

Improving Multi-path Congestion Control for Event-driven Wireless Sensor Networks by using TDMA

Fang-Yie Leu*, Hsin-Liang Chen, and Chih-Chung Cheng
Department of Computer Science, TungHai University, Taiwan
leufy@thu.edu.tw, carl30412@gmail.com, s1003906@thu.edu.tw

Abstract

In this paper, we propose a time division method to solve the congestion control problem for an event-driven sensor network. In this method, a node adjusts time (therefore bandwidth) assigned to each upstream node by employing a dynamic Time Division Multiple Access (TDMA) approach. The purpose is to efficiently use resources available to the system and avoid the situation in which a node cannot send event information to the Base Station (BS) on time through the insufficiently allocated bandwidth when it detects multiple events in a short time span. Experimental results show that this method can effectively improve the throughputs of a Wireless Sensor Network (WSN), shorten its end-to-end delays and reduce its packet loss rates.

Keywords: Congestion Control, Event-driven, TDMA, Bandwidth, WSN.

1 Introduction

With the capability of sensing human activities and the environmental conditions, wireless sensor networks (WSNs), are now widely used in detecting environmental change or something changed. For example, they can be utilized in a hospital to monitor patient's health conditions [12] and determine his/her treatment needs, in supermarkets for surveillance of products placement and preventing them from being stolen [2] [11], and in our living environment for monitoring whether a particular region is on fire [2], flooded [2] or having mudslide [2], etc. Because wireless sensing devices need to be adapted to their surrounding landscape or embedded into hardware components or objects in order to avoid impacting the environment or occupying too much space, their volume must be reduced as much as possible. Thus, they are often equipped with power supply components of small volumes. If placed at locations without any ordinary power supply, they need to be powered by small batteries with limited capacity. As a result, a number of studies focus on how to effectively reduce energy consumption [3] [10], or how to improve system performance and lifetime of WSNs [4] [20] [17]. In a wireless sensor network, when the sensor node detects occurrence of an event, it will send packets to notify the Base Station (BS). Generally, all the nodes of the network form a tree structure, in which the BS is the root node and all the end nodes of a path from the BS are leaf nodes. When a node Z 's immediate upstream nodes $U_Z=U_{Z1}, U_{Z2}, \dots, U_{Zm}$ simultaneously send packets to Z , congestion and collision of packets may occur. Consequently some packets will be dropped, meaning the relevant information will be lost. In some systems, those dropped packets will be resent, often shortening lifetime of sensor nodes. In general, network congestion will reduce network throughput, and increase packet transmission delay. Therefore, how to prevent packets from collision and how to engage congestion control of network traffic have been a main research topic in WSN studies [19] [16]. Currently, Frequency Division Multiple Access (FDMA) [15] and Time Division Multiple Access (TDMA) [8] [1] are two common technologies used to solve network congestion

Journal of Internet Services and Information Security (JISIS), volume: 5, number: 4 (November 2015), pp. 1-19

*Corresponding author: Science Technology Building Room 425, Tunghai University, Taichung, Taiwan, Tel: +886-930-069-809, Web: <http://d1lab.csie.thu.edu.tw>

