert messages in Table 2. These values are determined through previously performed test runs.

Evaluation process is in the following. We prepare the proposed system that loaded the above trace data set. We asked five subject, who are students who have studied both the Internet management and sonsor network protocols, to use the system for 20 minutes and also asked to list up the events that is important from the viewpoint of three management aspects, i.e., structure, failure, and performance management. After the above operations, we carefully examined the trace data set to find all the events that administrators should be aware of. We confirmed whether the subject can find all the events or not.

5.3 Results

By examining the events that the subjects found, we evaluate the practical efficacy of the proposed system. First of all, we say that the subject found all the events that we judged administrators should be aware of. This result shows that the proposed system works well and useful in managing WSNs.

Table 1: WSN Environment

Simulation Start Time	12:00
Simulation Time	60[min]
Field Size	200[m]×200[m]
Number of Nodes	50
Communication Range	Circle with Radius 50[m]
Beacon Interval	40[sec]
Sensing Interval	5[min]
Battery Capacity	27000[J]

Table 2: Threshold for Alert Table

Packet Loss Ratio	10%
Delivery Delay	60[sec]
Elapsed Time after Last Frame Reception	30[min]

In the following, we see the detailed description seen from the three aspects of network management, i.e., structure, failure, and performance management.

5.3.1 On Node Failure (Failure Management)

All the five subjects indicated the failure of node 32, which we intentionally did in the middle of the simulation time. From hearing from the subjects, this event was found by watching the delivery tree view. Figure. 7 shows the delivery tree at some time point where the node failure is clearly seen; since packets from node 32 has not been reached the sink for a long time, child nodes of node 32 no longer exist and the color of node 32 changes. Simultaneously, this event is displayed in the alert table. In addition, by watching administrative values table of node 32, all the subjects confirmed that only one sensed packet of the node 32 has arrived at the sink node during the evaluation test.

In this way, all the subjects actually found failure of node 32 in our evaluation test using the alert table, the delivery tree view, and the administrative values table. From the above, the system provides several mechanisms to help administrators find node failure. Therefore we confirmed that the failure management was well performed using our system.

5.3.2 On Path Transition (Structure Management)

All the five subjects explained how and why delivery tree changes as time passes. This is also possible using delivery tree view. When the subjects saw the delivery tree just after failure of node 32, which is shown in Fig. 8, they found that all nodes whose next-hop was node 32 changed their next-hop one after one. Also, the subjects found that, node 20 observes far larger number of control frames than usual just after node 32 fails. Note that the deployed MAC and routing protocol [6] transmits far larger number of control messages in face of topology changes to speed up the tree reconstruction. The subjects observed this behavior of nodes, which also supports that the root cause that changes next-hops is failure of node 32.

Also, two of the subjects confirmed that nodes around node 28 changed their next-hops to avoid node 28 in the similar

