Performance improvement of periodic flows in multi-hop wireless sensor networks

Performance improvement of periodic flows in multi-hop wireless sensor networks

Nguyen Anh Huy

January 2019

Doctoral Thesis at Osaka Prefecture University

Performance improvement of periodic flows in multi-hop wireless sensor networks

Nguyen Anh Huy

January 2019

Doctoral Thesis at Osaka Prefecture University

Summary

In recent years, the demand for wireless networks treating various types of periodic flows increased, for example, wireless sensor networks (WSNs) for healthcare, smart meter networks, or structural health. Particularly, in healthcare networks, there are numerous periodic data flows, such as blood pressure, heart rate, and blood oxygenation level. In such wireless networks treating periodic flows, the following problems must be solved.

Firstly, periodic flows cause the inherent problem of continual packet collisions, which results in successive packet losses and decrease in communication quality. Specifically, if the packet generation timing on different source nodes overlaps, packet collisions among the different interfering flows occur continually, until the interfering sessions are terminated. Although the IEEE 802.11 distributed coordination function (DCF) fixes packet collisions, its random backoff to avoid subsequent packet collisions and retransmission reduces network effective bandwidths, which results in packet loss due to network congestion. Additionally, the workloads for relay nodes increase due to the retransmission and timer expiration processes. Therefore, another collision avoidance mechanism to deal with periodic flows is required.

Secondly, given a large number of sensor nodes placed in a large area, hidden node problem is the problem that occurs when a node (node A) is visible to a node (node B) but not to other node (node C) which is communicating with node B. When these nodes are in hidden node topology, if node C is transferring packet to node B, and node A also

start transferring packet to node B, a collision occurs. This collision will not happen if node A is in range of node C and thus knows that node C is transferring its packet to node B. Hidden node problem also becomes serious in addition to general contention between data flows. Moreover, once periodic packet transmission phases are synchronized among different periodic data flows, they will contend continually.

Many existing protocols that schedule the timing of sending packets are based on time division multiple access (TDMA). However, TDMA is not widely spread for the following reasons. First, the installation cost of nodes is expensive. Second, TDMA is not suitable for dynamically changing network environments and TDMA-based systems need complicated controls, such as time synchronization.

This thesis attempts to propose methods to improve the performance of periodic flows in wireless sensor networks (WSNs). Furthermore, we try to avoid the constraints of the related work such as time synchronization and high installation cost.

As for the detailed content, this thesis is organized as follows:

In Chapter 1, we show the research overview of this thesis. We also describe the problems and some related solutions. In particular, this thesis will focus on two problems, the inherent problem of continual packet collisions and the compounded effect of the hidden node and the continuous collision problems.

In Chapter 2 and Chapter 3, we tackle the first challenge of this thesis which is the problem of continual packet collisions by shifting the packet generation timing. In Chapter 2, we propose a simple method to choose the shift-time. The simulation in single-hop network environment shows the positive results.

In Chapter 3, we propose a new formula for predicting whether two heterogeneous periodical flows from different source nodes have overlapping packet transfer durations. From this formula, we propose transfer scheduling methods that shift the packet generation

phase (timing) to avoid future collisions. These methods adopt naive random-access control, like DCF, for the MAC layer process. In addition, source nodes do not require significant computational power, because only the sink intensively schedules the timing and informs to the corresponding source. Therefore, compared to existing methods in which each source node completely schedules the timing of creating packets based on TDMA, our methods require less complexity, and computational power. Finally, we demonstrate the effectiveness of our methods through simulation in both single and multi-hop environments.

As the next challenge, in Chapter 4, this thesis tackles a compounded negative effect of the hidden node problem and a continuous collision problem among periodic data packet flows in WSNs. This is not a simple and well-studied solution for just the hidden node problem but the compounded problem. With the rapid increase in IoT (Internet of Things) applications, more sensor devices, generating periodic data flows whose packets are transmitted at regular intervals, are being incorporated into WSNs. However, packet collision caused by the hidden node problem becomes serious particularly in large-scale multi-hop WSNs. Moreover, focusing on periodic data flows, continuous packet collisions among periodic data flows are caused once periodic packet transmission phases are synchronized. To address this challenge, we propose a new MAC layer mechanism. The proposed method predicts a future risky duration during which collision can be caused by hidden nodes by taking into account periodic characteristics of data packet generation. In the risky duration, each sensor node stops the transmission of its data packets in order to avoid collisions. To the best of our knowledge, this is the first work that considers the compounded effect of hidden nodes and continuous collisions among periodic data flows. Other advantages of the proposed method include that any new control packets are not required and it can be implemented in widely-diffused IEEE 802.11 and IEEE 802.15.4 devices.

Finally, in Chapter 5, we conclude the thesis and discuss about the future work.

Acknowledgments

Firstly, I would like to express my sincere gratitude to my advisor Prof. Hideki Tode for the continuous kindly support of not only my Ph.D study and research but also my life in Japan. His guidance helped me in all the time of Ph.D course from the admission to the graduate.

Besides my advisor, I would like to thank Associate Prof. Yosuke Tanigawa for his insightful comments and encouragement during my Ph.D course. His support is very important because he was the one who read my draft papers and helps me fixing a lot of writing mistakes.

My sincere thanks also goes to Prof. Koichi Kise and Prof. Yushi Uno. Although they were busy at that time, the professors had accepted to proofread this thesis and evaluate it.

I thank my fellow lab mates for the help and for all the fun we have had in the last three years. In particular, I am grateful to my friend Le Hong Nam, who helped me and my family a lot in the paper work and the daily life at the first time we came to Japan.

Last but not the least, I would like to thank my family: my parents and my wife for supporting me spiritually throughout writing this thesis and my life in general.

Contents

Chapter 1

Introduction

1.1 Overview

1.1.1 Inherent problem of continual packet collisions

In recent years, the demand for wireless networks treating various types of periodic flows increased, for example, wireless sensor networks (WSNs) for healthcare [1], smart meter networks [2–4], or structural health [5]. Particularly, in healthcare networks, there are numerous periodic data flows, such as blood pressure, heart rate, and blood oxygenation level. In such wireless networks treating periodic flows, the following problems must be solved.

The first problem is that periodic flows cause the inherent problem of continual packet collisions, which results in successive packet losses and decrease in communication quality. Specifically, if the packet generation timing on different source nodes overlaps, packet collisions among the different interfering flows occur continually, until the interfering sessions are terminated. Although the IEEE 802.11 distributed coordination function (DCF) fixes packet collisions, its random backoff to avoid subsequent packet collisions and retransmission

reduces network effective bandwidths, which results in packet loss due to network congestion. Additionally, the workloads for relay nodes increase due to the retransmission and timer expiration processes. Therefore, another collision avoidance mechanism to deal with periodic flows is required.

To tackle this challenge, we focus on shifting the timings of packet creation to only certain source nodes in order to prevent packet collisions by adaptively equalizing creation phase differences among periodic flows.

The proposed scheduling method has the following design particularities and advantages. First, for high feasibility and low network and node installation costs, the method assumes a random-access based protocol at the MAC layer. One representative random-access method is the IEEE 802.11 standard, which has been widely diffused and employed for most current portable devices. Second, for low complexity and computational power at the sensor nodes, each source node simply changes its packet generation timing based on an instruction from its sink, and only the sink calculates the appropriate packet generation timing of all source nodes. The overhead for the sink to notify packet generation timing to each source node is maintained small because only one control packet is transferred per source node. Finally, the proposed method is implemented at the application layer between the sink and each source node. Therefore, no modification of the MAC layer is required. Moreover, because of the above-mentioned flexibility, such as the random access-based method and no modification to the MAC layer, the proposed methodology is applicable to various types of wireless networks from the viewpoint of the number of hops to a sink, routing tree topology, etc.

Although setting the packet generation timing of all the source nodes in advance would be ideal for avoiding an overlapping packet generation, it is impossible to pre-establish the scheduled traffic for a dynamic topology. In such environments, there are cases in which nodes join or leave the network during operation (e.g., wearing a new biological sensor or

terminating usage of a smart meter). If nodes with different data generation cycles join or leave the network, it is necessary to recalculate the timing.

Many existing protocols that schedule the timing of sending packets are based on time division multiple access (TDMA). However, TDMA is not widely spread for the following reasons. First, the installation cost of nodes is expensive. Second, TDMA is not suitable for dynamically changing network environments and TDMA-based systems need complicated controls, such as time synchronization.

1.1.2 Compounded negative effect of hidden node and continuous collision problems

With the rapid increase in IoT (Internet of Things) applications, for increasing the sensing coverage while reducing the power consumption of sensor nodes, many WSNs use multi-hop connections[6][7][8][9][10][11][12]. However, as shown in Fig.1.1, these applications treating various periodic flows face a common challenge of solving packet collisions among periodic data flows whose data packets are generated at regular intervals.

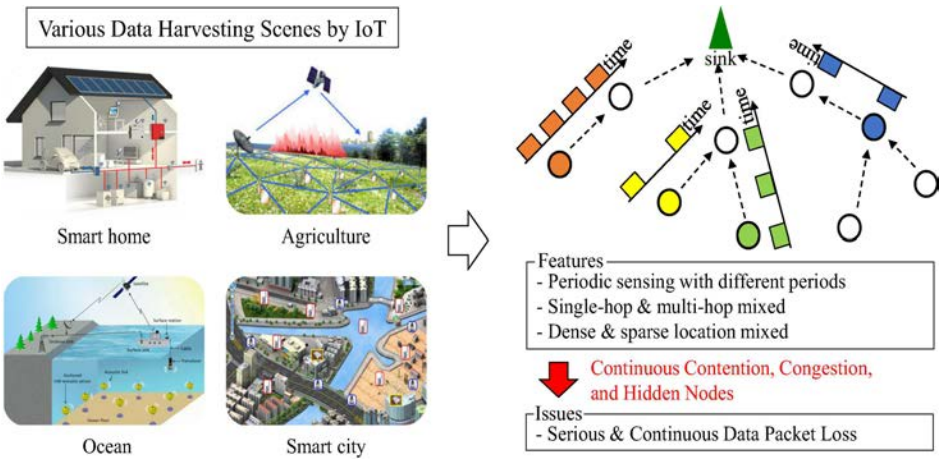


Figure 1.1: Various data harvesting scenes by IoT and their technical issues.

Given a large number of sensor nodes placed in a large area, hidden node problem is the problem that occurs when a node (node A) is visible to a node (node B) but not to other nodes communicating with node B. Hidden node problem becomes serious in addition to general contention between data flows. Moreover, once periodic packet transmission phases are synchronized among different periodic data flows, they will contend continually.

Therefore, the second problem that we tackle in this thesis is the compounded negative effect of the hidden node problem and the continuous collision problem among periodic data flows in multi-hop WSNs. These problems, when compounded, become catastrophic for the network, which is not just the well-studied hidden node problem but the compounded problems. To realize this objective, we propose a new MAC layer mechanism. The proposed method predicts a future risky duration during which collision can be caused by hidden nodes by taking into account periodic characteristics of data packet generation. In the risky duration, each sensor node stops the transmission of its data packets in order to avoid collisions.

Other advantages of our proposed method include that any new control packets are not required and random access-based MAC layer protocol is assumed. Thus, it can be implemented in widely-diffused IEEE 802.11 and IEEE 802.15.4 devices.

1.2 Contribution of thesis

The contributions of this thesis can be summarized as follows.

1.2.1 Contribution in solving inherent problem of continual packet collisions

- We mathematically analyze the packet contentions occurred by the approach that shifts the timings of packet creation at the application layer. This is novel because most traditional methods focus on the MAC layer and few other related studies on this application do not measure how serious contention is in their systems.

- We propose and prove a formula used to derive the minimum time differences of packet creation between two periodic flows. This formula is important because of the following two reasons. First, it can detect interferences between heterogeneous periodic flows. Second, it shows unique characteristics of the contention between periodic flows.

- Based on the formula, we propose a new scheduling method at the application layer, which rapidly adjusts packet creation timing to reduce contentions and packet collisions. In the proposed method, no modification to the IEEE 802.11 standard is required.

- The effectiveness of the proposed method is shown in a realistic application using a packet-level simulator. The simulation system includes 100-node single- and multi-hop networks with heterogeneous packet creation intervals.

1.2.2 Contribution in solving compounded negative effect of hidden node and continuous collision problem

- To the best of our knowledge, this thesis firstly considers the compounded effect of hidden nodes and continuous collisions among periodic data flows. The compounded effect not only greatly increases packet loss rate, but also decreases the fairness among

data flows. This has been never considered in existing works.

- We propose a novel MAC layer control to deal with the combination of periodic flow and hidden node problem. The simulation results show that the proposed method significantly improves both packet loss rate and the fairness of the system.
- The proposed method's effectiveness is demonstrated in a complex system using a packet-level simulator. We use a multi-hop wireless network composed of up to 100 nodes with heterogeneous packet creation intervals for the simulation.

1.3 Organization of thesis

Chapter 2 and Chapter 3, which correspond to [13] and [14] respectively, present two proposed methods to deal the inherent problem of continual packet collisions. Chapter 4, which corresponds to [15], describes the proposed method for solving the compounded negative effect of the hidden node and the continuous collision problem. Finally, Chapter 5 concludes the findings of this thesis.

Chapter 2

Binary Division Method: a simple approach

In this Chapter, we introduce the first attempt to deal with the inherent problem of continual packet collisions. This method is simple for implementing and shows the good results in comparison with the related work. The content of this chapter corresponds with the work that published in [13].

2.1 Target system and related work

2.1.1 Target system

Our target system is shown in Fig. 2.1, where each sensor node sends data packets to its sink periodically by single- or multi-hop communication. Each sensor node has a different packet generation period, which depends on the requirements of the practical applications. This model could be applied to various WSNs (e.g., nursing homes and hospitals).

If a collision occurs between two packets, it will continue because of the periodic

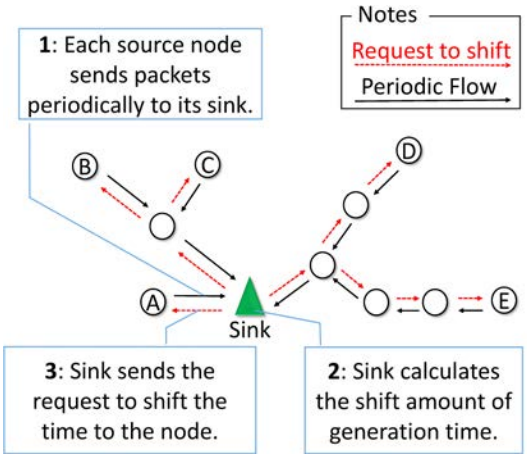


Figure 2.1: Target system overview.

characteristic. As shown in Fig. 2.2, our solution is to shift the packet generation (sending) time to avoid future collisions. The purpose of this approach is determining the best shift amount of time for each source node, which is complex, particularly in realistic environments where source nodes dynamically arrive at and leave the network.

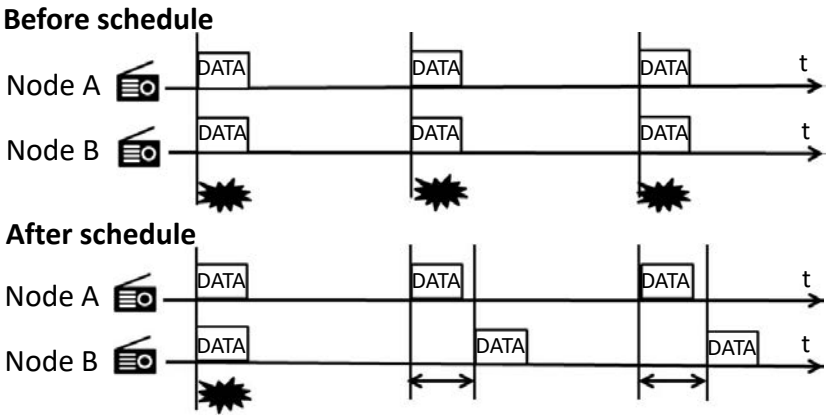


Figure 2.2: Solution to shift packet generation time.

In this system, the sink is a control center. By using the information on assumed packet generation and the periodic interval at each source node, the sink calculates the shift amount of packet generation time and sends a request to shift the time to the sensor node.

2.1.2 Related works

2.1.2.1 Existing methods to prevent packet collisions

Numerous scheduling methods have been proposed to improve the performance of periodic systems. These methods deal with various problems (e.g., reducing energy consumption[16–19], improving data quality [20], satisfying time constraints [21][22], and reducing packet collisions[23–36]). In this thesis, we focus on avoiding packet collisions among different periodic flows.

However, most existing methods for collision avoidance do not focus on periodic flows [23–29]. Some protocols focus only on a single type of periodic flow systems [30][31], which does not suit most real applications, while others use TDMA-based methods and require global synchronization [32–35], which is not feasible in a realistic environment. Further, to the best of our knowledge, the packet collision problem (among different types of periodic flows) has not been fundamentally solved.

There are several MAC layer protocols for scheduling periodic traffic. For instance, RMAC [16] is a representative method, in which each node independently schedules the timing of packet transmission. When a node does not communicate, it goes to sleep mode to reduce energy consumption. However, there is no protocol that dynamically schedules transmission timing so that it avoids collisions with forthcoming periodic packets at application level control without any modification of the MAC layer protocol.

Concerning traffic scheduling, there are several protocols based on TDMA. For example, S-MAC [17] introduces a fixed duty cycle that periodically puts nodes into sleep mode. However, this increases latency in heavy-traffic environments. On the other hand, Z-MAC [18] dynamically switches between carrier sense multiple access (CSMA) and TDMA depending on traffic. Under this scheme, the scheduling overhead is incurred mostly at deployment time. Similar to TDMA, each node is statically assigned a time slot, but unlike

TDMA, a node can transmit both its time slot and slots assigned to other nodes, where the owners of the current time slot always have higher priority over non-owners in accessing the channel.

These protocols are similar to our proposed method in terms of scheduling the timing of sending packets. However, such TDMA based protocols are not suitable for a dynamically changing network environment [35] because the time slot assigned to each node is fixed and, hence, protocols based on TDMA are not flexible. Additionally, TDMA-based systems tend to be more complicated and expensive than random access ones.

Another TDMA based protocol is introduced in [32], which solves the contention problem of periodic flows by scheduling packet transmission timings and changing the periods of nodes. However, it requires that nodes' periods are known before scheduling and cannot adapt to dynamically changing network environments.

A decentralized approach that transfers calculation complexity to sensor nodes was thus proposed [30]. However, our method has the advantage of a low computational power requirement for sensor nodes because sensor nodes in WSNs have limited hardware and calculation power. Moreover, the above related study[30] assumes all nodes transfer with the same interval. By contrast, our proposal deals with heterogeneous periodic flows, which is a more complex problem.