occurring on the transmission channel between U_{Zi} and Z , $i=1, 2, \dots, m$. In our previous work [13], we employed the FDMA method to control flow congestion. In that study, the bandwidth BW of the path between Z and its downstream nodes D_i is allocated to those nodes $U_{Z1}, U_{Z2}, \dots, U_{Zn}$, $n \leq m$, which have detected occurrence of events and are still transmitting packets. The amount of data sent to Z from each upstream node U_{Zi} is fixed. Even if an upstream node U_{Zi} detects another event, it still transmits packets for the formerly detected event. Also, when traffic U_{Zi} sent to Z is less than before, the bandwidth remains unchanged until the next bandwidth adjustment time period (BATP) starts. In other words, no matter how many events have been detected by U_{Zi} and its upstream nodes, the available data transmission rate allocated to U_{Zi} in a BATP is a constant. Therefore, when a node detects h events, $h > 1$, during a BATP, it may not be able to report them to BS in time with the constant transmission rate. When h is larger, the situation is worse. To solve this problem, one can use a dynamical bandwidth allocation scheme to dynamically adjust U_{Zi} 's bandwidth. In this scheme, when an upstream node U_{Zi} needs to send a hugely increased amount of data, Z can dynamically allocate a larger bandwidth to it, or reduce the bandwidth when the traffic U_{Zi} sends to Z is decreased. This can greatly enhance bandwidth utilization and reduce packet loss. On the other hand, TDMA methods have advantageous features, such as they can be easily re-programmed to adapt data traffic [6] [8] [1], and only a single frequency for data transmission channel is required, resulting in less susceptibility to outside interference [7]. Therefore, in this paper we propose a time division scheme, named TDMA-based Multi-path Congestion control System (TMCoS for short) to solve the congestion control problem for an event-driven sensor network. In this system, a node adjusts time (therefore bandwidth) assigned to each upstream node by using a dynamic TDMA approach to efficiently utilize available resources of the system and avoid the situation in which a node cannot send event information to the BS on time through the insufficiently allocated bandwidth when it detects multiple events in a short time span. To avoid packet collision in the TMCoS, the time among a node Z and its immediate upstream nodes, e.g., $U_{Z1}, U_{Z2}, \dots, U_{Zm}$, must be synchronized. After allocating bandwidth/time to one of its upstream nodes, e.g., U_{Zi} , Z must notify U_{Zi} of the time pair where is U_{Zi} 's transmission starting time and is the time duration in which U_{Zi} is allowed to transmit sensed data. Therefore, by time synchronization, all upstream nodes can be managed to send packets to Z one by one, thus preventing the case in which U_{Zi} starts sending packets to Z while Z is still receiving packets sent by U_{Zj} , $i \neq j$. This can effectively avoid packet collision. This paper is organized as follows. Chapter II describes the related work of this study. Chapter III introduces the proposed system. Experimental results are presented and analyzed in Chapter IV. Chapter V concludes this paper and summarizes our future studies.

2 Related Work

When nodes in a WSN simultaneously transmit packets to the same downstream node, congestion and packet collisions may occur, resulting in packet loss. This will consume extra energy to retransmit the lost packets [19]. So some proposed studies [16] [21] [5] to control packet transmission so as to avoid the occurrence of congestion and collision. Some congestion control methods [9] have been presented to extend lifetime of WSN systems. Patil et al. in [16] proposed a method called Priority-based Congestion Control Protocol (PCCP), in which a node is given a priority level in accordance with its privilege. When packet congestion occurs among nodes, nodes with higher priorities are allowed to send packets; and the privilege or the priority level of each node is adjusted after each packet transfer to optimize the congestion control. Zawodniok et al. in [21] proposed a predictive method for congestion control, which dynamically records the amount of traffic for each node and predicts nodes that might be soon congested. Hence, flow of these nodes will be divided and redirected to other nodes to relieve congestion. Farzaneh et al. in [5] proposed a resource control protocol, in which based on conditions of paths of a node, packets

are transmitted via paths with lower degree of congestion or more available resources. The condition of a path is reevaluated for every data transfer to reduce congestion. Maggs et al. [14] proposed a time synchronization method called Consensus Clock Synchronization (CCS), which synchronizes all nodes or devices within a system via a virtual consensus clock as a consistent synchronization object, thereby reducing clock errors among nodes caused by the geographical environment.

3 The Proposed Scheme

The proposed system consists of three phases: spanning tree generation, bandwidth/time allocation, and bandwidth/time adjustment. In the first phase, the BS broadcasts a tree-establishment packet P to connect all the nodes of a WSN into a single spanning tree, which is rooted at BS. All the outermost layer nodes with no upstream nodes are leaf nodes. Other nodes are intermediate nodes. The BS is directly connected to many nodes, which are called level-1 nodes and represented as S_i^1 , $1 \leq i \leq m$, where m is the total number of level-1 nodes. Each level-1 node S_i^1 is again the root node of a sub-tree. Those immediate upstream nodes of S_i^1 are called level-2 nodes of S_i^1 , denoted by S_{ij}^2 which is the j^{th} upstream node of S_i^1 , $1 \leq j \leq |S_i^1|$. In general, a level- j node's immediate upstream nodes are called level- $(j+1)^{th}$ nodes, $1 \leq j \leq k-1$, where k is the height of the spanning tree, defined as $k = \max(h(1), h(2), \dots, h(s))$ in which s is the number of leaf nodes of the tree and $h(i)$ is the path length of leaf node i from BS, $1 \leq i \leq s$. In other words, k is the number of link counts between BS and the farthest leaf node, e.g., f . In the second phase, the BS allocates time period/bandwidth to each level-1 node S_i^1 , $1 \leq i \leq m$. The bandwidth is proportional to the total number of nodes inside the sub-tree rooted at S_i^1 . Then each level-1 node S_i^1 allocates the bandwidth proportional to the total number of nodes in the sub-tree rooted at level-2 node, i.e., $|S_{ij}^2|$ to $|S_{ij}^2|$, $1 \leq j \leq n_i$, where n_i is the number of immediate upstream nodes of S_i^1 . We will define the allocated time period/bandwidth later. The third phase begins after the second phase in the situation when a sensor node having detected an event of the system starts sending data packets to its downstream node, and ends at the end of the life of the WSN. In this phase, whenever the data rate of a node is changed, bandwidth allocation and time period adjustment for each node will be triggered periodically.