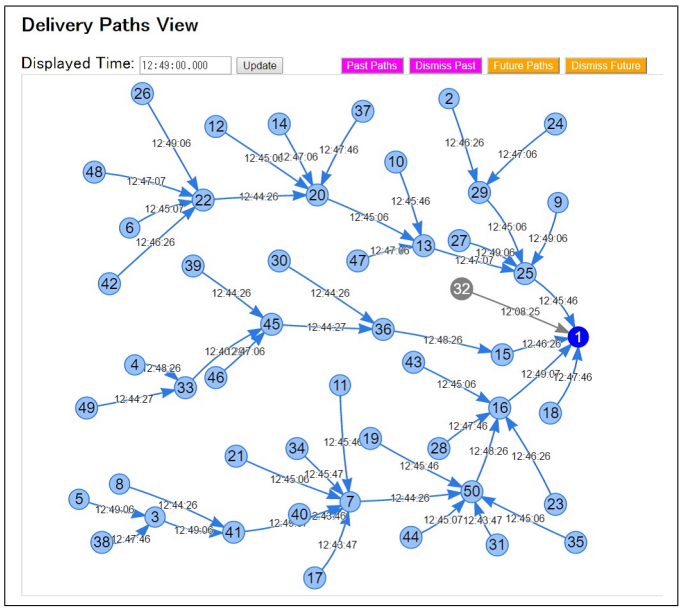

Figure 7: Delivery Tree View in Finding Node Failure

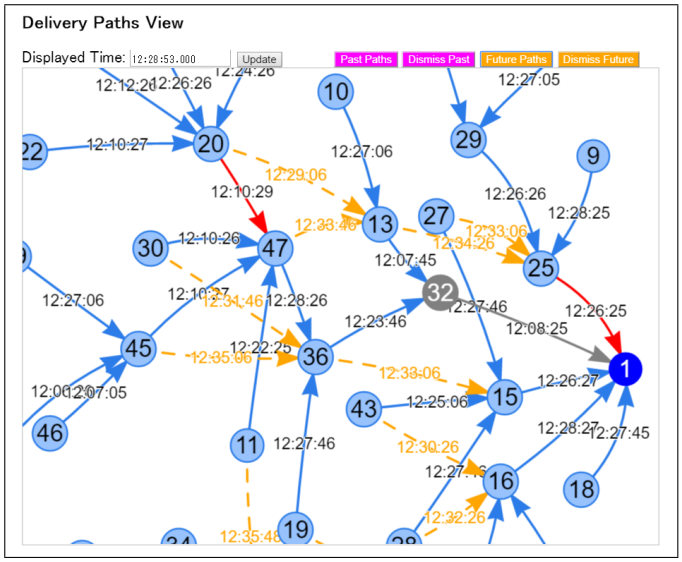

Figure 8: Delivery Tree View in Tree Transition

way to the case of node 32. In this case, however, node 28 did not fail. Instead, the subjects found that node 7, which is a child of node 28, has high packet loss ratio on the link to node 28. In this way, subjects could identify the cause of the delivery tree by means of referring the administrative values table. As above, we confirmed that the structure management was well performed.

5.3.3 On Performance Degradation (Performance Management)

As for the performance of the network, all the five subjects indicated that there are several nodes that had high frame loss ratio, which is found by the alert table and the delivery tree view. When the subjects examined the network state around node 20, several nodes adjacent to node 20 transmit control frames more frequently. So, the subjects naturally judges that

those control messages are the cause of the congestion and the frame loss.

Also, three subjects found that all children of the sink node had high packet loss ratio. It seems that the cause could be the failure of node 32 because node 32, which failed, was a child of the sink. However, the subjects found that the main cause was different from it. They found the packet loss ratio of node 18, which was also a child of the sink, was as high as 33% because the number of children of node 18, as well as its traffic, was large. They also found that, the number of children of node 18 decreases after that, as the children of node 18 change their next-hops. As a result, the collision among the children of the sink decreased, and the packet loss ratio improved accordingly. This indicates that the main cause was the concentration of traffic at node 18. As above, we confirmed that performance management was also well performed.

In summary, through the evaluation test, we found that the subjects performed all the tasks of failure, structure, and performance management. Consequently, we confirmed that the proposed system works well in a practical scenario of WSN management.

Additionally, note that our result implies that four basic components (i.e., delivery tree, administrative values table, line graphs, and alert table) are sufficient for a management tool of WSNs. Several visualization tools that have complicated views have been presented so far. However, from the viewpoint of functional design, the propose system can offer a new principle of WSN management tools.

6 DISCUSSION

In this section, we discuss two important issues over the proposed approach; one is (A) how the system can help administrators surely recover WSNs after detection of disorders, and the other is (B) on the overhead introduced by the administrative values attached to every packet.