Other approaches [31][36] also concern avoiding collisions. However, they deal with bit collision, which is considered as a small transfer duration. By contrast, our method deals with packet collision, in which transfer duration is significant. Hence, they are two different problems.

Another popular TDMA based method is the STDMA [33], where each source node chooses the time slot it requests to use and broadcasts it to other nodes. However, it requires synchronization among all nodes. Additionally, this method requires sending control

packets continuously to reserve a slot, which leads to reducing the effective bandwidth and causing overhead problems.

Table 2.1: Comparison Table of Periodic Collision Avoidance Methods

Features	BDM [13]	CSMR [14]	DSM [37]	DTM [30]	EDF [34]	C-F [32]	APIS [31]	RDMA [36]
Multi-hop	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Heterogeneous period	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
Synchronization	No	No	No	No	Yes	Yes	Yes	Yes
Adaptation	No	Fast	Slow	No	No	No	No	No
Bit/Packet collision	Packet	Packet	Packet	Packet	Packet	Packet	Bit	Bit

As described in Table 2.1, there exist various methods for avoiding periodic collisions. Some methods deal with bit collisions in nano-networks which are different from our target system. Among packet collisions researches, several TDMA based methods are proposed. The existing methods based on TDMA require a significant amount of expansion in the MAC layer and require precise time synchronization. These restrictions make existing methods unsuitable for WSNs because the sensors are small and cannot provide high computational power. Furthermore, when the number of nodes is large, the synchronization request is impractical. By contrast, our proposed method does not need any expansion in the MAC layer including precise time synchronization, and the computational load is concentrated on the sink. Therefore, the method has high feasibility and a low cost.

2.1.2.2 Delta Shifting Method

Form the related work, we choose delta shifting method (DSM) [37] to compare with our proposed methods because DSM meets the requirements of our design policy. Specifically, DSM could apply to multi-hop model with heterogeneous period and does not require synchronization.

In the DSM, even in the situations in which the packet creation interval is unknown, packet collision events are recorded at each relay node and information is gathered at the sink. Subsequently, the sink detects the traffic that experiences most collisions and instructs the corresponding source node to shift the timing of packet creation.

This method consists of three steps. The first step is to provide the information on packet collisions between the sink and each source node. The second step is to decide which node's packet creation timing should be shifted. Finally, the third step is to notify the target source node to shift packet creation timing at the node by sending a shift-request packet.

This method is effective, particularly when packet creation intervals are unknown. However, the method requires a significant amount of time to significantly shift packet creation timing because only a small constant time, delta, can be shifted each time a control packet is sent. Furthermore, this method needs to shift nodes continuously. Further, we cannot predict whether the system status will improve after a shift-request and how long it will take to improve the system significantly. By contrast, the proposed method improves system performance immediately by finding the most suitable shift amount.

2.2 Proposed method

In this section, we describe the proposed method, Binary Division Method (BDM). The purpose is to find the suitable shift time of packet generation phase for each source node. Source nodes shift their packet transmission timings. In contrast to TDMA-based methods, the transmission of a source node is independent of the other node's one. Thus, each source node does not need to synchronize with other nodes, which is one of most attractive features. The proposed method minimizes the number of contentions in general. The contentions, which cannot be avoided in some random cases (e.g. control packets), are resolved by a MAC layer protocol like DCF.

Moreover, the sink performs all the extra calculation and storage. Each source node is required to shift its scheduled transfer timing only once after the first packet, which reduces hardware requirements and energy consumption.

Let T_i be the packet generation interval of the i^{th} node. Assume that we know the value of d, which is a common divisor of $\left(T_1, T_2, \ldots, T_{|N|}\right)$. In BDM, for scheduling, the sink manages data structure representing virtual time slots as show in Fig. 2.3. The x-axis is the time point of slots, and y-axis shows the levels of slots. We slip the time into slots p, $p < d$, and put the slots into levels l. The first and second level ($l = 0$ and $l = 1$) have one slot: 0 and $d/2$ respectively. The third level ($l = 2$) have two slots: $d/4$ and $3d/4$. The fourth level ($l = 3$) has four slots, and so on. We put a node's schedule into time slots from level 1. After filling all time slots in a level, we fill the next higher level. (i.e., $node_1$ into slot 0, $node_2$ into slot $d/2$, $node_3$ into slot $d/4$, $node_4$ into slot $3d/4$, and so on).

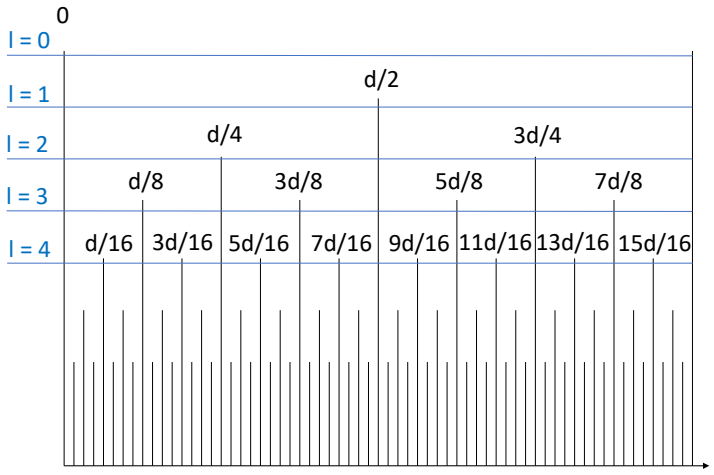


Figure 2.3: Virtual time slot structure managed by sink in BDM.

The formulas with which calculate the l and p of $node_i$ are

$$l = \texttt{ceil}\left(\log_2\left(i\right)\right) \tag{2.1}$$

$$p = \frac{\left(2\left(i - 2^{l-1}\right) - 1\right)}{2^l} \tag{2.2}$$

The shift time (s_i) of $node_i$ is determined by Algorithm 2.1.

Algorithm 2.1 Algorithm of finding s_i in BDM method

```
1: procedure FIND_SI(i, d, t_0i)
2:     l ← ceil(log_2(i))
3:     p = (2(i − 2^{l−1}) − 1) · d/2^l
4:     if (t_0i mod d) < p then
         /*Packet comes before time slot p*/
5:         s_i = p − (t_0i mod d)
6:     else
         /*Packet comes after time slot p*/
7:         s_i = (t_0i mod d) − p
8:     end if
9:     return s_i /*The suitable shift time*/
10: end procedure
```

We can prove that the δ_{min} of any two nodes will be greater than or equal to $\frac{d}{2^{\texttt{ceil}(\texttt{log}_2|N|)}}$. Moreover, if

$$\frac{d}{2^{\texttt{ceil}(\texttt{log}_2|N|)}} \geq C, \tag{2.3}$$

then any two nodes will never use the channel at the same time and they are not in contention.

If we do not know the interval of all nodes, we can assume that $d = 1$ (time unit of T_i). However, the smaller d we choose, the worse result we get. The best choice is $d = \texttt{gcd}\left(T_1, T_2, \ldots, T_{|N|}\right)$.

BDM uses very little memory (only storing the number of previously coming nodes) and calculation complexity. However, it depends on choosing d, and if $\frac{d}{2^{\texttt{ceil}(\texttt{log}_2|N|)}} < C$, then

contention and collision will occur frequently.

2.3 Performance evaluation

In this section, we describe simulated results that compare our proposed method BDM with the general case which is widely used in the current wireless network and another shift time method DSM.

2.3.1 Simulation environment

We used QualNet v5.2 [38] for our simulation. The parameter settings are described in Table 2.2.

Table 2.2: Details of Simulation Parameters

Name of parameter	Value
Data packet size	128 Bytes
Physical layer	802.11b
Data rate	2 Mbps
MAC layer	802.11
Retransmition limit	1
RTS/CTS	N/A
Routing	Static routing

We measure packet loss rate in a single-hop transmission environment to precisely evaluate the collision prevention of the proposed method without any external factors. Although some WSNs use a multi-hop model, single-hop networks, with more popular and cheap equipment, are also used for body area networks to obtain life-log or health information, Wi-Fi-based sensing systems in rooms for elderly care in apartments or hospitals, wide area sensor networks based on IEEE 802.11ah, and so on.

In the simulation environment, all nodes are allocated at random in the circumference of the propagation range from the sink. This topology is similar to the topology in applications such as healthcare equipments and smart homes. The packet generation interval of the nodes is chosen from $\{$100 ms, 200 ms, 300 ms, or 400 ms$\}$. The first, second, third, and fourth nodes' intervals are 100 ms, 200 ms, 300 ms, and 400 ms, respectively. Then, the intervals repeat (i.e., the fifth and sixth are 100 ms and 200 ms, respectively). All the nodes begin sending a data packet at a random time between 0 ms and 5000 ms.

2.3.2 Packet loss rate comparison

This section details the comparison among BDM, DSM and DCF in packet loss rate. Because DSM needs time to improve the system, we get simulated data at 50 s, 900 s, and 1800 s of simulation time.

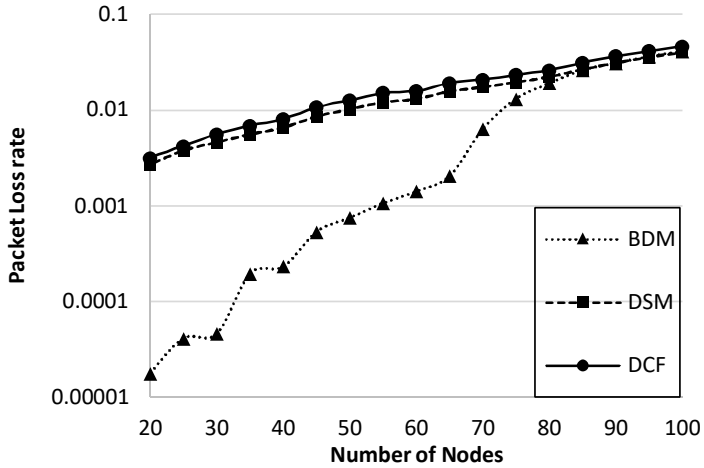


Figure 2.4: Packet loss rate by the number of nodes at 50 s simulation time.

The results in Figs. 2.4, 2.5, and 2.6 show that the packet loss rates of BDM and DCF do not change much over time. BDM is the best when the number of nodes is small. However, the packet loss rate of BDM increases rapidly after the number of nodes exceeds 30 and 65. Eq. (2.3) has a coefficient, of the power of two, $\texttt{ceil}\,(\log_2|N|)$, so that BDM's

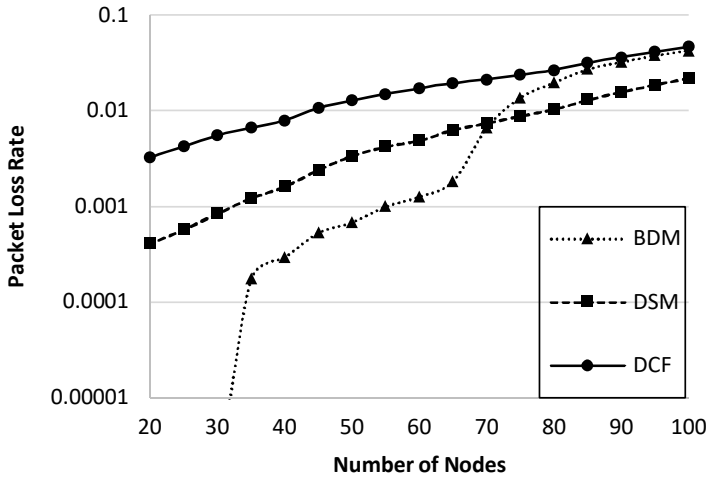


Figure 2.5: Packet loss rate by the number of nodes at 900 s simulation time.

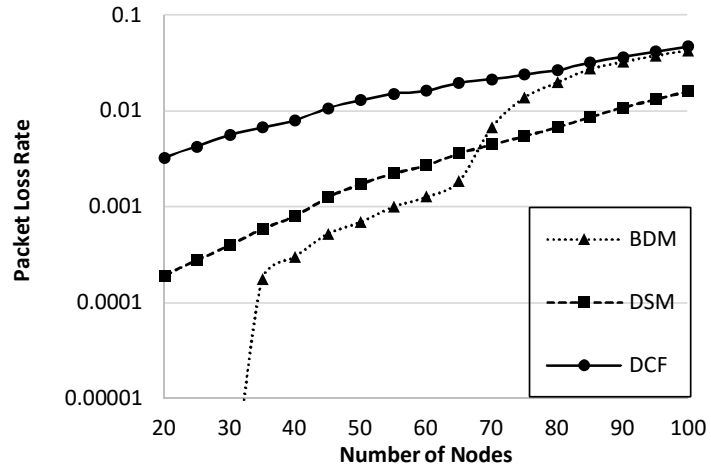


Figure 2.6: Packet loss rate by the number of nodes at 1800 s simulation time.

quality is greatly reduced when the number of nodes is greater than a power of two (e.g. 2, 4, 8, 16, 32, or 64). Therefore, when the number of nodes is too large, BDM does not improve packet loss rate.

On the other hand, DSM shows a poor result just after each node starts transmitting packets and the performance improves over time. DSM shows great improvement from 50 s to 900 s but little improvement from 900 s to 1800 s of simulation time. Therefore, DSM will take a very long time to improve the system. Moreover, as the system changes with more coming and leaving nodes, DSM will need more time to adapt.

2.3.3 End-to-end delay comparison

This subsection details the comparison among BDM, DSM and DCF in packet loss rate. Because the results in Sec. 2.3.2 show that DSM almost saturates from 900 s, we get the end-to-end data at 900 s of simulation time.

The result in Fig. 2.7 shows that all the methods' end-to-end delays increase when the number of nodes increases. BDM reduces the contention duration and thus reduces the waiting-for-transferring duration. Therefor, BDM shows the shortest end-to-end delay. We could conclude that our proposed method, BDM, not only significantly reduces the packet loss rate but also has a good effect on the end-to-end delay.

2.4 Conclusion

In this Chapter, to solve an inherent problem of repeated packet collisions in WSNs treating periodic traffic, we proposed to schedule packet creation timing at each source node based on a formula to address the contention problem. Through simulations, we demonstrated the effectiveness of BDM. In the next Chapter, we apply BDM in multi-hop model and also

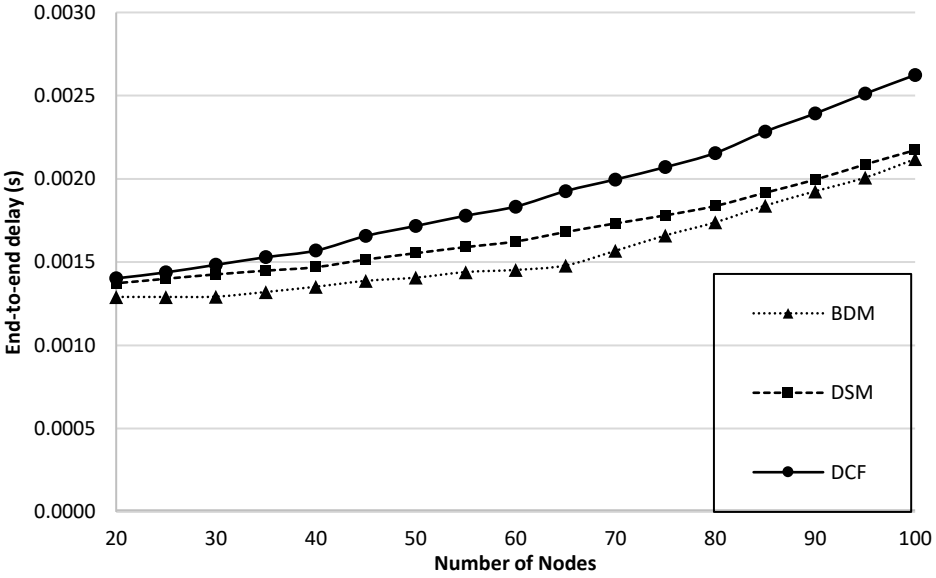


Figure 2.7: End-to-end delay by the number of nodes at 900 s simulation time.

propose a more effective method to solve the inherent problem of repeated packet collisions.

Chapter 3

Contention Score Method: a mathematics-based approach

In this Chapter, we present a more effective method to deal with the inherent problem of continual packet collision. The content of this Chapter corresponds with the work that published in [14].

3.1 Introduction

In the last chapter, we try to solve the inherent problem of continual packet collisions. However, the previous proposed method, BDM, has some disadvantages.

Firstly, BDM shows the worse performance in density network environment. Specifically, every time the number of source nodes surpasses the power of two values, the performance is drastically reduced. This is a serious problem because our target system could has a very large number of source nodes.

Second, BDM does not consider the multi-hop environment characteristics. Multi-hop topology is more complicated and has more random factors. For example, in multi-hop model,

because the packet is relayed through many relay-nodes, there are random congestions in the middle. Next, each time the packet is relayed, a random back-off time is added.

Third, BDM is only good for short-run applications. For long-run ones, the proposed method should have the ability to adapt to the change of network status.

Therefore, we need to investigate the inherent problem of continual packet collisions more carefully and propose a new method to improve the disadvantages of previous proposal.

3.2 Collision reduction analysis of proposal

This section presents the analysis of collisions among different periodic packets and identifies how to predict collisions.

3.2.1 Definitions

In this subsection, we show the definition of notations because we want to describe the periodic packet transmissions by a mathematical model. The notations used in this Chapter are listed in Table 3.1.

The set of nodes in a network is denoted by $N = \{node_i\}$. The number of elements of N is $|N|$. The i^{th} node ($node_i$) is characterized by (t_{0i}, s_i, T_i, C_i), where each property is an integer value. t_{0i} is the time point, at which the first data packet from $node_i$ reaches the sink. s_i is the shift amount of packet generation time at the node. T_i is the data packet generation interval at the node. C_i is the amount of time for which the node uses the channel to send a data packet. For simplicity, we assume all nodes have the same value $C = C_0$ in single-hop communication. The C_0 is the average duration for transferring a data packet from one node to its neighbor. In multi-hop environments,

$$C = n_h \cdot C_0, \tag{3.1}$$

Table 3.1: Notations

$\texttt{gcd}\,(a, b)$	greatest common divisor of a and b
$\texttt{lcm}\,(a, b)$	least common multiple of a and b
$\texttt{frac}\,(x)$	fraction component of x (e.g., $\texttt{frac}\,(1.234) = 0.234$)
$\texttt{int}\,(x)$	integer component of x (e.g., $\texttt{int}\,(1.234) = 1$)
$N = \{node_i\}$	set of nodes in a network
$\lvert N \rvert$	number of elements of N
t_{0i}	time point at which the first data packet reached the sink from $node_i$
s_i	shift amount of packet generation time at $node_i$
T_i	data packet generation interval of $node_i$
C_i	amount of time for which $node_i$ uses the channel for sending a data packet to the sink
C_0	amount of time for which a node uses the channel for sending a data packet to its neighbor
δ	difference in sending time of two packets

where n_h is the number of hops from the node to the sink. The difference in the sending time of two packets is δ.