3.1 Spanning tree generation phase

The phase begins when the system starts to work, and lasts until the spanning tree is generated under the assumption that all nodes of the system are reachable from BS so that they can be connected together to form a spanning tree and to send detected data to the BS along the links of the tree. The tree is generated in the following manner

OP_code=LREQ	SourceID	SenderID	BWdefault	Path
--------------	----------	----------	-----------	------

Figure 1: Format of an LREQ packet

OP_code=LACK	SenderID	ReceiverID
--------------	----------	------------

Figure 2: Format of an LACK packet.

First, BS broadcasts a Link Request (LREQ) packet. The format of this message, as shown in

Figure 1, includes five fields: OP_code, SourceID, SenderID, BWdefault, and Path, where OP_code = LREQ indicates that it is a Link Request packet, SourceID records the node which first issues the LREQ, SenderID represents ID of the node that sends the packet, BWdefault is the network-card bandwidth of the SenderID, and Path lists the nodes that the LREQ packet has gone through. Once the system is started, when a node, e.g., Z, first receives an LREQ packet from node S, it will establish a connection between it and S (and thus, SenderID = S). Afterwards, node Z replies S with a Link Acknowledgement (LACK) packet, the format of which as shown in Figure 2 consists of three fields: OP_code, SenderID, and ReceiverID, where OP_code = LACK shows that it is a link acknowledgement packet, SenderID represents ID of the sender (i.e., node Z), and ReceiverID shows ID of the receiver (i.e., S). Z will subsequently generate another LREQ packet, in which OP_code = LREQ, SourceID=S, SenderID = Z, BWdefault = bandwidth of the network card of Z, and Path = Z, and then broadcast this packet. Thereafter, if Z receives any other packet with OP_code = LREQ and SourceID = S, it discards the packet regardless of its sender. This procedure will be repeated until all nodes have at least received the LREQ packet once. In a WSN, the link between any two adjacent nodes forms a part of the path to transmit detected data from upstream nodes towards the BS. The time consumed in this phase can be calculated as $k*(T_{tr}+T_{pr}+T_{prt})$, where k is the height of the spanning tree, $T_{tr}(T_{rec})$ is the time required to send (receive) a packet and T_{pr} is the time consumed for processing the received packet and prepare the transmitted packet.

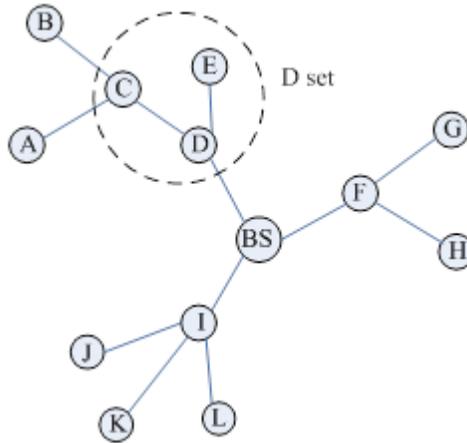


Figure 3: A case of a spanning tree.

Figure 3 shows an example of a spanning tree. When a node, e.g., D, first receives a LREQ packet, it establishes a relationship table, as shown in Table 1, in its memory to record the relationship between itself and other neighbor nodes; for example, the downstream node (i.e., the node that sends a packet with OP_code = LREQ to D, namely the BS in Figure 3) and the downstream nodes (i.e., those nodes that receive a packet with OP_code = LREQ from D, i.e., nodes C and E). In Table 1, there are six fields. NodeID represents the node's ID, e.g., D, LinkNodeID includes IDs of those nodes directly connected to node D (e.g., C and E), and the node itself, i.e., D, LinkType shows the relationship between the node (e.g., C) and D (e.g., UP means C is an upstream node of D, DW indicates that the corresponding node (e.g., BS) is a downstream node of D), Time_SynTarget represents time synchronization objects of LinkNodeID nodes (e.g., node C and E are time synchronized with D, and D is time synchronized with the BS), Bandwth shows the current available link bandwidth between D and node LinkNodeID, and Default Bandwth depicts that the maximum link bandwidth between D and node LinkNodeID (usually the maximum bandwidth of the network card).