First, we discuss (A), the function for failure recovery. Requirement to this system for managing WSNs is to make administrators surely notice disorders of networks quickly, and simultaneously to provide them the information that is essential in recovering the networks. We have various possible causes of disorders in WSNs such as node failure, communication failure due to severe interference or obstacles, and incorrect behavior of nodes due to buggy software. To enable administrators to recover those disorders, it is necessary for WSN management systems to provide the information that enables administrators to identify and specify the parts of the system that cause the disorders. However, because there are too many types of disorders that possibly occur, it is generally impossible to provide the information that guarantees to identify all cases and specifies the root causes of disorders. On the other side, in our evaluation scenario, it is shown that administrators could detect performance degradations due to node failure and congestion, and also that they could grasp the network state that the administrator should be aware of in order to understand what happens in the WSN. Although this evaluation results do not guarantee the ability of our system to deal with all kinds of disorders, we actually showed that the proposed system has ability to handle several typical cases of disorders that occur in WSNs.

Next, we explain about (B) how much the overhead of administrative values influences on communication performance. Since WSNs are generally discussed without thinking of the concept of management operations, packets sent to sink are usually supposed to include only the minimum information such as the value sensed at each sensor, meaning that the packet size is considered very small. However, to enable administrators to manage WSNs properly so that they can specify the cause of most of troubles, we need at least 16 administrative values as discussed in section 4.2. Therefore, we discuss in the following whether the overhead needed for the management affects the behavior of WSNs. First, as a typical example of the packet sizes of the 16 items listed in section 4.2, we assume that items (a), (b) spends 8 bytes, (m), (n) do 4 bytes, (p) does 8 bytes, and the others do 2 bytes. The total size of them is 54 bytes even when we require positional information obtained from GPS attached to sensor nodes. On the other side, in recent years, the typical transmission speed in WSNs is generally around 250 Kbps, as it is also proposed in several standards such as IEEE802.15.4. Consequently, the overhead calculated as $54 \times 8 = 432$ bits is approximately 0.17 % of the transmission speed. Although nodes near the sink needs to transfer a large number of packets, it is surely expected that the overhead would hardly increase congestion as well as worsen communication performance, because sensor nodes in most cases would have sufficient sleep time to save their battery power to work as members of the WSN. On the other hand, from the viewpoint of power consumption, we cannot deny the negative influence by forcing additional 54 bytes, which may increase packet size by several times as the original size, even though we require the overhead of beacons and control messages to compute forwarding paths with a routing protocol. However, we would again emphasize that we must accept the administrative values as overhead required for management operations in WSNs.

7 CONCLUSION

In this paper, we proposed a system to manage WSNs that deploy low-power MAC protocols. We designed the system from the viewpoint of the network management in the Internet, i.e., we aim at executing structure, failure, and performance management. We evaluated the proposed system using a simulation trace data set. As a result, all the events that should be noticed are listed up by the subject, while all three management aspects are well managed, showing that the proposed system possibly works effectively in the real WSN management operations.

To apply the system to the real environment instead of simulation is one of the important task for the future.

ACKNOWLEDGMENT

This work is supported by JSPS KAKENHI (15H02691).

REFERENCES

[1] IEEE802.15.4, http://www.ieee802.org/15/pub/TG4.html (referred November 21, 2015).

[2] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," In Proc. of SenSys'04, pp.95–107, (2004).

[3] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-MAC: A Short Preamble MAC Protocol for Duty-cycled Wireless Sensor Networks," In Proc. of SenSys'06, (2006).

[4] Y. Sum, O. Gurewits, and D. B. Johnson, "RI-MAC: A Receiver-initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Loads in Wireless Sensor Networks," In Proc. of SenSys'08, pp.1-14, (2008).

[5] P. Huang, C. Wang, and L. Xiao, "RC-MAC: A Receiver-Centric MAC Protocol for Event-Driven Wireless Sensor Networks," In Proc. of IWQoS'10, (2010).

[6] S. Kojima, T. Yoshihiro, "A Low Management Cost Wireless Sensor Network Based on Receiver Initiated MAC Protocols," Journal of Information Processing, Vol.57, No.2, pp.480-493, (2016) (In Japanese).