3.2.2 Negative effects of contention

In a DCF-based system, a contention is the situation that more than one nodes want to use the channel to transfer their packets at the same time. The contention could become collision if there are more than two nodes. The backoff time is an random waiting duration after channel becomes idle and before transferring packet. The backoff mechanism helps reduce collision probability, but cannot completely avoid it. Collision probability increases when many source nodes contend the channel at the same time.

In multi-hop communication, the hidden node problem will prevent source nodes from correctly sensing the channel status. Consequently, the contention among source nodes will lead to a serious packet collision problem.

Another difference in multi-hop communication compared to single-hop is the duration

of transferring a packet from source nodes to the sink. These durations are nearly the same for single-hop environments, but vary in multi-hop communication. Hence, estimating contention time in multi-hop communication is more difficult.

3.2.3 Minimum sending time difference between packets

Here, we give the formula used to calculate the minimum sending time difference between two data packets (δ_{min}) generated from two nodes. Our method is mainly based on this formula.

The time at which $node_i$ sends its m^{th} packet is calculated as $t_{m,i} = t_{0i} + s_i + m \cdot T_i$.

We consider the m^{th} packet of $node_i$ (packet A) and the n^{th} packet of $node_j$ (packet B). Without loss of generality, we assume packet B will be sent before packet A ($t_{m,i} > t_{n,j}$). The sending time difference between these two packets is

$$\delta_{i,j} = t_{m,i} - t_{n,j}. \tag{3.2}$$

By substituting $t_{m,i}$ and $t_{n,j}$ into (3.2), we obtain $\delta_{i,j} = \{(t_{0i} + s_i) - (t_{0j} + s_j)\} + (m \cdot T_i - n \cdot T_j)$.

We set $\Delta t = (t_{0i} + s_i) - (t_{0j} + s_j)$.

Then,

$$\delta_{i,j} = \Delta t + (m \cdot T_i - n \cdot T_j)\,. \tag{3.3}$$

The minimum $\delta_{i,j}$ of $node_i$ and $node_j$, that is, $\delta_{i,jmin}$, can be calculated by

$$\delta_{i,jmin} = d \cdot \mathtt{frac}\left(\frac{\Delta t}{d}\right), \tag{3.4}$$

where $d = \gcd(T_i, T_j)$. A full proof is presented in the next section.

Equation (3.4) shows $\delta_{i,jmin}$ depends on Δt and when Δt changes, $\delta_{i,jmin}$ also changes with period d.

If packet B is going to be sent after packet A ($t_{m,i} < t_{n,j}$), similarly, we have

$$\delta_{j,imin} = d \cdot \mathtt{frac}\left(\frac{-\Delta t}{d}\right). \tag{3.5}$$

There are some important characteristics of the contention between two periodic flows as follows:

1. Regarding the choice of shift amounts s_i and s_j, we always have

$$0 \leq \delta_{i,jmin} \leq d. \tag{3.6}$$

2. The common interval of two periodic flows is

$$T = \mathtt{lcm}(T_i, T_j). \tag{3.7}$$

3. Two periodic source nodes will not contend if we can choose suitable s_i and s_j, so that $\delta_{i,jmin} \geq C_j$ and $\delta_{j,imin} \geq C_i$.

4. The total contention duration for a common interval is

$$t_c = t_{ci} + t_{cj}, \tag{3.8}$$

where

$$t_{ci} = \begin{cases} 0 & \text{when } \delta_{i,jmin} \geq C_j \\ C_j - \delta_{i,jmin} & \text{when } \delta_{i,jmin} < C_j \end{cases}$$

$$t_{cj} = \begin{cases} 0 & \text{when } \delta_{j,imin} \geq C_i \\ C_i - \delta_{j,imin} & \text{when } \delta_{j,imin} < C_i. \end{cases}$$

3.2.4 Proof of minimum time difference equation

This section presents the full proof of the formula used to calculate the minimum sending time difference between the packets of $node_i$ and $node_j$ ($\delta_{i,jmin}$).

We need to prove

$$\delta_{i,jmin} = d \cdot \texttt{frac}\left(\frac{\Delta t}{d}\right).$$

In which $\Delta t = (t_{0i} + s_i) - (t_{0j} + s_j)$,

and $d = \texttt{gcd}\,(T_i, T_j)$.

We already have (3.3)

$$\delta_{i,j} = \Delta t + (m \cdot T_i - n \cdot T_j).$$

Because $d = \texttt{gcd}\,(T_i, T_j)$, we can set $T_i = d \cdot T_i'$ and $T_j = d \cdot T_j'$, where $\texttt{gcd}\left(T_i', T_j'\right) = 1$.

By substituting T_i and T_j into (3.3), we have

$$\delta_{i,j} = \Delta t + \left(m \cdot d \cdot T_i' - n \cdot d \cdot T_j'\right).$$

Therefore,

$$\delta_{i,j} = d \cdot \left\{\frac{\Delta t}{d} + \left(m \cdot T_i' - n \cdot T_j'\right)\right\}. \tag{3.9}$$

Let $\frac{\Delta t}{d} = k + x$, where $k = \texttt{int}\left(\frac{\Delta t}{d}\right)$ and

$x = \texttt{frac}\left(\frac{\Delta t}{d}\right)$.

By substituting $\frac{\Delta t}{d}$ into (3.9), we obtain

$$\delta_{i,j} = d \cdot \left\{ x + \left(m \cdot T_i^{'} - n \cdot T_j^{'} + k \right) \right\}.$$

Let $N = m \cdot T_i^{'} - n \cdot T_j^{'} + k$.

Then,

$$\delta_{i,j} = d \cdot (x + N). \tag{3.10}$$

Because $m, n, k, T_i^{'}, T_j^{'}$ are integers, N is also an integer. $T_i^{'}, T_j^{'}, d$ are constants. In (3.10), because $0 \leq x < 1$, the value of $\delta_{i,j}$ is minimized when $N = 0$. This leads to $\delta_{i,jmin} = d \cdot x$.

We need to prove $\forall k \in \mathbb{Z}$, $\exists m, n$ satisfies the condition $N = 0$. To this end, we use Bezout's identity as follows.

If gcd(T_i', T_j') $= 1$, then $\exists x, y$ are integers that satisfy

$$x \cdot T_i' + y \cdot T_j' = 1. \tag{3.11}$$

We could choose $m = -k \cdot x$ and $n = k \cdot y$, so (3.11) becomes $-\frac{m}{k} T_i' + \frac{n}{k} T_j' = 1$.

Therefore, $m \cdot T_i^{'} - n \cdot T_j^{'} + k = 0$, or $N = 0$. As such, we arrive at the formula we set out to prove.

3.2.5 Difficulty in predicting contention

We propose (3.8) to predict contention based on a strict mathematical proof. However, when we implement this method in actual WSNs, there are difficulties we need to resolve. Because of desynchronization (e.g., no global clock), source nodes cannot notify packet

send-time information to the sink. Therefore, the sink has to estimate packet send-time from the packet arrival time and an average transfer time (C).

However, the real transfer duration of each packet from the source node to the sink is not constant (even from the same source node), but such transfer durations change depending on the random backoff duration and system congestion. The congestion is unpredictable, but we could avoid it by choosing the channel free time, the time that there is no occurring transfer in the channel. The random backoff is unavoidable but has a constant average value. Therefore, the estimation is more accurate if the packet transfers in the channel free time.

3.3 Proposed methods

3.3.1 Application approach for collision problem

This section details the proposed method, the Contention Score Method (CSM). The main purpose of this method is to determine the suitable shift amount of the packet generation phase for each source node. After the source nodes shift their packet transmission timings, in contrast to TDMA-based methods, each source node does not need to strictly synchronize time with other nodes, which is one of the most attractive features of the proposed method. The method thus minimizes the number of contentions in general. The contentions that cannot be avoided in some random generation cases (e.g., control packets) are resolved by a MAC layer protocol (e.g., DCF).

As a specific application of the proposed method, we assume some data with small sizes are periodically generated. For example, WSNs that gather human health data such as blood pressure and heart rate, or environmental data such as temperature for IoT applications are considered. Therefore, all data generated at the application layer is quickly passed on to

lower layers by being contained into one IP and MAC packet (protocol data unit), meaning the delay caused when a large amount of data is divided into several packets does not occur.

Table 3.2 shows the different characteristics of the conventional interface layers, including MAC and physical layers, for other approaches and ours. The most important difference is represented by the goal. While the interface layer approach, because of physical channel characteristic, allows only one transfer at a time, our application layer approach, a virtual transfer, allows the concurrent transfer of multiple packets. Thus, the goal of the interface layer approach is to schedule each packet transfer without overlapping durations. On the other hand, the goal of our approach is to reduce the number of concurrent transfers. As the number of concurrent transfers decreases, collisions occur with less probability.

Moreover, the application layer approaches have different challenges. First, in the application layer, we could not expect the clock works correctly at microsecond. Hence, there should be a larger margin between scheduled consecutive packet transfer timings. Second, the transfer duration is long and unstable, which is the characteristic of application layer in multi-hop model. Third, our calculations use the schedule information of all nodes in the system. Therefore, the calculation burden is large and only the sink node, which usually has unlimited power supply and strong hardware, could handle it.

Finally, our approach has unique advantages. First, strict time synchronization is not required. This is very important for scalability. Second, because of its end-to-end viewpoint, a characteristic of application layer, it is topology independent. Therefore, our proposal is more flexible and can be applied to any types of topology. Third, this approach is independent from the interface layer. Consequently, hardware modifications are not required.

3.3.2 CSM overview

The flowchart of the proposed CSM method is shown in Fig. 3.1.

Table 3.2: Characteristics of the Proposal (Application Layer) and Interface Layer Approaches

Proposal Approach	Interface Layer Approach
Many concurrent transfers	Only one transfer at a moment
Minimize number of concurrent transfers	Avoid any overlap transfer
Clock is in second or millisecond order	Clock is in microsecond order
Consider all nodes in system	Consider nodes in transfer range
End-to-End aspect	Point-to-Point aspect

At first, the record is empty. When the first packet of a new $node_j$ arrives, the sink stores information about $node_j$ into its record. This information includes $(ID, t_{0j}, s_j, T_j, C)$, in which ID is the identification number of $node_j$, and the others have been defined in Section 3.2.1. The node provides T_j to the packet header. C is calculated by Eq. (3.1). In this calculation, the number of hops (n_h) is recorded in the packet header. This hop number is increased every time the packet is relayed. t_{0j} is recorded as the first packet arrival time. The sink then calculates a suitable s_j by using this information and its own record. Subsequently, the sink sends a request to shift the periodic packet generation time of the node. An extension of the original CSM is the CSM with rescheduling (CSMR), which reschedules the source node that has most packet loss.

3.3.3 Determining best shift amount

Here, we describe the process to determine the shift amount of packet generation time for each source node, which is the most important part of CSM. During this process, the sink finds the best shift amount for each node by checking several shift amount candidates (s). These candidates are selected by gradually increasing a value from 0 to the packet generation interval of node (T) in a $step$-wise manner. Because of the periodical characteristic, checking candidates larger than T is redundant. As the $step$ becomes smaller, the shift amount candidates are finer. However, the $step$ value should not be lower than the slot time of DCF

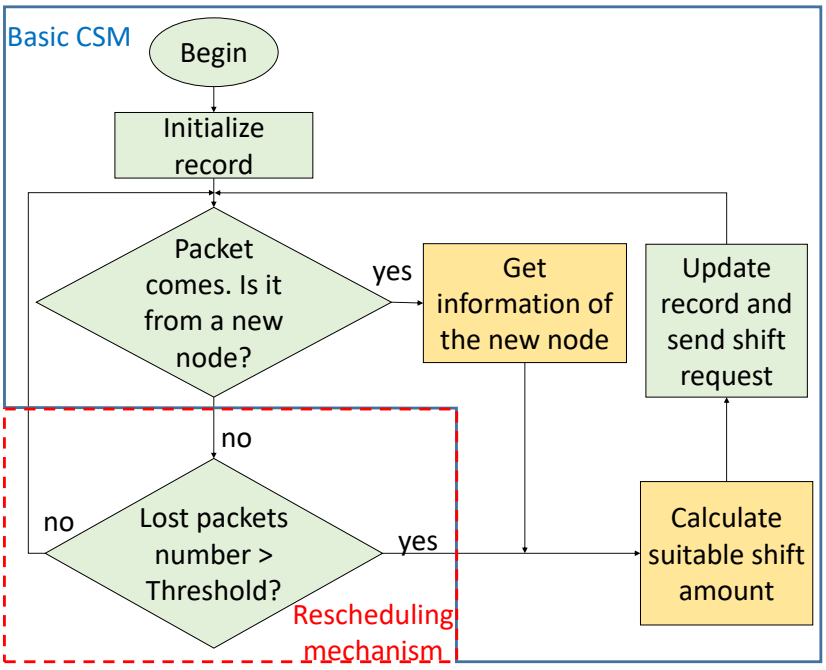


Figure 3.1: Flowchart of the proposed method, CSM.

$(20\ \mu s)$. In the simulation section, we use 100 μs as $step$ value.

The procedure for determining the suitable shift amount, (s_j) of $node_j$, is described in Algorithm 3.1, where N is the number of shift amount candidates. For each shift amount candidate (s), a contention score number (CS) is calculated. This calculation is expressed as $Calculate\_CS(s)$ in Algorithm 3.1 and is explained in the next paragraph. After calculating all CSs for all candidates, the s with the smallest CS $(minCS)$ is determined as the suitable shift amount (s_j).

Further, the contention score number (CS) is calculated to judge the shift amount s. The larger the CS, the longer and more frequent contention time is. The CS of $node_j$ is calculated as

$$CS = \sum_{i=1}^{j-1} \frac{t_c}{T_{ij}}, \tag{3.12}$$

where t_c determines the contention duration in the common interval of $node_i$ and $node_j$. We use Eq. (3.8) in Section 3.2 to calculate t_c. The value of t_c is scaled with contention duration. Therefore, minimizing CS also means minimizing contention duration.

The common interval of $node_i$ and $node_j$ (T_{ij}) could be calculated using Eq. (3.7), but it is also the generation interval of the two nearest packets. The larger the T_{ij} is, the less often the two nearest packets will be generated.

As shown in Algorithm 3.1, to find the suitable s_j, we calculate CS with each $s \in [0, T_j]$ and choose s_j as the smallest CS.

Algorithm 3.1 Algorithm of determining s_j in CSM

```
1: procedure FINDING SHIFT AMOUNT
2:     N ← T_j / step
3:     for index from 0 to N − 1 do
4:         s ← index ∗ step
5:         CS[index] ← Calculate_CS(s)
6:     end for
7:     [minCS, index_minCS] ← Find_Min_CS(CS)
8:     s_j ← index_minCS ∗ step
9:     return s_j                          ▷ The suitable shift amount
10: end procedure
```

3.3.4 Essential enhancements to CSM

CSM is based on a mathematical formula that requires knowing the exact value of transfer duration (C). Unfortunately, because of congestion and the random backoff mechanism, the value of C is not stable. Therefore, the solutions to this problem are essential to our proposal.

3.3.4.1 Choosing suitable time to calculate the schedule

This enhancement improves the "Get Information of the New Node" module in Fig. 3.1. Figure 3.2 shows the idea of this enhancement. When the packet of a new node arrives at the sink, the sink checks the free channel duration from the time it received the last packet. If the free channel duration is larger than C, which means no redundant delay, such as queuing or retransmission, could not occur just before the arrival with high probability, we use the packet arrival time to calculate the shift amount of the new node. On the other hand, if this free channel duration is smaller than or equal to C, which means the packet from the new node would delay its transfer until the previous other transfer finished, then the channel would suffer contention at that moment with a high probability. In this case, the estimation of the packet send-time is inaccurate because of the waiting time. Therefore, the sink does not use this packet arrival time but the next one satisfying the "good" condition in Fig. 3.2.

Algorithm 3.2 shows the detailed implementation. First, when a packet arrives, the sink will obtain the information in the packet header. T_c is the current time of the system, ID the node identification number, and $freeDuration$ the duration from the last packet arrival to the current time. If ID is not in the $record$, this packet owner is a new node. Next, if $freeDuration$ is larger than C, the current time is in the good state, as shown in Fig. 3.2, and the sink calculates the shift amount and sends the shift request. If $freeDuration$ is smaller than C, the current time channel is busy (not good) and we send a small shift request (C) and wait for the next packet. Finally, $lastArrivedTime$ is updated.

3.3.4.2 Choosing middle of minimum Contention Scores

When we calculate the contention score of each candidate shift amount, we typically find some group of continuous shift amount candidates whose contention score equals zero.

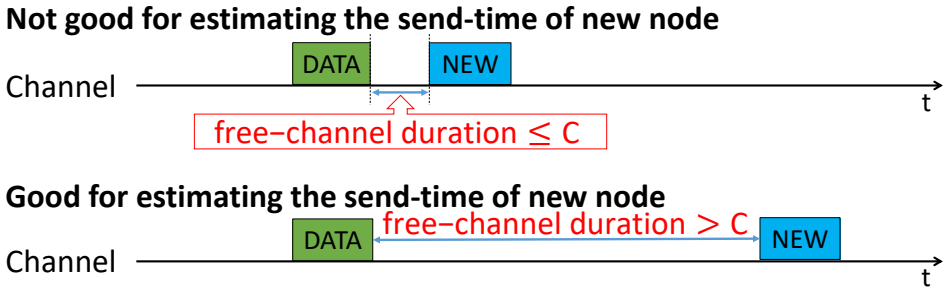


Figure 3.2: Choosing better estimating packet send-time of new nodes.

Algorithm 3.2 Algorithm of choosing the arrived packet to calculate the shift amount

```
1: procedure DECISION TO CALCULATE THE SHIFT AMOUNT
2:     T_c ← System time
3:     C ← Total transferred time of packet
4:     freeDuration ← T_c − lastArrivedTime
5:     if ID in record then
6:         if freeDuration > C then
7:             Make new record in record table.
8:             Calculate shift amount using Algorithm 3.1.
9:             Send shift request with calculated shift amount.
10:        else
11:            Send shift request that shift amount equals C.
12:        end if
13:    end if
14:    lastArrivedTime ← T_c
15: end procedure
```

Hence, we can choose any shift amount candidate among these and the node will never contend with other nodes as in Fig. 3.3. However, because the contention score uses the estimated transfer time, which is not always accurate, we should use the shift amount candidate that puts the packet send-time in the middle of longest free-channel duration. In this way, we create a margin that alleviates the accuracy requirement when we estimate transfer time.