NodeID	LinkNodeID	LinkType	Time_SynTarget	Bandwth	Default Bandwth
D	C	UP	D	default bandwidth	default bandwidth
D	E	UP	D	default bandwidth	default bandwidth
D	BS	DW	BS	default bandwidth	default bandwidth
D	D	GN	C	default bandwidth	

Table 1: Relationship Table of node D shows the relationship between itself and all its upstream and downstream nodes

Assume a level- i node Z^i has a total of m immediate upstream nodes, denoted by $USN(Z^i) = U_{Z^1}^{i+1}, U_{Z^2}^{i+1}, \dots, U_{Z^m}^{i+1}$ at level- $(i+1)$. Let $set(Z^i) = Z^i \cup USN(Z^i) = Z^i, U_{Z^1}^{i+1}, U_{Z^2}^{i+1}, \dots, U_{Z^m}^{i+1}$. For example, in Figure 3, $Set(D) = D, C, E$.

3.2 Bandwidth/time allocation phase

In the second phase, a node N in the spanning tree is assigned bandwidth/time by its downstream node according to the number of nodes in the subtree rooted at N . In $Set(Z^i)$, if the bandwidth/time allocated to an upstream node by Z^i is not properly synchronized, the time point at which $U_{Z^j}^{i+1}$ finishes packet transfer, denoted by $T_{U_{Z^j}^{i+1}}^{last}$ may not be equal to the time point at which $U_{Z^{(j+1)}}^{i+1}$ starts sending packets to Z^i , denoted by $T_{U_{Z^j}^{i+1}}^{start}$, i.e., $T_{U_{Z^j}^{i+1}}^{last} \neq T_{U_{Z^{(j+1)}}^{i+1}}^{start}$, but they should be equal. If the time $T_{U_{Z^j}^{i+1}}^{last}$ is later than the time $T_{U_{Z^{(j+1)}}^{i+1}}^{start}$, i.e., $T_{U_{Z^j}^{i+1}}^{last} > T_{U_{Z^{(j+1)}}^{i+1}}^{start}$, the packets sent by $U_{Z^j}^{i+1}$ may collide with the packets transmitted by $U_{Z^{(j+1)}}^{i+1}$. Under a TCP environment, both collided packets will be dropped and re-transmitted, resulting in waste of bandwidth and delaying the arrival of these packets at BS. On the contrary, if the time $T_{U_{Z^j}^{i+1}}^{last}$ is earlier than the time $T_{U_{Z^{(j+1)}}^{i+1}}^{start}$, i.e., $T_{U_{Z^j}^{i+1}}^{last} < T_{U_{Z^{(j+1)}}^{i+1}}^{start}$, then $U_{Z^j}^{i+1}$ has already completed transmission, but $U_{Z^{(j+1)}}^{i+1}$ is unable to seamlessly start sending packets in time, also causing the waste of bandwidth. Likewise, the packets sent by $U_{Z^{(j+1)}}^{i+1}$ might possibly collide with those packets delivered by $U_{Z^{(j+2)}}^{i+1}$.

3.2.1 Bandwidth/time allocation in a set

As described above, when the WSN system starts, the initial bandwidth allocation is in accordance with the node number of a sub-tree rooted at a level-1 node, i.e., the more nodes a level-1 sub-tree has, the wider bandwidth the sub-tree will be allocated under the assumption that the probabilities of event detection for all nodes are equal, i.e., a uniform distribution. Assume the sub-tree rooted at level-1 node U_{BSj}^1 has $|U_{BSj}^1|$ nodes, the bandwidth/time duration allocated to U_{BSj}^1 , denoted by D_j^1 , is:

$$D_j^1 = \frac{|U_{BSj}^1|}{\sum_{h=1}^m |U_{BS^h}^1|} \times D_{BS}^0, \quad 1 \leq j \leq m \quad (1)$$

where m is the total number of level-1 nodes, $\sum_{h=1}^m |U_{BS^h}^1|$ represents the total number of nodes of the spanning tree (excluding the BS since it does not sense the environment), and D_{BS}^0 is the default cycle period set by the BS for bandwidth reallocation. In other words, in the next phase, i.e., the bandwidth/-time adjustment phase, every D_{BS}^0 time unit, the BS reallocates bandwidth to each level-1 node U_{BSj}^1 , $1 \leq j \leq m$. After that, a level-1 node will allocate a part of the allocated bandwidth to one of its level-2