[7] X. Fafoutis, A.D. Mauro, M.D. Vithanage, and N. Dragoni, "Receiver-initiated medium access control protocols for wireless sensor networks," Computer Networks, Volume 76, Pages 5574, (2015).

[8] SNMP Standard, http://www.snmp.com/ (referred December 6, 2015).

[9] N.Ramanathan, K.Chang, R.Kapur, L.Girod, E.Kohler, and D.Estrin "Sympathy for the Sensor Network Debugger," In Proc. Sensys'05, (2005).

[10] K. Liu, M. Li, and Y. Liu, "Passive Diagnosis for Wireless Sensor Networks," IEEE/ACM Transactions on Networking, Vol.18, Issue 4, pp.1132–1144, (2010).

[11] B.Parbat, A.K.Dwivedi, and O.P.Vyas, "Data Visualization Tools for WSNs: A Glimpse," International Journal of Computer Applications, Vol.2, No.1, pp.14–20, (2010).

[12] K. Liu, Q. Ma, Xibin Zhao, and Yunhao Liu, "Self-Diagnosis for Large Scale Wireless Sensor Networks," In Proc. INFOCOM'11, (2011).

[13] M. Khan, H. Le, H. Ahmadi, T. Abdelzaher, and J. Han, "Dustminer: Troubleshooting Interactive Complexity Bugs in Sensor Networks," In Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems, pp.99-112, (2008).

[14] X. Li, Q. Ma, Z. Zhicao, K. Liu, and Y. Liu, "Enhancing Visibility of Network Performance in Large-scale Sensor Networks," In Proc. of 34th International Conference on Distributed Computing Systems (ICDCS'14), (2014).

[15] X. Miao, K. Liu, Y. He, D. Papadias, Q. Ma, and Y. Liu, "Agnostic Diagnosis: Discovering Silent Failures in Wireless Sensor Networks," IEEE Transactions on Wireless Communications, (2013).

[16] R.Jurdak, A.G. Ruzzelli, A. Barbirato, and S. Boivineau, "Octopus: Monitoring, Visualization and Control of Sensor Networks" Wireless Communications and Mobile Computing, Vol. 11, Issue 8, pp. 1073–1091, (2011).

[17] E. Karapistoli, P. Sarigiannidis, and A.A. Economides, "SRNET: A Real-time Cross-based Anomaly Detection and Visualization System for Wireless Sensor Networks," In Proc VisSec'13, (2013).

[18] Information-technology Promotion Agency, Japan (IPA), "Knowledge for Network Management I," Open Source Software Model Curriculum Version 1, https://jinzaiipedia.ipa.go.jp/wp-content/uploads/oss/basic_Guidance_12.pdf (referred December 5, 2015).

[19] vis.js, http://visjs.org/ (referred January 15, 2016).

[20] Highcharts, http://www.highcharts.com/ (referred January 15, 2016).

[21] The Apache Software Foundation, http://www.apache.org/ (referred December 10, 2015).

[22] Contiki, http://www.contiki-os.org/ (referred February 1, 2016).

(Received October 20, 2016)
(Revised February 3, 2017)

Yuki Urata received the B.E. degree from Wakayama University in 2016. He is currently a Master-course student in Wakayama University.

Takuya Yoshihiro received his B.E., M.I. and Ph.D. degrees from Kyoto University in 1998, 2000 and 2003, respectively. He was an assistant professor in Wakayama University from 2003 to 2009. He has been an associate professor in Wakayama University from 2009. He is currently interested in the graph theory, distributed algorithms, computer networks, medial applications, and bioinformatics, and so on. He is a member of IEEE, IEICE, and IPSJ.

Yutaka Kawahashi received his the B.S. degrees from Wakayama University, Japan in 1993, and received the M.S. degrees from Nara Institute Science and Technology, Japan in 1995, and received the Ph.D. degrees from Graduate School of Engineering, Osaka Prefecture University, Japan in 2013. He is an Assistant Professor at Center for Information Science, Wakayama University, Japan. His research interests include Internet architecture, network management and network security. He is a member of WIDE Project.