Algorithm 3.3 shows the detailed process. First, we create an array of shift amount candidates ($arrSA$). With every element in $arrSA$, we calculate the Contention Score

by using Eq. (3.12) and create an array of Contention Score ($arrCS$). Subsequently, we determine the minimum value of $arrCS$ ($minCS$). Then, we find the longest sub-array of $arrCS$ ($subCS$), which has all elements equal to $minCS$. Finally, the middle element in $subCS$ is determined. The corresponding shift amount with this middle element is the one we want to determine.

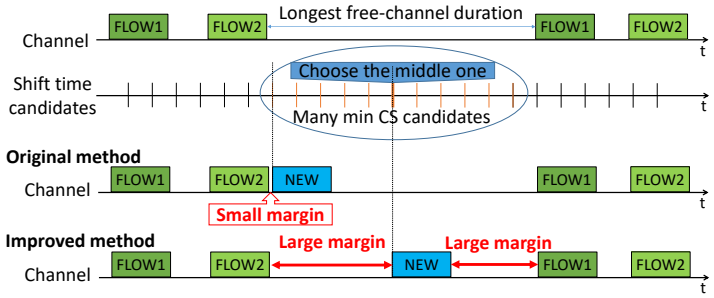


Figure 3.3: Choosing the middle from the minimum CS candidates for shift amount candidates.

As the MAC and physical layers work in the order of microseconds, it is impractical to achieve this in the application layer. However, by using the improvement in this subsection, we can increase the margin between packets from $step$ (100 μs) to a half of the longest free channel duration. The longest free channel duration has the same order with the data interval, which is usually in the order of milliseconds or larger. Because the margin is large, the proposal does not require a very accurate clock in the application layer to work effectively.

3.3.4.3 Multi-hop parallel transfer for CSM

In a multi-hop environment, packets can be spatially sent in parallel. For simplicity, we assume the information on network topology that the sink can collect only the number of hops from the source node to sink (n_h). The value n_h is determined by adding a variable to the packet header and incrementing it every time the packet is relayed to the next node.

Algorithm 3.3 Choosing middle of minimum contention scores

```
1: procedure FINDING SHIFT AMOUNT OF MIDDLE OF MINIMUM CONTENTION SCORE
2:     T ← Get Node Interval
3:     arrSA ← Create Array(0 : step : T)
4:     for (index, shiftAmount) in arrSA do
5:         arrCS[index] ← Calculate_CS(shiftAmount)
6:     end for
7:     minCS ←Minimum value in arrCS
8:     subCS ← longest sub-array of minCS in arrCS.
9:     index ← index of middle element in subCS.
10:    suitableShiftTime ← arrayST[index]
11: end procedure
```

By using this information, we divide the nodes into groups with different n_h levels, as illustrated in Fig. 3.4. When a sensor node at level i sends a packet, it could interfere with nodes at levels $i+1$ and $i-1$. Conversely, if $|i-j|>3$, node levels i and j will not interfere with each other. Therefore, the C value will be reduced by the parallel transfer time. Using this characteristic, we can more accurately predict contention time and improve CSM effectiveness.

3.3.5 CSM with rescheduling (CSMR)

Usually, the first scheduling of the sink places a node into a good schedule. However, we can improve the packet loss rate even more by subsequently rescheduling nodes. This is why our proposal depends on estimated packet send-time, which is not always accurate. If there is a random error higher than the safety margin, our first schedule might not work as expected. In multi-hop communication environments, contention is a serious problem because if there is hidden node problem, packet collision surely occurs.

The sink can then determine the node which needs rescheduling by counting the lost packets. The number of lost packets is measured by numbering the data packet. When

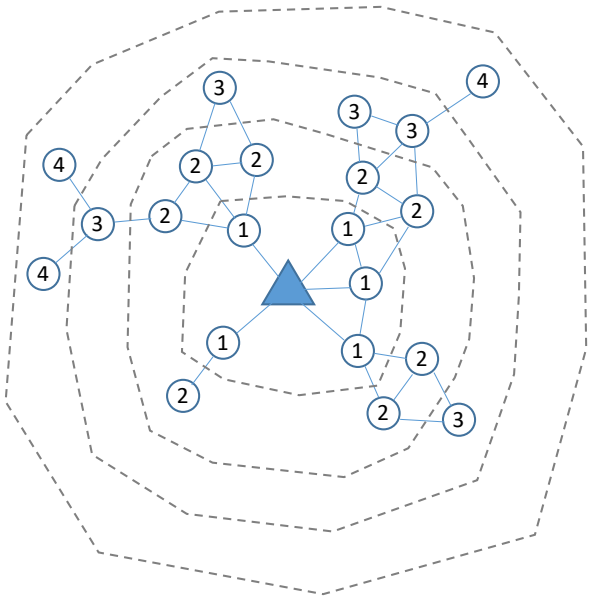


Figure 3.4: Hop level diagram.

the sink detects lost packets for a node whose number is higher than a threshold, it will recalculate the schedule and send a new shift amount request to the node.

When the sink recalculates the shift amount of the node, the shift amount will have a new value because of the following two reasons. First, the sink uses the newest packet arrival time for the calculation. Second, the record of the sink is continuously updated, that is, it is changed every time a new node comes or a node is being rescheduled.

Algorithm 3.4 presents implementation details. This procedure is processed when a packet arrives to the sink. First, we need to update the arrival time in the record table. Second, N_s and $lastN_S$ are the sequence numbers of the current and last packets, respectively. Because the interval of data generation is typically much larger than the end-to-end delay, the packets should arrive in order. By using the difference between sequence numbers, we can detect the number of lost packets ($lostPacket$). Third, when $lostPacket$ is larger than the threshold (t_h), the new shift amount (s_{re}) is recalculated and sent to the corresponding node. Finally, $lastN_s$ is updated.

Algorithm 3.4 Contention score with rescheduling mechanism

```
1: procedure RESCHEDULING MECHANISM
2:     Continuously update the arrived time in Record Table.
3:     N_s ← Sequence number in packet header
4:     t_h ← Threshold of the Rescheduling mechanism
5:     if N_s − lastN_s > 1 then
6:         lostPacket = lostPacket + N_s − lastN_s − 1
7:     end if
8:     if lostPacket ⩾ t_h then
9:         s_re ← Using Algorithm 3.3 to find shift amount
10:        Send a shift request that shift amount equals s_re.
11:    end if
12:    lastN_s ← N_s
13: end procedure
```

3.3.6 Complexity of CSM

The contention score needs to be calculated every time the sink reschedules the source node. The complexity of this calculation is based on the number of nodes in the sink record ($|S|$) and the number of shift amount candidates ($n = T/step$). Therefore, because $step$ is constant, the time complexity will be $\Theta(|S| \cdot T)$.

Space complexity includes the memory amount required to store a record. The number of records is equal to the number of the source nodes, which is $|S|$. Therefore, the space complexity of CSM is $\Theta(|S|)$.

3.4 Performance evaluation

Here, we present the simulation results by comparing our proposed methods, CSM and CSMR, with other methods: the general-case DCF and the previous shift time methods DSM and BDM.

3.4.1 Simulation environment

3.4.1.1 Single-hop simulation

We use QualNet v5.2 [38] for our simulation and the detailed parameter settings are described in Table 3.3.

Table 3.3: Details of Simulation Parameters

Name of parameter	Value
Data packet size	128 Bytes
Physical layer	802.11b
Data rate	2 Mbps
MAC layer	802.11
Retransmission limit	1
RTS/CTS	N/A
Routing	Static routing

We measure the packet loss rate in a single-hop transmission environment to precisely evaluate the collision prevention performance of the proposed method without external factors. Although some WSNs use a multi-hop model, single-hop networks with more popular and cheaper equipment are also used for applications such as body area networks, to obtain life-log or health information, Wi-Fi-based sensing systems in rooms for elderly care in apartments or hospitals, wide area sensor networks based on the IEEE 802.11ah, etc.

Under this scenario, all source nodes are allocated at random in the circumference of the propagation range of a sink so that they communicate with the sink via a single-hop. The packet generation interval of each source node is chosen from {100 ms, 200 ms, 300 ms, and 400 ms}. The first, second, third, and fourth node intervals are 100 ms, 200 ms, 300 ms, and 400 ms, respectively. Then, the intervals are repeated (i.e., the fifth and sixth are 100 and 200 ms, respectively, and so on). All nodes begin sending a data packet at a

random time between 0 and 5,000 ms and repeat sending a packet at their own interval. The number of nodes increases from 20 to 100, by 10 nodes. Because the real transfer duration of a packet, including the MAC layer backoff process, is around 1.5 ms, the channel utilization (U) will be 16% and 78% in the models with 20 and 100 nodes, respectively.

An interesting feature of the proposed method is that, while many TDMA-based methods cannot be used if the U value is above 50% [32][34], our proposal can work for even very congestion condition.

In this simulation setting, we set the retransmission limit to 1 and the RTS/CTS handshake is disabled to avoid increasing end-to-end packet transfer delay and peak packet queue length at each node. Longer packet transfer delays affect communication quality and larger peak packet queue lengths consume more memory on sensor nodes, which should be cheap and have limited computational performance. The case of the default setting of the retransmission limit with enabling RTS/CTS is evaluated in Section 3.4.4, and we demonstrate that a larger retransmission limit and enabling RTS/CTS affect packet transfer delays and the peak packet queue lengths.

3.4.1.2 Multi-hop simulation

We use the same environment characteristics as in the single-hop model, except for the following modification. All nodes are put in a square area[39], by unifying a randomized method with a sink in the middle, as shown in Fig. 3.4. The side of the square is 200 m. The transfer range of nodes is 32 m. The isolated node problem is eliminated by re-randomizing new node positions. Because the collision problem in the multi-hop simulation is more serious, the retransmission limit is set to 2. The transfer route of each packet is statically determined before running the simulation. Actually, the hop number from the source node to the sink is varies from one to six hops.

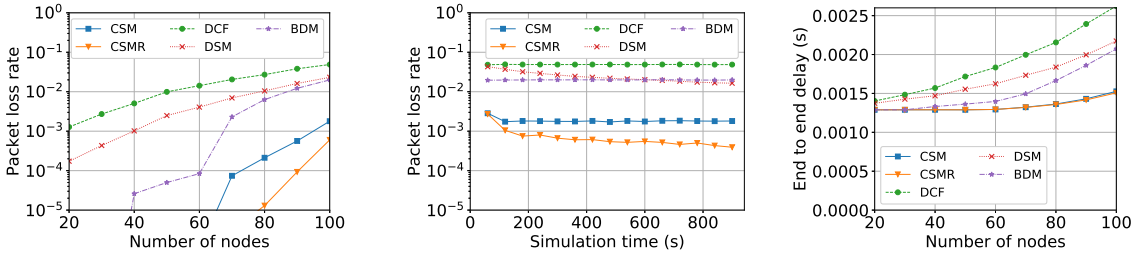
Second, because transfer time is increased, to maintain channel utilization, we increase the data intervals from 100, 200, 300, 400 ms to 300, 600, 900, 1200 ms, respectively. Therefore, the U value is around 19% to 91% in the 20 nodes and 100 nodes models, respectively.

3.4.2 Packet loss rate

This section details the comparison of packet loss rates (PLR) among DCF, DSM, BDM, CSM, and CSMR.

3.4.2.1 Packet loss rate by number of nodes

Because all methods require a transition time to become stable, we obtain the average PLR in 60 s–900 s simulation time, as shown in Figs. 3.5a and 3.6a. These figures show our method works effectively in both single- and multi-hop models. The native DCF has the worst performance, because it does not adjust the system and there are continuous contention and collisions between data flows. DSM does not perform well on average because it requires a significant amount of time to collect collision information and only slightly improves the system. Because of the power of the two coefficients [13], BDM performs well with a small number of nodes and greatly reduces performance within a higher density environment. CSM is better than BDM because its detailed mathematical model reduces the contention in the system more effectively. Finally, the CSMR is the best because it inherits the specific mathematical model of CSM and has the improving characteristic over time.

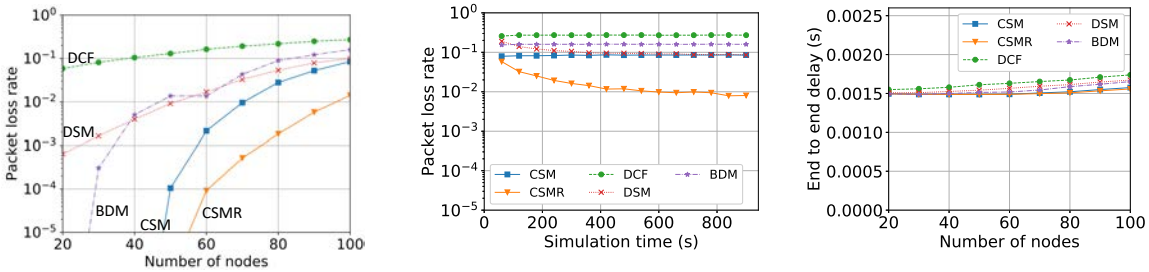


(a) Packet loss rate by the number of nodes at 60 s–900 s simulation time.

(b) Packet loss rate over time in 100 nodes.

(c) End-to-end delay in single-hop model.

Figure 3.5: Single-hop model simulation results.



(a) Packet loss rate by the number of nodes at 60 s–900 s simulation time.

(b) Packet loss rate over time in 100 nodes.

(c) End-to-end delay in multi-hop model.

Figure 3.6: Multi-hop model simulation results.

3.4.2.2 Comparison of packet loss rate over time

Figures 3.5b and 3.6b show the packet loss rate by time in the 100-node model. CSM, DCF, and BDM do not change PLR over time because of the lack of a rescheduling mechanism. First, CSM and CSMR are the best methods because they use a good scheduling formula. Second, DSM improves performance over time but the process is slow because DSM only uses a small constant shift amount. By contrast, CSMR calculates shift amount based on the above mathematical formula and surpasses all other methods in long term. At 900 s, in the multi-hop model, CSMR PLR is reduced by 78.35%, 99.19% 97.61%, and 98.03%

in comparison to CSM, DCF, DSM, and BDM, respectively. In summary, for a short-run application (below one minute), we should use CSM for simplifying the implementation. However, for long-term applications, we should use the CSMR to obtain the best performance.

3.4.3 End-to-end delay comparison

Figures 3.5c and 3.6c show the end-to-end delay for each packet to reach the sink from its source node in single- and multi-hop models, respectively. Because our proposal reduces contention time, it also reduces the time a packet waits for the channel to be free.

The effect is clearly shown for a high-density environment. In the single-hop model, for the 100-node model, CSMR, reduces the end-to-end delay by 42.55%, 30.66%, and 27.29% compared to DCF, DSM, and BDM, respectively. In a low-density environment, our proposal improves the end-to-end delay slightly.

3.4.4 Comparison of different MAC setting

Here, we conduct simulations with different MAC parameters to observe the performance differences of the different settings in the proposed methods and related studies. This is important because some methods may work only in a specific situation. Moreover, this also indicates the suitable MAC parameters when we implement the proposed methods under particular applications.

3.4.4.1 Simulation settings

We change two important parameters in the MAC layer: the retransmission limit and RTS/CTS mode. For comparison, we use three different settings. In the first setting, the retransmission limit is set to 2, an intentionally smaller value because of the lower latency and simpler nodal device requirement, and we disable RTS/CTS (Retransmit 2). These settings

were used in Section 3.4.1.2 (multi-hop simulation). In the second setting, we increase the retransmission limit to 7 and disable RTS/CTS (Retransmit 7). This retransmission limit is the default value for many current wireless devices. In the last setting, the retransmission limit is 2 and we enable RTS/CTS (R2-RTS/CTS enabled). These settings aim at examining the effects of the RTS/CTS handshake. To concentrate on the most complex situation, we use the multi-hop model with 100 nodes. The other parameters are described in Section 3.4.1.2.

3.4.4.2 Packet loss rate comparison

Figure 3.7 shows the packet loss rate comparison. In all three settings, our final proposal, CSMR, shows the best performance.

When the retransmission limit is increased to 7, PLR decreases in all methods. The decrease ratios in CSMR and DSM are smaller than in the other methods, because they need to detect packet loss to improve their schedules as to shift packet generation times. However, CSMR maintains its PLR minimum among the compared methods because of its good mathematical model so as to decide the shift amount on each source node.

On the other hand, enabling the RTS/CTS handshake increases PLRs in all methods, except for DCF. This is because RTS/CTS creates a large number of overhead control packets and disrupts shift schedules.

3.4.4.3 End-to-end delay comparison

The results of the end-to-end delay are shown in Fig. 3.8. The end-to-end delay is also minimized in our final proposal, the CSMR.

When the retransmission limit is increased to 7, the end-to-end delay also increases in all compared methods. This is because packets can be retransmitted several times before they reach the sink or are dropped due to the retransmission limit. However, our proposals,

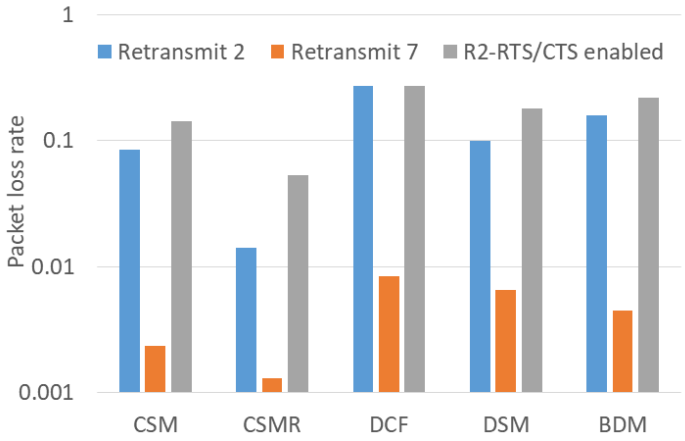


Figure 3.7: Comparison of PLR under different MAC settings in multi-hop model.

CSM and CSMR, increase the delay by smaller amounts because the proposed methods create a large margin between two neighboring packets on the time axis. This margin gives transferred packets a capacity for retransmission if collision happens.

The RTS/CTS handshake also increases end-to-end delay, because the duration for sending and receiving RTS/CTS increases the total amount of the end-to-end delay.

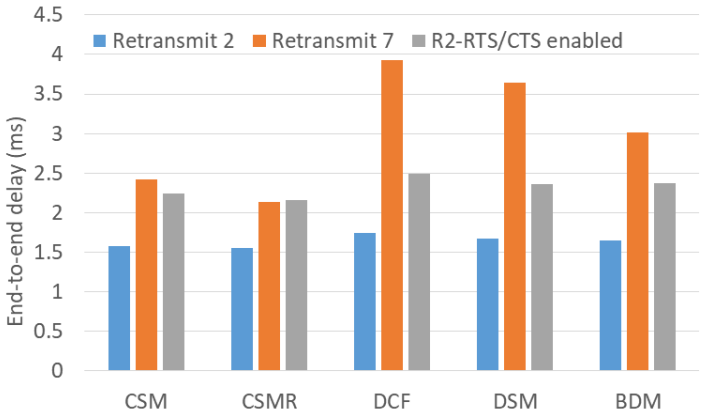


Figure 3.8: Comparison of the end-to-end delay under different MAC settings in multi-hop model.

3.4.4.4 Peak queue length comparison

Here, we show the peak of packet queue lengths for all sensor nodes. For any node, when packets cannot be sent at that moment, they need to stay in the node's queue. Longer queue peaks require more memory on each node, which increases the hardware requirements of each node. However, in WSNs, sensor nodes are typically cheap and have limited hardware.

Figure 3.9 compares peak packet lengths. When the retransmission limit is increased to 7, the peak queue length also increases largely because the waiting time until transmission is increased.

However, our proposals have shorter peak queue lengths even when the retransmission limit is 7. Because CSM and CSMR have fewer collisions, fewer packets need to be retransmitted. Moreover, because CSM and CSMR distribute evenly the traffic in the network, they reduce choke time and, therefore, reduce the queue length peaks.

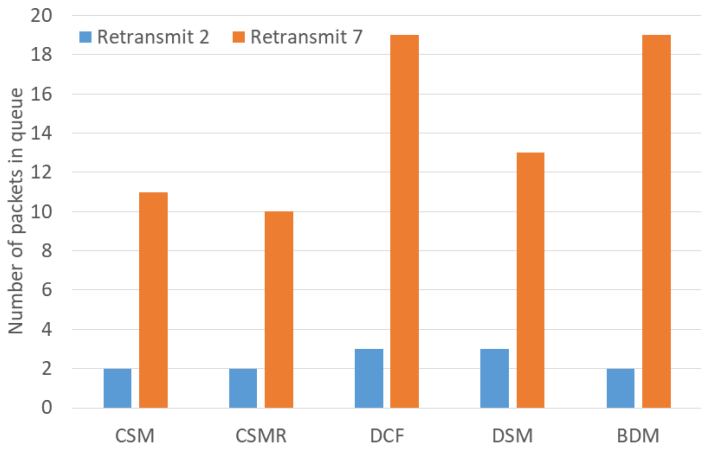


Figure 3.9: Comparison of peak queue lengths under different MAC settings in multi-hop model.

3.4.4.5 Conclusions about MAC settings

First, enabling the RTS/CTS handshake degrades system performance. Second, increasing the retransmission limit is a trade-off among PLR, end-to-end delay, and the memory capacity of nodes. Finally, the proposed method could work effectively under different settings.

3.4.5 Choosing CSMR threshold

The CSMR threshold (Threshold) is defined as a certain number of lost packets. If a node has more lost packets than the threshold, it will be rescheduled. After that, the number of lost packets in all nodes will be reset. If the threshold is too small, the system will be too sensitive and will generate too many shift requests. In contrast, if the threshold is too large, the system requires a longer time for saturating.

We measure the average packet loss rate and sum of shift requests of the CSMR as 600 s–900 s in the 100-node model.

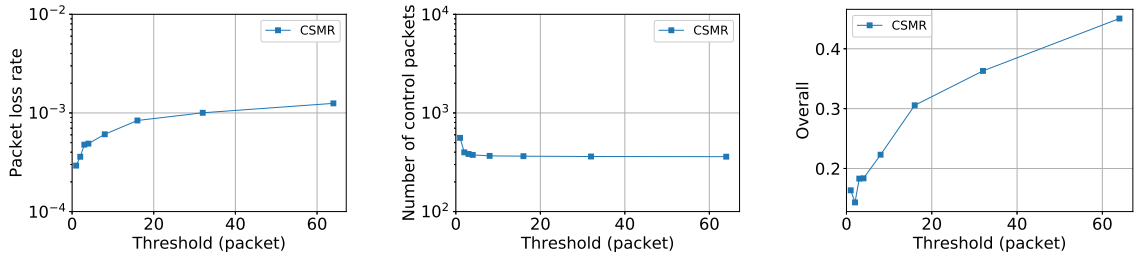
Figures 3.10a and 3.10b show the PLR and sum of control packets by the threshold value in the single-hop model. Because of the slow convergence, the average PLR increases when the threshold value increases. By contrast, the number of control packets is large when the threshold value is low. To balance these two values, we suggest an $Overall$ number, which is calculated by

$$Overall = PLR \cdot NumberOfControlPackets.$$

We do not need to normalize PLR and NumberOfControlPackets to [0-1] because we only want to determine the minimum value of $Overall$. Figure 3.10c shows the $Overall$ value by the threshold value. Because we want to determine the threshold that keeps both PLR and the number of control packets low, we choose the threshold that minimizes the value of $Overall$. As per Fig. 3.10c, in the single-hop case, the best threshold value is 2.

Similarly, Figs. 3.11a and 3.11b show the PLR and sum of control packets by the

threshold value in the multi-hop model. By using the $Overall$ value in Fig. 3.11c, the best threshold value for the multi-hop model is 4.

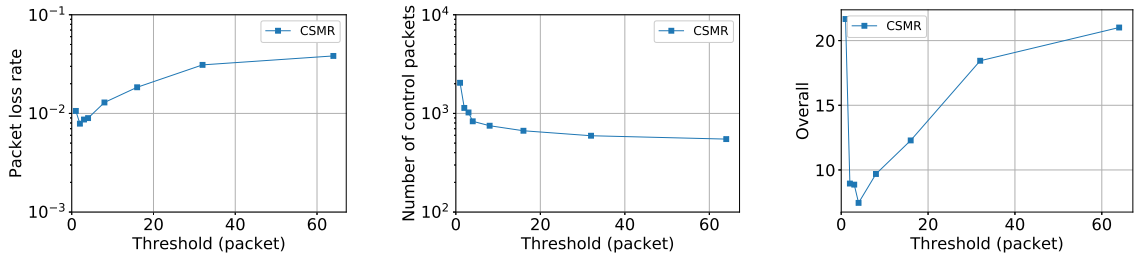


(a) Effect of threshold on packet loss rate in single-hop.

(b) Effect of threshold on number of control packets in the single-hop model.

(c) $Overall$ value by threshold in the single-hop model.

Figure 3.10: Finding the best threshold for CSMR in the single-hop model.



(a) Effect of threshold on packet loss rate in the multi-hop model.

(b) Effect of threshold on number of control packets in the multi-hop model.

(c) $Overall$ value by threshold in the multi-hop model.

Figure 3.11: Finding the best threshold for CSMR in the multi-hop model.

3.5 Conclusion

In this Chapter, to solve the inherent problem of repeated packet collisions, we designed a new formula to address the contention problem in heterogeneous periodic flows system. Based on this formula, we proposed protocols that schedule packet creation timing at each

source node. The basic proposed scheduling method, CSM, is simpler and shows better performance in short run applications. The extended proposed method with rescheduling function, called CSMR, is best for long run applications. This property was preserved in the multi-hop network environment. In the next Chapter, we will deal with another problem in periodic flows system.

Chapter 4

Prediction of Hidden Transfer: a MAC layer approach

This chapter has the same research theme of improving periodic multi-hop networks performance with Chapter 2 and 3. However, in this chapter, we describe the proposed method for solving another problem; the compounded negative effect of the hidden node and the continuous collision problem. Furthermore, the new proposed method works in MAC layer, while the previous ones are in application layer. The content of this Chapter corresponds with the works that published in [15].

4.1 Problems and related works

4.1.1 Target System

Figure 4.1 describes the common overview of our target system. Every sensor node transmits a data packet every constant period to its sink. Specifically, at a random time, it initiates the transmission of the first data packet to the sink. Subsequent to the initial packet, data

packets are transmitted periodically. The sink collects all the data packets. The interval of data transmissions among the nodes may vary. The data packets are then transferred through several relay nodes before arriving at the sink. A multi-hop network is used for increasing the network coverage area.

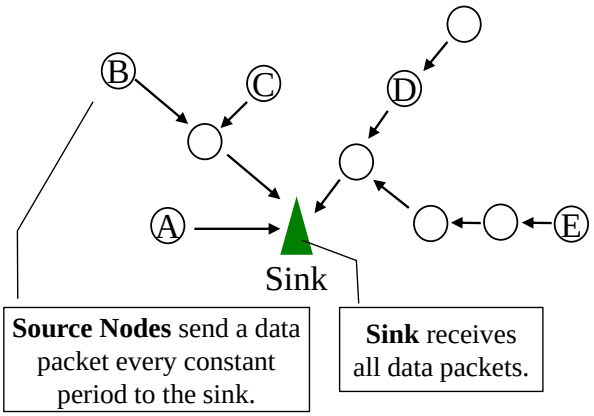


Figure 4.1: Our target system model.

There are some characteristics that the proposal for this system should satisfy. We focus on four characteristics. First, the system is dynamic; the node may arrive and leave anytime. Thus, methods that require a priori information are not suitable.

Second, the number of nodes in the system could be large, with random topology. In this Chapter, we simulate a random mesh system with up to 100 nodes. Most TDMA (Time-Division Multiple Access)-based methods are not suitable for such systems because they do not comply with the time synchronous requirement among sensor nodes.

Third, because of the presence of several types of sensor nodes, the resulting data flows may have different periods. In general, a system where the periodic flows are heterogeneous is much more complex than its homogeneous counterpart.

Finally, the fairness of the system is critical aspect. In most applications, the lost packets should separate more evenly between all data flows than on only some flows.

In the MAC layer, we use the CSMA/CA (Carrier-Sense Multiple Access with Collision

Avoidance) mechanism. CSMA/CA is used across popular standard protocols, including IEEE 802.11 and IEEE 802.14.5. In CSMA/CA, before sending a data packet, the source node senses the channel status. Then, if the channel is free, the source node will initiate a random backoff count down. At the end of the countdown, the transfer is initiated. Finally, the destination node confirms a successful transfer by sending an ACK (ACKnowledgment) to the source node.

In this thesis, we focus on IEEE 802.11 standard because some real devices can use only this standard, with which we are going to implement our method as a near future work, and IEEE 802.11 is more preferred than IEEE 802.15.4 [40]. Although IEEE 802.15.4 is also expected to be applied to WSNs, IEEE 802.11 has been applied to WSNs in many researches [41][42][43][44]. In practice, there are some new WSN devices (e.g. ESP8266, ESP32) that use IEEE 802.11 standard [45]. However, our proposed method can apply also to IEEE 802.15.4 with a small modification, because the method uses only ACK and overheard packets' header at each node for the control, and no additional control packets are required. We will consider the application to IEEE 802.15.4 as a future work.

4.1.2 Continuous packet collisions

The problem associated with continuous packet collisions is illustrated in Fig.4.2. All sensor nodes transmit a data packet at fixed periods; when two nodes have the same interval and contend once, they will contend continually. Further, even when the interval of two nodes (T_1 and T_2) are different, owing to the periodic nature, if they contend once, they will contend again repeatedly at subsequent Least-Common-Multiple (T_1, T_2) durations.

This continuous contention problem creates several contention durations during which collisions can occur. In the IEEE 802.11 standard, the CSMA/CA mechanism addresses the contention problem to some extent by checking the channel status. However, in certain

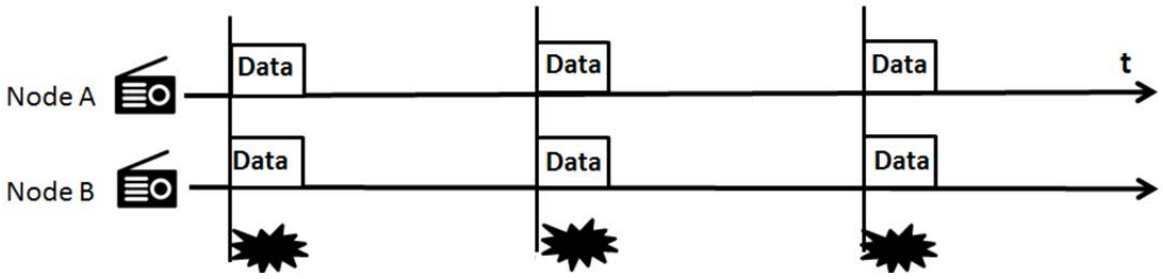


Figure 4.2: Continuous packet collisions.

scenarios, such as in the presence of a hidden node, the channel may not be sensed accurately.

Several studies have attempted to address the contention problem. The approaches can be broadly classified into three main categories: synchronized TDMA [16, 18, 32–35], unsynchronized TDMA [30] and non-TDMA [13][37][46] based methods.

The synchronized TDMA-based methods suffer from the time synchronization problem. Time synchronization in a scaled up WSN is hard to achieve and thus is impractical.

In unsynchronized TDMA-based method [30], the channel time is divided into even intervals. Each node checks the number of neighbor nodes and evenly divides the interval into equal number of time points. Each node waits and only starts transferring packets at its own time point. The proposal is not suitable for our target system because of two reasons. First, this paper assumes that all nodes send data with the same period. Second, the waiting-for-transferring mechanism, as described above, significantly increases the end-to-end delay every relayed hop.

Therefore, we select non-TDMA-based methods. In [13][37][46], the authors propose shift-time methods. An overview of these methods is illustrated in Fig.4.3. The problem of continuous contentions is addressed by shifting the data transfer schedule to a suitable duration at a contending source node. These proposals are implemented on the application layer and have the potential to be cooperated with our new proposal on the MAC layer

presented in this Chapter, to further improve the performance of the system.

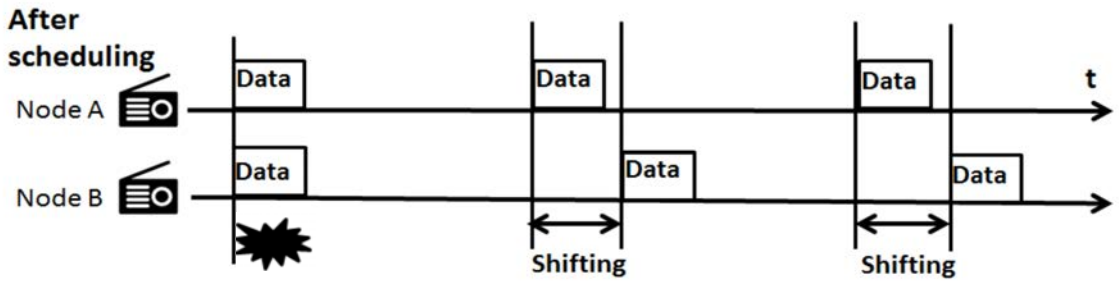


Figure 4.3: Shifting time method.

4.1.3 Hidden node problem

An example of the hidden node problem is shown in Fig.4.4. This assumes a case that node A transmits a packet to node B and node C also has a packet to send to node B. Because node C is not in the transmission range of node A, it cannot know whether node B is receiving a packet. Here, if node C begins the transmission of its packet, the collision between the packets occurs and ruins both packets.

In [6][7][8], the authors have attempted to detect or address this problem. One such approach is a static approach that checks statistics at MAC layer, like the number of overheard packets from neighbor node, to detect hidden nodes [6]. This work can be applied only to IEEE 802.11n. Alternatively, the relation between data packet and ACK can also be used as proposed in hidden station detection (HSD) [7]. Hidden nodes are determined when an ACK is observed without the corresponding data packet. After detecting a hidden node, these related works activate RTS/CTS mechanism to address the hidden node problem. However, RTS/CTS tends to incur large overhead. In another research, a hidden node is detected by comparing the relationship between the durations of transmitting and interference signals [8]. However, in the impaired channel environment, this method is

not feasible.

A research about hidden node problem in vehicular ad-hoc network is introduced in [9]. This method uses ready-to-broadcast/clear-to-broadcast (RTB/CTB) signal to avoid the hidden node problem. Each of RTB and CTB includes the location information of the vehicular, which helps detecting hidden nodes. This research is different from our target system in which the location information of nodes is not used. Furthermore, RTB/CTB causes heavy overhead in WSNs that use small data packets as our target system.

Shin and Chung [10] investigate more about RTS/CTS mechanism. This paper shows that, in high bit error rate environments, the RTS/CTS mechanism cannot guarantee successful transmission of data packet. Thus, RTS/CTS is not a good solution for hidden node problem in such conditions.

Reference [11] tries to find the suitable communication range of each node to balance the exposed and hidden node problem and to maximize the throughput in linear wireless network. The communication range depends on the active rate of nodes. This method is different from our proposal because of the following reasons. First, this method is applicable only to linear topology. Second, modifying the communication range requires more hardware cost of the nodes. In contrast, our proposal can apply for all topologies and does not change the communication range.

In [12], a grouping algorithm is proposed to deal with hidden node problem. This proposal has three steps. First, the information about nodes that are not in hidden node topology is collected. Second, the nodes are divided into groups so that each group is free from hidden nodes. Finally, each group is assigned time slots to transfer its packets. This method requires the time synchronization among the nodes, which is very hard to be archived in large WSNs.

Through the above descriptions, we can explicitly note that the combined effect of

continuous contention and hidden node has not been considered in these related works. Each of these methods has different characteristics and constraints as shown in Table 4.1. By using this information, we found that HSD [7] has common assumptions with ours and could apply to our target system. Therefore, we will use this method to compare with our proposed method in performance evaluation section (Sect. 4.3).

Table 4.1: Comparison of researches to deal with HNP (hidden node problem).

Constrains	Proposed method	HSD [7]	[6]	[8]	[9]	[11]	[12]
Assumed network	Periodic WSNs	General WSNs	IEEE 802.11n only	General WSNs	Vehicular network	Linear network	General WSNs
Treat/Detect HNP	Treat HNP	Treat HNP	Only detect	Only detect	Treat HNP	Treat HNP	Treat HNP
Require location	No	No	No	No	Yes	No	Yes
Require synchronization	No	No	No	No	No	No	Yes
Require extra control packet	No	RTS/CTS	No	No	RTB/CTB	No	Yes

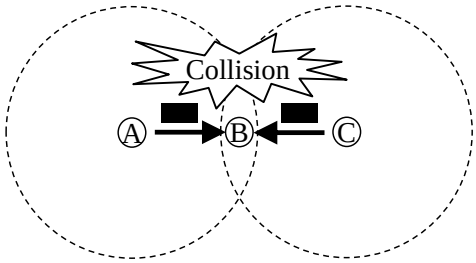


Figure 4.4: Hidden node problem.

4.2 Proposed method

The proposed method can be decomposed into three steps. First, each sensor node records the information of interfering flows by overhearing data packets in the flows for itself and

the corresponding ACKs (Sect. 4.2.1). Second, based on this information, each node then detects the corresponding hidden nodes (Sect. 4.2.2). Finally, each node predicts a risky duration during which collision can be caused by the hidden nodes (Sect. 4.2.3) and the node stops its packet transmission in order to avoid collisions (Sect. 4.2.4).

4.2.1 Recording information of interfering flows

In the proposed method, each data packet includes its flow ID and the generation interval in its header. Each ACK, obtained from the receiver nodes of data packets, has flow ID, size information, and the generation interval of the corresponding data packet in its header.

When a node overhears an ACK and if a hidden node is detected by the ACK based on the process in Sect. 4.2.2, the flow ID, the data packet size, and the data packet generation interval included in the header are recorded in addition to the arrival time of the ACK. This information is used to avoid collisions, as described in Sect. 4.2.3.

4.2.2 Detection of hidden nodes

Each sensor node detects its hidden node, whose packet transmission can be in contention with the transmission from itself. Specifically, if a node overhears only an ACK and does not overhear the corresponding data packet, the receiver of the ACK (the sender of the data packet) is detected as its hidden node.

In the example shown in Fig.4.5, when node A sends a data packet to node B, and B replies with the ACK, there may be three types of positions at which a node overhears the transfer process.

1. Node C is in the transmission range of both A and B, and overhears both the data packet and the ACK. Thus, node C detects no hidden nodes.

2. Node D is in the transmission range of only node B, and overhears only the ACK. Thus, node D detects node A as its hidden node.

3. Node E is in the transmission range of node A, and detects the data packet. By referring to the duration field of the data packet, the corresponding (forthcoming) ACK from node B will be also detected virtually. Thus, node E detects no hidden nodes.

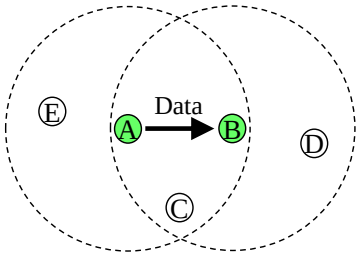


Figure 4.5: Position relationship related to hidden node problem.

4.2.3 Prediction of future risky durations

Each node estimates a risky duration in which one of its hidden nodes is transmitting a periodic data packet; ideally, the node should not send its packet in order to avoid the contention problem.

The process to estimate the risky duration is shown in Fig.4.6 that assumes nodes' position in Fig.4.5. When a node (Node D in Fig.4.5 and Fig.4.6) overhears an ACK, it calculates the arrival time of the next data packet in the periodic flow based on the data packet size, the ACK arrival time, and the data packet generation interval included in the header.

Here, because the arrival time of the next data packet may fluctuate due to the random backoff process, we consider an error margin around the risky duration.

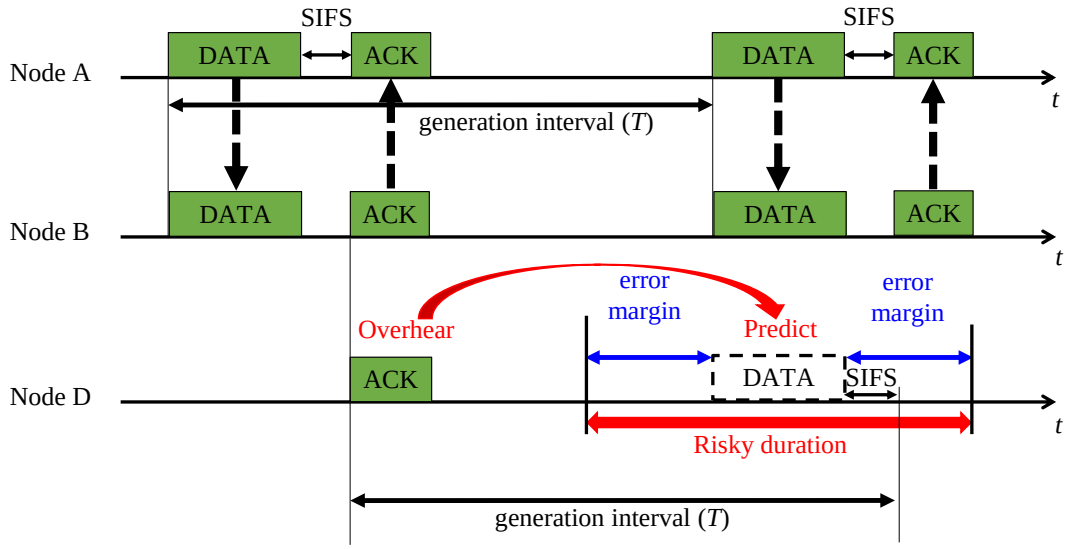


Figure 4.6: Prediction of a future risky duration.

Risky duration is calculated as follows. The notations used in this Chapter are listed in Table 4.2. By using t_{ACK_begin}, we can predict the t_{DATA_end} and t_{DATA_begin} of the next data packet as follows:

$$t_{DATA\_end} = t_{ACK\_begin} + T - D_{SIFS} \quad (4.1)$$

$$t_{DATA\_begin} = t_{DATA\_end} - \frac{packet\_size}{data\_rate}. \quad (4.2)$$

Here, the value of the predicted time of t_{DATA_begin} is the ideal case, and actually, it is fluctuated. We study the error of the predicted time t_{DATA_begin} in Sect.4.3.2.1. t_{DATA_begin} and t_{DATA_end} determine the risky duration. However, because of the random fluctuation, we add an error margin. Thus, risky duration can be represented as:

$$t_{risky\_begin} = t_{DATA\_begin} - m \quad (4.3)$$

$$t_{risky\_end} = t_{DATA\_end} + m. \quad (4.4)$$

Table 4.2: Notations used in Chapter

t_{ACK_begin}	Time stamp at which the ACK starts to be transferred.
T	Packet generation interval of the data flow.
t_{DATA_begin}	Time stamp at which the data packet starts to be transferred.
t_{DATA_end}	Time stamp at which the data packet finishes to be transferred.
D_{SIFS}	Duration of SIFS (Short Interframe Space), a constant value.
t_{risky_begin}	Time stamp at which the the risky duration starts.
t_{risky_end}	Time stamp at which the the risky duration ends.
m	The error margin for risky duration.
$packet\_size$	The data packet size.
$data\_rate$	The channel data rate.

The selection of m has a significant bearing on the performance of the proposed method. We discuss the optimization of the error margin in Sect. 4.3.2.

4.2.4 Avoidance of packet transmission in risky duration

In the risky duration, it is highly probable that packet transmission from the corresponding hidden node occurs. Thus, to avoid the collision, the node prohibits its packet transmission and waits until the risky duration ends.

Figure 4.7 shows the transmission avoidance process. By checking its own record, each node finds the nearest risky duration. If the duration from the current time to the nearest risky duration is larger than the time to transfer a data packet (Case 1), the node will begin the transmission process. Otherwise (Case 2), the node will hold on until the end of the

risky duration and repeats the process.

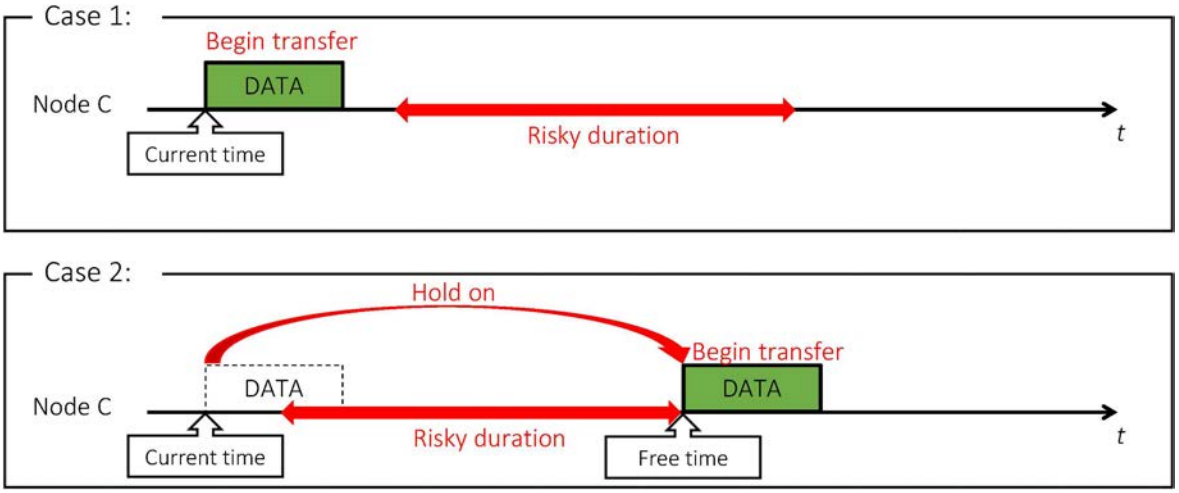


Figure 4.7: Transmission avoidance in the risky duration.

4.2.5 Computation complexity

To find the nearest risky duration, each node searches a minimum value in its record. Thus, the computation complexity will be O(N), where N is the number of periodic flows detected by the node. This complexity could even be reduced to O(log(N)) if we do the sorting before searching for the entry. Therefore, the complexity is not heavy.

4.2.6 Flowchart of the proposed method

Figure 4.8 summarizes the flowchart of the proposed system.

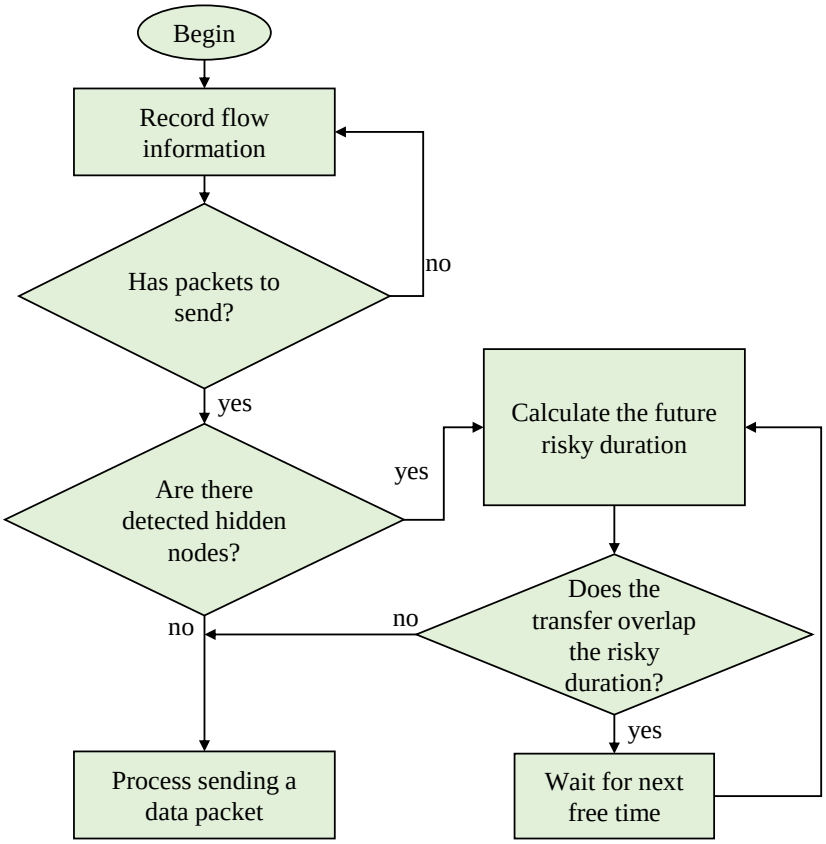


Figure 4.8: Flowchart of the proposed method.

4.3 Parameters tune-up for dynamic control and resultant performances

In order to evaluate the performance of the proposed method, we compare it against three related methods—HSD (Hidden Station Detection) [7], BDM (Binary Division Method) [13] and IEEE 802.11 DCF. HSD is a MAC layer approach while BDM is an application layer approach. As for proposed method, two methods, whose value of the error margin m is set based on the schemes discussed in Sects. 4.3.2.2 and 4.3.2.3 respectively, are defined. The former proposed method is called PHTS (Prediction of Hidden Transfer with Static error margin), and the latter proposed method is called PHTA (Prediction of Hidden Transfer

with Adaptive error margin).

4.3.1 Simulation environment

Table 4.3: Environment details

Name	Value
Packet size	512 bytes
Physical layer	IEEE 802.11b
Radio propagation range	32 m
MAC layer	IEEE 802.11
Retransmission limit	2
Routing	Static routing

We used QualNet v5.2 [38] for the simulations. The configuration of the simulation environment is described in Table 4.3. We evaluated the performance of the proposed method by measuring the packet loss rate in a multi-hop transmission environment. In this environment, all sensor nodes were deployed randomly in a 200 m $\times$ 200 m square area; the radio propagation range of each node was set to 32 m. A sink was located at the center of the square. The packet generation interval of each node was chosen from 1 s, 2 s, 3 s, or 4 s. The intervals of the first, second, third, and fourth nodes were 1 s, 2 s, 3 s, and 4 s, respectively. Then, the configuration was repeated (i.e., the fifth and sixth are 1 s and 2 s, respectively). Each node begins transmitting a data packet at a random time between 0 s and 5 s, and subsequently sends packets at a fixed interval. The number of nodes is incrementally increased by 10 nodes, from 20 to 100 nodes. Actually, after randomizing the nodes' location, the number of hops from nodes to the sink is from 1 to 6 hops.

4.3.2 Discussion for optimizing error margin for risky duration

The error margin (m defined in Sect. 4.2.3) is a very important parameter in our proposed method. If m is too large, it wastes the available channel resource and increases the end-to-end delay. On the other hand, if m is too small, it cannot cover the error of the predicted time t_{DATA_begin} defined in Sect.4.2.3 and avoid collisions by hidden nodes. In this subsection, we discuss the optimization of the error margin.

4.3.2.1 Survey of error of predicted time

The error (ε) of predicted time of t_{DATA_begin} defined in Sect. 4.2.3 can be determined by

$$\varepsilon = t_{en,k+1} - t_{en,k}, \tag{4.5}$$

in which $t_{en,k+1}$ and $t_{en,k}$ are the end-to-end delays of k^{th} packet and $(k+1)^{th}$ packet in a same flow, respectively. End-to-end delay includes backoff delay (t_{bo}) required to wait for backoff time, transmission delay (t_{tr}), propagation delay (t_{pr}), and congestion delay (t_{co}) that includes queuing delay and the delay to wait for channel to be idle. Another variable n_h is defined as the number of hops from the source node to the sink. With these variables, we can calculate ε as follows,

$$\varepsilon = \sum_{i=1}^{n_h}(t_{bo,k+1} + t_{tr,k+1} + t_{pr,k+1} + t_{co,k+1}) - \sum_{i=1}^{n_h}(t_{bo,k} + t_{tr,k} + t_{pr,k} + t_{co,k}). \tag{4.6}$$

Because the propagation delay and transmission delay are constant, we obtain

$$\varepsilon = \sum_{i=1}^{n_h}(t_{bo,k+1} + t_{co,k+1}) - \sum_{i=1}^{n_h}(t_{bo,k} + t_{co,k}). \tag{4.7}$$

The backoff delay is random, but its average value is constant. Because the transmission durations of k^{th} packet and $(k+1)^{th}$ packet is near, the congestion delay can be considered as constant approximately. Thus, we expect that ε value is random and its average value is zero. By running basic simulation under the same parameters as described in Sect. 4.3.1, we obtained statistics (as shown in Fig.4.9) of ε of a source node which is connected by one hop from the sink. We could expect, if the number of hops is n_h, statistics of ε have the same shape with Fig.4.9 but are multiplied by n_h. From Fig.4.9, in this environment setting, the 1 ms error margin for each hop could cover most of ε.

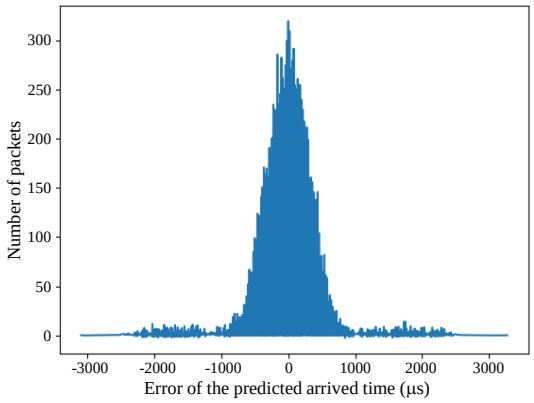


Figure 4.9: The error of the predicted arrived time in one hop.

4.3.2.2 Static setting of error margin

The first approach to determine m is to use a constant error margin for all nodes. Static setting of m has the advantage of lower computational cost at each sensor node. We find the optimized static error margin by applying the different value of error margin to our system and getting PLR (Packet Loss Rate) results as shown in Fig.4.10. From Fig.4.10, we conclude that the good error margin could be from 4 ms to 6 ms. Because larger error margin causes larger end-to-end delay, we choose 4 ms as the final value for static error margin.

By using the statistic result in Sect. 4.3.2.1 and the information that the hops number is from 1 to 6 hops, we see that the result of static error margin is reasonable.

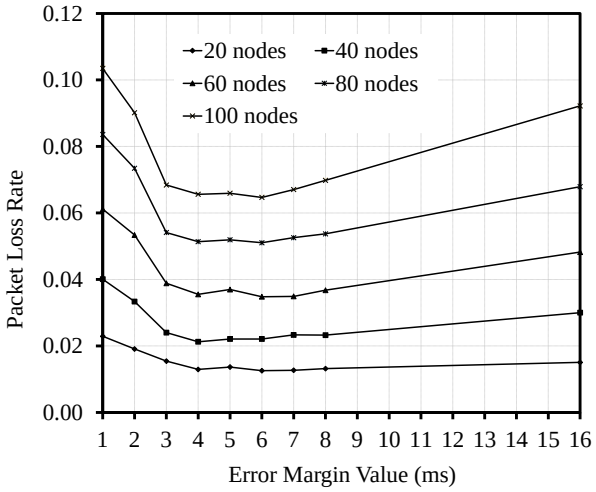


Figure 4.10: Packet loss rates for different error margins.

4.3.2.3 Dynamic setting of the error margin

In our system, because each data flow has different transfer route, static m may not always be suitable for all data flows. Furthermore, the congestion of the system could change overtime. Thus, m should be adapted to the situation if each sensor node has remaining computational resource.

We propose a dynamic setting method of m by following AIMD (Additive Increase Multiplicative Decrease) mechanism with some modifications. First, m is set to a initial value, which is m_{min}, at all flows. Next, m of each flow is updated when a new ACK of this flow is received. The new m can be calculated by

$$m_{i+1} = \begin{cases} \max(m_i - a, m_{min}) & \text{if } m_i \geqslant \varepsilon \\ \min(m_i \cdot b, m_{max}) & \text{if } m_i < \varepsilon \text{ or packet is lost} \end{cases}. \tag{4.8}$$

Here, m_{min} and m_{max} are the minimum bound and maximum bound of m respectively.

Packet loss is determined if the time from the last received ACK to the current time is larger than 1.5 times of flow interval.

We will find the values of a, b, m_{min}, and m_{max} that minimize the PLR by the following recursion process.

- [Step 1] We set initial values to a, b.
- [Step 2] By using these a, b values, we run the simulation many times with different m_{min} and m_{max}, and determine the m_{min} and m_{max} that minimized the PLR.
- [Step 3] We keep the m_{min} and m_{max} found in [Step 2] and run the simulation many times with different a and b, and determine the a and b that minimized the PLR. Go to [Step 2].

The process is finished when a, b, m_{min} and m_{max} are judged to be converged.

The actual result is described as following. First, we set DIFS duration (50 μs) as the initial value of a. b is initially set to 2, which is usually used in AIMD. By running the simulation, we obtained the result in Fig.4.11. From the evaluation in Fig.12, we conclude that the PLR is minimized when m_{min} and m_{max} are set to 1000 μs and 10000 μs, respectively.

Next, by using $m_{min} = 1000$ μs and $m_{max} = 10000$ μs, we obtained the result in Fig.4.12 which evaluates the values of a and b. From the evaluation in Fig.4.12, we conclude that there are two optimized points which are $a = 20$ μs, $b = 2$ and $a = 10$ μs, $b = 1.6$. We choose the $a = 20$ μs, $b = 2$ because the calculation cost is smaller and $b = 2$ is more frequently used in AIMD.

Finally, by using $a = 20$ μs, $b = 2$, we obtained the result in Fig.4.13. As shown in Fig.4.13, PLR is minimized when $m_{min} = 1000$ μs and $m_{max} = 10000$ μs, which is the same as the result of Fig.4.11. Therefore, the optimized values of a, b, m_{min} and m_{max} are

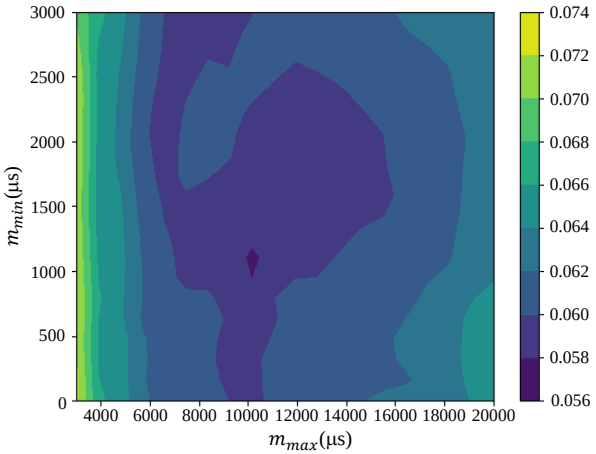


Figure 4.11: Evaluation result for optimizing m_{min} and m_{max}.

determined to be 20 μs, 2, 1000 μs and 10000 μs respectively. We will use these values for the following simulation.

4.3.3 Packet loss rate comparison

Figures 4.14 and 4.15 show the results of PLRs from the viewpoints of MAC layer approach and end-to-end (application layer) one, respectively.

We compare four MAC layer protocols; the proposed methods (PHTS and PHTA), HSD [7], and IEEE 802.11 DCF. The results presented in Fig.4.14 show that HSD contributes slightly to the performance improvement of DCF. HSD uses the RTS/CTS mechanism, which tackles the hidden node problem; however, it leads to significantly overloaded control packets. On the other hand, PHTS reduces the PLR by 73% (on average) in comparison with HSD because it avoids collisions due to hidden nodes and does not create any control packets. Finally, because of more flexible error margins, PHTA further reduces PLR by 14% in comparison with PHTS.

Figure 4.15 compares PHTA (MAC layer approach) with BDM (application layer approach) and DCF. PHTA (MAC layer approach) can be cooperated with BDM (application

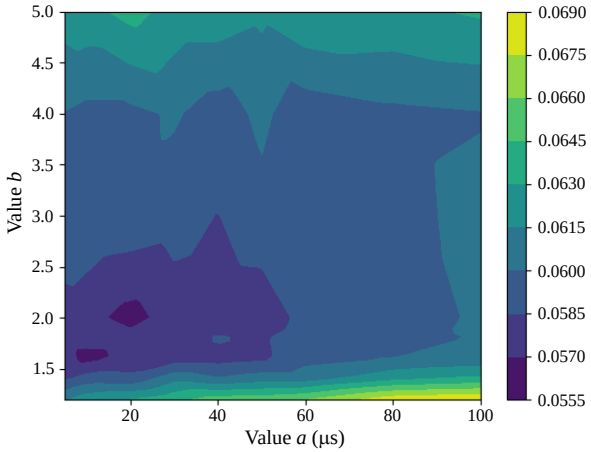


Figure 4.12: Evaluation result for optimizing a and b.

approach), and the cooperated method (PHTA + BDM) is also evaluated in Fig.16. PHTA and BDM improve the performance of naive DCF. Furthermore, the cooperation of PHTA and BDM can be regarded as a hybrid proposal. Because of the inherent advantages of both methods, the hybrid proposal (PHTA + BDM) exhibits the best performance and reduces the packet loss rate by 92% (on average) when compared with naive DCF.

4.3.4 Comparison of fairness of throughput ratio

Here, we compare four MAC layer protocols; the proposed methods (PHTS and PHTA), HSD, and IEEE 802.11 DCF. We use Jain's Fairness Index to compare the fairness [47]. Jain's Index is described as following:

$$f(X) = \frac{[\sum_{i=1}^{n} x_i]^2}{\sum_{i=1}^{n} {x_i}^2}, \tag{4.9}$$

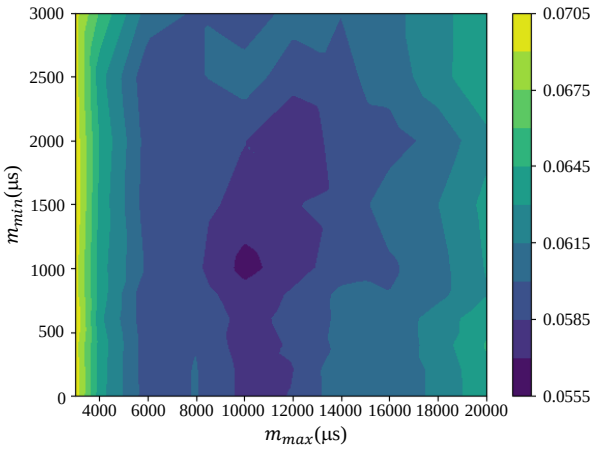


Figure 4.13: Optimized max and min margin recursion.

where, n is the number of nodes, and x_i is the ratio between actual throughput ($throughput_i$) and ideal throughput ($throughputMax_i$) of i^{th} node,

$$x_i = \frac{throughput_i}{throughputMax_i}. \tag{4.10}$$

The results in Fig.4.16 indicate that HSD improves the fairness of the throughput ratio among nodes by addressing the hidden node problem. However, PSTA has the best fairness index because it simultaneously addresses the hidden node problem while most effectively reducing PLR.

As for the comparison to the end-to-end approaches, as shown in Fig.4.17, both BDM and the proposed method show significant improvement in the fairness index. In general, the fairness index can also be improved by reducing PLR. However, in the range from 80 to 100 nodes, when the PLR becomes relatively large (as shown in Fig.4.15), the proposed method achieves better fairness index. On the other hand, the hybrid method (PHTA + BDM) exhibits the best fairness index because of the inherent advantages of the proposed method and BDM.

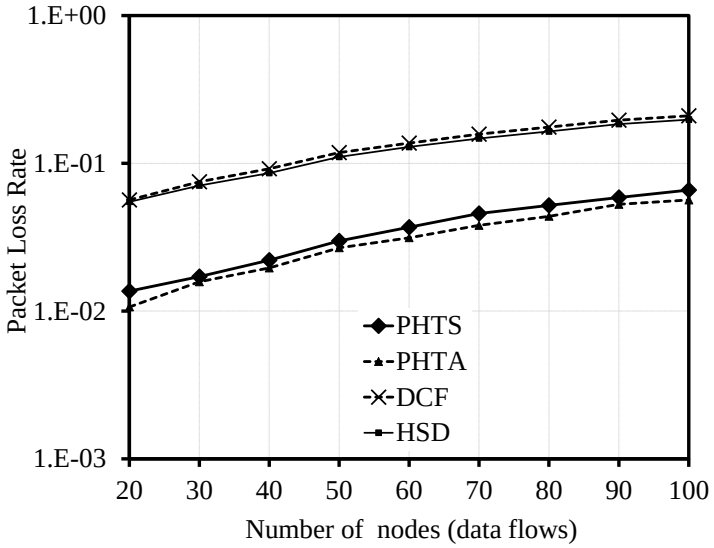


Figure 4.14: Packet loss rate (compared with MAC layer approaches).

4.3.5 Comparison of end-to-end delay

Our proposed methods reduce hidden collisions by making nodes hold on their transmission in the risky time. Therefore, we guess that the end-to-end delay will be increased.

To evaluate this trade-off, we usually use the end-to-end delay as a metric. However, the end-to-end delay is strongly affected by the number of hops (n_h) and each flow has very different n_h. Therefore, we use the delay per hop (t_{dh}) which can be calculated by

$$t_{dh} = \frac{t_{en}}{n_h} \tag{4.11}$$

to evaluate this problem.

Figure 4.18 shows the comparison of the delay per hop among our proposed methods (PHTS, PHTA, and the hybrid PHTA+BDM), BDM, HSD and IEEE 802.11 DCF. We could observe that, when number of nodes is small, the difference among the methods is small because there are few risky durations and the nodes rarely have to hold on their transmission.

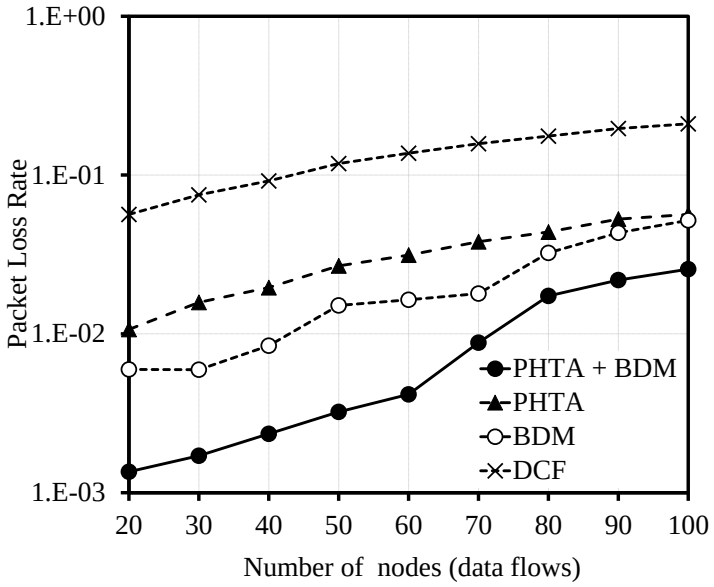


Figure 4.15: Packet loss rate (effect of hybrid approach).

Furthermore, when the number of nodes is larger, the delay of our proposed methods is also larger in comparison with the others. At 100 nodes, each delay per hop of PHTS, PHTA, and PHTA+BDM increases by about 2.3 ms, 3.6 ms, and 1.9 ms in comparison with the best delay (BDM), respectively. However, in comparison with the actual application data generation intervals which are usually in order of seconds or minutes, this increased delay is negligible. Thus, it is reasonable for the trade-off of these increasing of delay per hop with the improvement of the PLR and fairness.

4.3.6 Survey of the retransmission parameter

Here, we conduct simulations with different retransmission limits to observe the performance differences of the different parameter settings in the proposed methods and the related ones. This is important because some methods may work only in a specific situation.

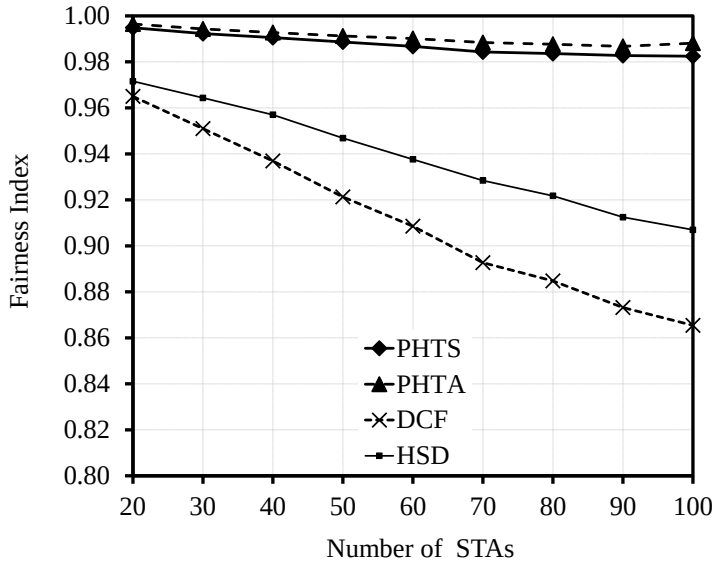


Figure 4.16: Fairness index of throughput ratio (compared with MAC layer approaches).

4.3.6.1 Simulation settings

For comparison, we use three different settings. In the first setting, the retransmission limit is set to 2, an intentionally smaller value because of the lower latency and simpler nodal device requirement (Retransmit 2). This settings were used as default setting in this thesis. In the second setting, we increase the retransmission limit to 4 (Retransmit 4). In the last setting, the retransmission limit is set to 7, which is the default value of IEEE 802.11b standard.

To concentrate on the most complex situation, we use the multi-hop environment with 100 nodes. The other parameters are described in Sect. 4.3.1.

4.3.6.2 Effect of retransmission parameter on packet loss rate and delay per hop

Figure 4.19 shows the packet loss rate comparison. When the retransmission limit is increased, PLR decreases in all the methods. Our proposed methods get the larger benefit because the retransmitted packet's transmission time is next to its initial transmission time

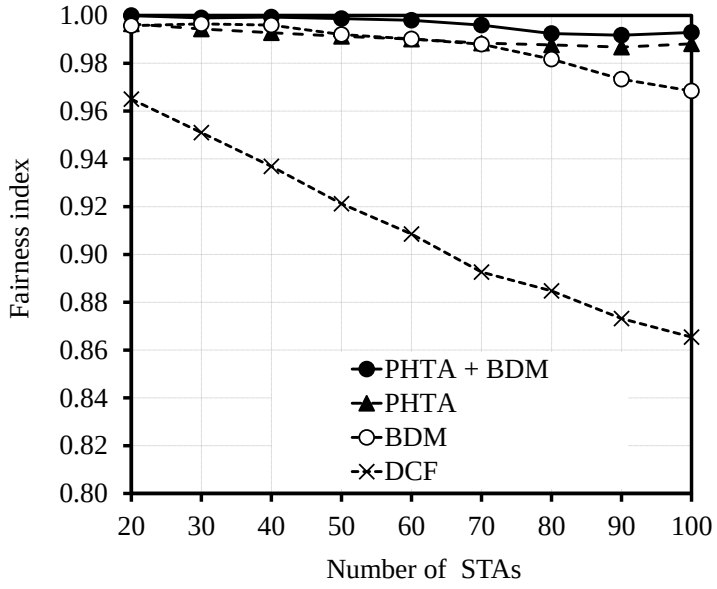


Figure 4.17: Fairness index of throughput ratio (effect of hybrid approach).

and is usually and partially protected by the risky duration mechanism. In all the three settings, our final proposal, BDM+PHTA, always shows the best performance.

Next, Fig. 4.20 presents the delay per hop comparison. When the retransmission limit is increased, the delay per hop is also increased. PHTA has the largest delay per hop because of the larger risky duration, which prevents the neighbor nodes from sending packets.

On the other hand, the cooperated method (PHTA + BDM) does not increase the delay per hop much in comparison with the others because it inherits the congestion reduction ability of BDM as the application layer approach.

In these results, the advantages of the cooperated method are shown. Furthermore, if the simpler nodal device and lower delay requirement are not considered, setting retransmission limit to 7 could be the better choice.

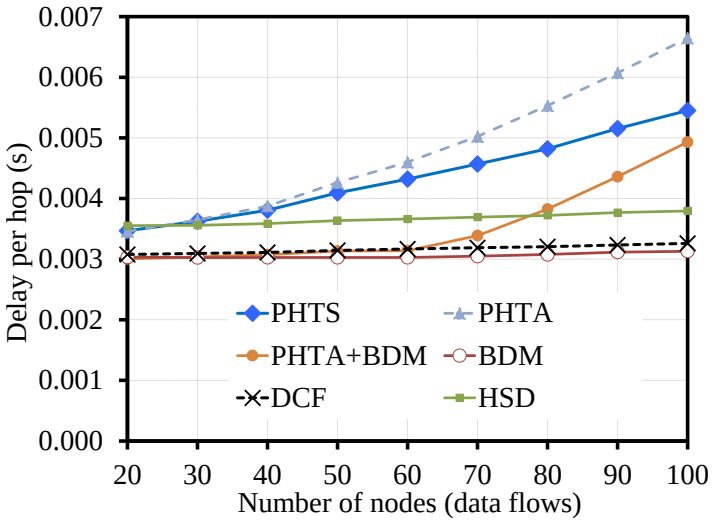


Figure 4.18: Delay per hop comparison.

4.3.7 Comparison of different network topologies

In this section, we apply the proposed methods on different topologies to survey the effect of our proposal in various situations.

4.3.7.1 Simulation settings

The first topology (Random in square, Fig. 4.21a) is already described in Sect. 4.3.1. In this topology, the number of hops between each node and the sink is from 1 to 6 hops.

In the second topology (Uniform random, Fig. 4.21b), A 200 m x 200 m square is divided into small 100 cells whose size is 20 m x 20 m. The sink is placed at one of the corners and each cells has one node at random location inside the cell. The transfer range in this case is 24 m. The number of hops between each node and the sink in this topology is from 1 to 10.

In the last topology (One hop with hidden nodes, Fig. 4.21c), the nodes' locations are similar to the first case. The transfer range is increased to 142 m. Thus, all the nodes transfer packets to the sink in one hop manner. In this case, there are some hidden nodes if

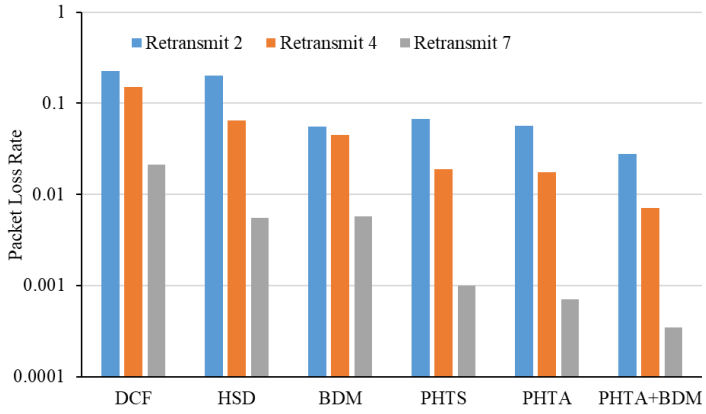


Figure 4.19: Packet loss rate with different retransmission limits.

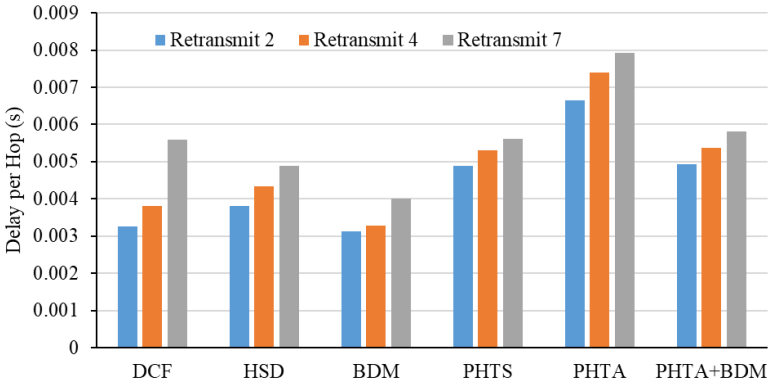


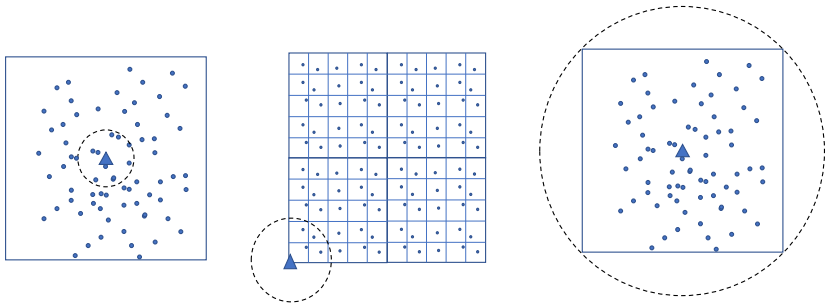
Figure 4.20: Delay per hop with different retransmission limits.

the nodes are located in the opposite sides.

To concentrate on the most complex situation, the number of nodes is always set to 100. The other parameters are described in Sect. 4.3.1.

4.3.7.2 Effect of network topologies on packet loss rate

Figure 4.22 shows the packet loss rate comparison. HSD, the traditional RTS/CTS approach, shows the good PLR only in the simple topology case (one hop with hidden nodes). In all the three topologies, our final proposal, PHTA + BDM, always shows the best performance.



a. Random in square b. Uniform random c. One hop with hidden nodes

Figure 4.21: Network topologies in this evaluation.

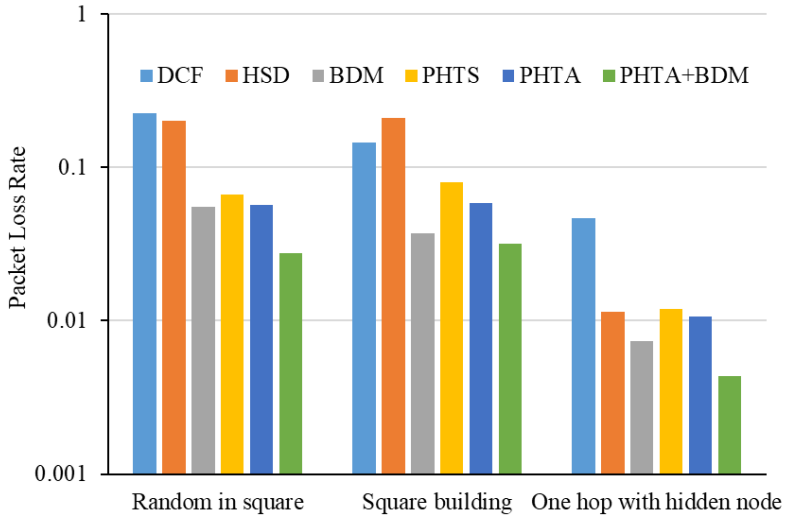


Figure 4.22: Packet loss rate with different topologies.

This result shows that our methods could work effectively in various typologies.

4.3.8 Comparison of different combinations

In this Chapter, because prioritize the simple implementation, we use BDM as application layer shift-time method to combine with PHTA. However, if the performance is the most important factor, the more effected shift-time method, CSM, will be chosen. In this section, we show the difference in PLR of the PHTA when it combines with BDM and CSM.

Figure 4.23 shows that PHTA+CSM is the best combination to reduce the PLR. Specifically, PHTA+BDM and PHTA+CSM reduce PHTA's PLR by 51% and 82% respectively. CSM could improve the PHTA's performance more than BDM because CSM solves the contention problem of the periodical flows more effective.

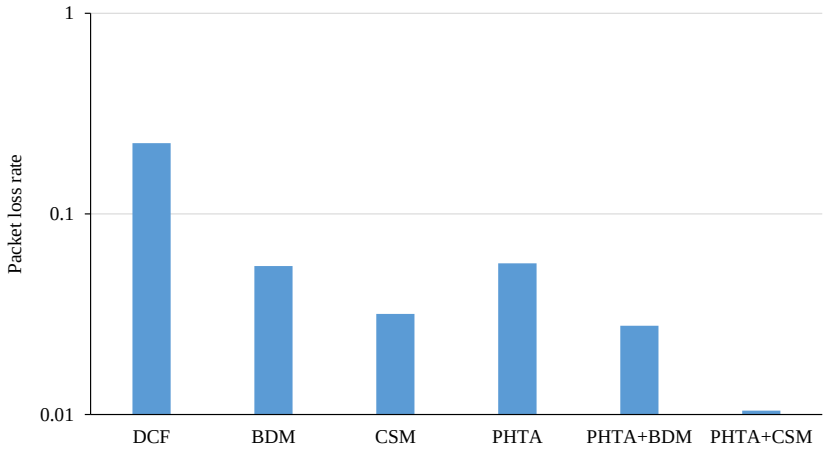


Figure 4.23: Packet loss rate with different combinations.

Chapter 5

Conclusion

In this thesis, we focused on improving the performance of heterogeneous periodic WSNs. The first problem we challenge was the continual packet collisions among different periodic flows. In WSNs treating periodic flows, the inherent problem of repeated packet collisions cannot be completely resolved by conventional protocols.

To solve this problem, firstly, we proposed an application layer shift-time method, BDM. BDM is easily implemented and shows the good performance when the number of nodes is small.

In the next step, we designed a new formula to address the contention problem in heterogeneous periodic flows system. Based on this formula, we proposed protocols that schedule packet creation timing at each source node. The basic proposed scheduling method, CSM, is simpler and shows better performance in short run applications.

The extended proposed method with rescheduling function, called CSMR. CSMR is best for long run applications because this method continuously updates the shift-time of the node with highest number of lost packets. Therefore, the system's performance will be improved over time. This property was preserved in the multi-hop network environment.

In conclusion, the proposed methods satisfy the conditions that only the sink calculates

the appropriate packet generation timing for all source nodes and that no modification of the MAC layer function is required.

In the second problem, we focused on compounded negative effect of the hidden node problem and the continuous collision problem among periodic data flows. To the best of our knowledge, this thesis firstly considers the compounded effect of hidden nodes and continuous collisions among periodic data flows. The compounded effect not only greatly increases packet loss rate, but also decreases the fairness among data flows.

Therefore, we proposed a novel MAC layer mechanism, PHT, to detect, predict, and avoid future collisions. PHT could be implemented by two ways, PHTS with the static error margin and PHTA with the adapted error. PHTS is simpler and requires less computation but PHTA has the better performance. Through simulations, we demonstrated the effectiveness of reducing packet loss rate drastically and improving the fairness among sensor nodes in the proposed method.

We also showed that PHT could be combined with application methods, BDM and CSM, to achieve even greater improvement. The combination of PHTA and CSM showed the best result in reducing PLR.

In our future work, we will implement the proposed methods on actual sensor nodes and evaluate the performance. Another future work includes considering the application of the proposed method to IEEE 802.15.4 networks.

Bibliography

[1] A. Alaiad and L. Zhou, "Patients' adoption of WSN-Based smart home healthcare systems: An integrated model of facilitators and barriers," *IEEE Transactions on Professional Communication*, vol. 60, no. 1, pp. 4–23, March 2017. doi:10.1109/TPC.2016.2632822

[2] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart grid - The new and improved power grid: A survey," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, pp. 944–980, Oct. 2012.

[3] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A survey on smart grid communication infrastructures: Motivations, requirements and challenges," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 5–20, Feb. 2013.

[4] X. Cao, J. Chen, Y. Cheng, X. S. Shen, and Y. Sun, "An analytical MAC model for IEEE 802.15.4 enabled wireless networks with periodic traffic," *IEEE Transactions on Wireless Communications*, vol. 14, no. 10, pp. 5261–5273, Oct. 2015. doi:10.1109/TWC.2015.2435006

[5] A. B. Noel, A. Abdaoui, T. Elfouly, M. H. Ahmed, A. Badawy, and M. S. Shehata, "Structural health monitoring using wireless sensor networks: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1403–1423, thirdquarter

2017.
doi:10.1109/COMST.2017.2691551

[6] M. Kim and C.-H. Choi, "Hidden-node detection in ieee 802.11n wireless lans," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 6, pp. 2724–2734, Jul. 2013.

[7] Y. Kim, J. Yu, S. Choi, and J. Kyunghun, "A novel hidden station detection mechanism in ieee 802.11 wlan," *IEEE Communications Letters*, vol. 10, no. 8, pp. 608–610, Aug. 2006.

[8] H. Ma, J. Zhu, and S. Roy, "On loss differentiation for csma-based dense wireless network," *IEEE Communications Letters*, vol. 11, no. 11, Nov. 2007.

[9] E. C. Eze, S. Zhang, E. Liu, E. N. Nweso, and J. C. Eze, "Timely and reliable packets delivery over internet of vehicles for road accidents prevention: a cross-layer approach," *IET Networks*, vol. 5, no. 5, pp. 127–135, Sep. 2016.

[10] S. Shin and J. M. Chung, "Performance analysis of packet errors on ieee 802.11-based dcf and edcf," *Electronics Letters*, vol. 51, no. 2, pp. 187–188, Jan. 2015.

[11] P. M. van de Ven, A. J. E. M. Janssen, and J. S. H. van Leeuwaarden, "Balancing exposed and hidden nodes in linear wireless networks," *IEEE/ACM Transactions on Networking*, vol. 22, no. 5, pp. 1429–1443, Oct. 2014.

[12] J. Y. Um, J. S. Ahn, and K. W. Lee, "Evaluation of the effects of a grouping algorithm on ieee 802.15.4 networks with hidden nodes," *Journal of Communications and Networks*, vol. 16, no. 1, pp. 81–91, Feb. 2014.

[13] N. Anh Huy, Y. Tanigawa, and H. Tode, "Scheduling methods to improve the performance of heterogeneous periodic flows in wireless sensor networks," in *Proc. 2017 IEEE 42nd Conference on Local Computer Networks (LCN)*, Oct. 2017.

[14] N. Anh Huy, Y. Tanigawa, and H. Tode, "Scheduling method for solving successive contentions of heterogeneous periodic flows based on mathematical formulation in multi-hop wsns," *IEEE Sensors Journal*, pp. 1–1, 2018.
doi:10.1109/JSEN.2018.2868327

[15] N. Anh Huy, Y. Tanigawa, and H. Tode, "Channel access control for collisions caused by hidden nodes and phase synchronization among periodic data flows," in *Proc. 2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, Jan. 2019.

[16] S. Du, A. K. Saha, and D. B. Johnson, "RMAC: A routing-enhanced duty-cycle MAC protocol for wireless sensor networks," in *Proc. IEEE INFOCOM 2007*, pp. 1478–1486, May 2007.

[17] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. IEEE INFOCOM 2002*, vol. 3, pp. 1567–1576, 2002.

[18] 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 (TON)*, vol. 16, no. 3, pp. 511–524, Jun. 2008.

[19] H. Harb, A. Makhoul, R. Couturier, and M. Medlej, "ATP: An aggregation and transmission protocol for conserving energy in periodic sensor networks," in *Proc. 2015 IEEE 24th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 134–139, Jun. 2015.
doi:10.1109/WETICE.2015.9

[20] S. Liu, M. Fardad, E. Masazade, and P. K. Varshney, "Optimal periodic sensor scheduling in networks of dynamical systems," *IEEE Transactions on Signal Processing*, vol. 62,

no. 12, pp. 3055–3068, Jun. 2014.
doi:10.1109/TSP.2014.2320455

[21] L. Shi, P. Cheng, and J. Chen, "Optimal periodic sensor scheduling with limited resources," *IEEE Transactions on Automatic Control*, vol. 56, no. 9, pp. 2190–2195, Sep. 2011.
doi:10.1109/TAC.2011.2152210

[22] O. Rottenstreich, M. D. Francesco, and Y. Revah, "Perfectly periodic scheduling of collective data streams," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1332–1346, Jun. 2017.
doi:10.1109/TNET.2016.2629092

[23] R. Geng, Z. Li, and L. Song, "AQMP: An adaptive QoS MAC protocol based on IEEE802.11 in Ad Hoc networks," in *Proc. WiCom 2009*, pp. 1–4, Sep. 2009.

[24] Y. Yang and R. Kravets, "Distributed QoS guarantees for realtime traffic in Ad Hoc networks," in *Proc. IEEE SECON 2004*, pp. 118–127, 2004.

[25] A. Veres, A. T. Campbell, M. Barry, and L. H. Sun, "Supporting service differentiation in wireless packet networks using distributed control," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 10, pp. 2081–2093, Oct. 2001.

[26] R. Ranganathan and K. Kannan, "Enhancing the selection of backoff interval using fuzzy logic over wireless Ad Hoc networks," *Scientific World Journal*, pp. 680–681, 2015.
doi:10.1155/2015/680681

[27] B. Kang, S. Myoung, and H. Choo, "Distributed degree-based link scheduling for collision avoidance in wireless sensor networks," *IEEE Access*, vol. 4, pp. 7452–7468,

2016.
doi:10.1109/ACCESS.2016.2622720

[28] M. Azeem, M. I. Khan, S. U. Khan, and W. Gansterer, "Efficient scheduling of sporadic tasks for real-time wireless sensor networks," *IET Wireless Sensor Systems*, vol. 5, no. 1, pp. 1–10, 2015.
doi:10.1049/iet-wss.2013.0065

[29] D. Buranapanichkit and Y. Andreopoulos, "Distributed time division multiple access protocol for multi-hop wireless sensor networks," in *Proc. TENCON 2015 - 2015 IEEE Region 10 Conference*, pp. 1–4, Nov. 2015.
doi:10.1109/TENCON.2015.7372751

[30] C. Muhlberger, "On the pitfalls of desynchronization in multi-hop topologies," in *Proceedings of the 2nd International Conference on Sensor Networks*, pp. 99–108, 2013.
doi:10.5220/0004230900990108

[31] B. N. Hang Yu and W. K. Seah, "Pulse arrival scheduling for nanonetworks under limited IoT access bandwidth," in *2017 IEEE 42nd Conference on Local Computer Networks (LCN)*, Oct. 2017.

[32] T. W. Carley, M. A. Ba, R. Barua, and D. B. Stewart, "Contention-free periodic message scheduler medium access control in wireless sensor/actuator networks," in *Proc. IEEE RTSS 2003*, pp. 298–307, Dec. 2003.
doi:10.1109/REAL.2003.1253276

[33] T. Gaugel, J. Mittag, H. Hartenstein, S. Papanastasiou, and E. G. Strom, "In-depth analysis and evaluation of self-organizing TDMA," in *Proc. 2013 IEEE Vehicular*

Networking Conference, pp. 79–86, Dec. 2013.
doi:10.1109/VNC.2013.6737593

[34] C. Feng, Y. Jiang, C. Jun, and T. Xin, "Dynamic scheduling management for periodic real-time traffic," in *Proc. 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC)*, pp. 2039–2042, Dec. 2013.
doi:10.1109/MEC.2013.6885386

[35] I. Jawhar and J. Wu, "QoS support in TDMA-based mobile Ad Hoc networks," *Journal of Computer Science and Technology*, vol. 20, no. 6, pp. 797–810, 2005.

[36] A. P. Shrestha, S. J. Yoo, H. J. Choi, and K. S. Kwak, "Enhanced rate division multiple access for electromagnetic nanonetworks," *IEEE Sensors Journal*, vol. 16, no. 19, pp. 7287–7296, Oct. 2016.
doi:10.1109/JSEN.2016.2598579

[37] H. Domura, Y. Tanigawa, and H. Tode, "Dynamic time-shift scheduling of periodical traffic in wireless sensor network," in *Proc. IEEE SECON 2014*, pp. 182–184, Jun. 2014.

[38] "Qualnet simulator version 5.2," http://www.scalable-networks.com, Scalable Network Technologies.

[39] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. of the 33rd Annual Hawaii International Conference on System Sciences*, vol. 8, Jan. 2000.
doi:10.1109/HICSS.2000.926982

[40] C. E. Campbell, K. Loo, H. A. Kurdi, and S. Khan, "Comparison of ieee 802.11 and ieee 802.15.4 for future green multichannel multi-radio wireless sensor networks,"

International Journal of Communication Networks and Information Security (IJCNIS), vol. 3, no. 1, 2011.

[41] T. Pei, F. Lei, Z. Li, G. Zhu, X. Peng, Y. Choi, and H. Sekiya, "A delay-aware congestion control protocol for wireless sensor networks," *Chinese Journal of Electronics*, vol. 26, no. 3, pp. 591–599, 2017.

[42] J. K. T. Ha and J. M. Chung, "He-mac: Harvest-then-transmit based modified edcf mac protocol for wireless powered sensor networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 3–16, Jan. 2018.

[43] H. Gharavi and B. Hu, "Cooperative diversity routing and transmission for wireless sensor networks," *IET Wireless Sensor Systems*, vol. 3, no. 4, pp. 277–288, Dec. 2013.

[44] K. S. A. Kumar and R. Balakrishna, "Comparative analysis of delay and throughput using ieee 802.11 and receiver centric-mac protocol in wireless sensor networks," in *International Conference on Innovations in Power and Advanced Computing Technologies (i-PACT)*, Apr. 2017.

[45] "Esp8266 community," https://www.esp8266.com, espressif Systems.

[46] Y. T. A. H. Nguyen and H. Tode, "Scheduling method for solving successive contentions of heterogeneous periodic flows based on mathematical formulation in multi-hop wsns," *IEEE Sensors Journal*, vol. 18, no. 21, pp. 9021-9033, Nov. 2018.

[47] H. Shi, R. V. Prasad, E. Onur, and I. G. M. M. Niemegeers, "Fairness in wireless networks:issues, measures and challenges," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 5-24, First Quarter 2014.