nodes according to the number of nodes in the sub-tree rooted at this level-2 node. The general rule is that for a level- i node Z^i and $\text{Set}(Z^i) = Z^i, U_{Z^i}^{i+1}, U_{Z^i}^{i+1}, \dots, U_{Z^i}^{i+1}$, the time $D_{Z^i}^{i+1}$ allocated to $U_{Z^i}^{i+1}$ by Z^i is:

$$D^i + 1_{Z^i,j} = \frac{|U_{Z^i}^{i+1}|}{1 + \sum_{h=1}^{n_{Z^i}} |U_{Z^i}^{i+1}|} \times D_{Z^i}^i, \quad (2)$$

$$D'_{Z^i} = \frac{1}{1 + \sum_{h=1}^{n_{Z^i}} |U_{Z^i}^{i+1}|} \times D_{Z^i}^i, \quad (3)$$

$$D'_{Z^i} + \sum_{h=1}^{n_{Z^i}} D_{Z^i}^{i+1} \times D_{Z^i}^i, \quad (4)$$

where Z^i is the Z^i -th node in level- i , $1 \leq i \leq k, 1 \leq Z^i \leq q$, where k is the height of the spanning tree, q is the number of level- i nodes (including Z^i), n_{Z^i} is the number of immediate upstream nodes of Z^i , D'_{Z^i} is the time duration that Z^i allocates to itself, $U_{Z^i}^{i+1}$ is the j -th upstream node of node Z^i (of course, at level- $(i+1)$), $|U_{Z^i}^{i+1}|$ is the number of nodes in the sub-tree rooted at $U_{Z^i}^{i+1}$ and $D_{Z^i}^i$ is the bandwidth/time allocated to node Z^i by Z^i 's downstream node S . In Eqs.(2) and (3), 1 is added to take account of Z^i node. The above procedure repeats until all level- k nodes have been allocated with bandwidth/time. For example, in Figure 3, the numbers of nodes in the sub-trees rooted at D, F, and I are 5, 3, and 4, respectively. Thus, the bandwidth allocated to nodes D, E, and F are $5/12$, $3/12$, and $4/12$, respectively, of bandwidth of the BS where 12 is the number of nodes in the WSN, excluding BS.

OP_code=BTA	SenderID	ReceiverID	$T_{U_{Z^i}^{i+1}}^{start}$	$D_{U_{Z^i}^{i+1}}$
-------------	----------	------------	-----------------------------	---------------------

Figure 4: Format of a bandwidth/time duration allocation packet (sent by Z^i to $U_{Z^i}^{i+1}$).

OP_code=BTA	SenderID	ReceiverID
-------------	----------	------------

Figure 5: Format of the bandwidth/time duration allocation acknowledge packet (sent by $U_{Z^i}^{i+1}$ to Z^i).

After level- i node Z^i finishes calculating the bandwidth/time that will be allocated to level- $(i+1)$ nodes, it will send a bandwidth/time allocation packet (in the format shown in Figure 4) to its immediate upstream node $U_{Z^i}^{i+1}$, $1 \leq j \leq m$, where m is the number of upstream node of Z^i . This packet consists of five fields, including OP_code, SenderID, ReceiverID, $T_{U_{Z^i}^{i+1}}^{start}$, and $D_{U_{Z^i}^{i+1}}$, where OP_code=BTA represents that it is a bandwidth/time allocation packet, SenderID is the level- i node Z^i , ReceiverID is the upstream node in level- $(i+1)$ (i.e., $U_{Z^i}^{i+1}$), and $T_{U_{Z^i}^{i+1}}^{start}$ and $D_{U_{Z^i}^{i+1}}$ are the transmission starting time and transmission time duration allocated to $U_{Z^i}^{i+1}$, respectively. When receiving the bandwidth/time allocation packet, $U_{Z^i}^{i+1}$ replies Z^i with a bandwidth/time allocation reply packet, which as shown in Figure 5 consists of three fields, including OP_code, SenderID, and ReceiverID, where OP_code=BACK represents that it is an allocation acknowledgement packet, SenderID is the sender node in level $(i+1)$, i.e., $U_{Z^i}^{i+1}$, and ReceiverID is the immediate downstream node of the sender, i.e., Z^i .

3.2.2 bandwidth/time synchronization

OP_code=TSYN	NodeID	DR
--------------	--------	----

Figure 6: Format of a time synchronization packet (sent by $U_{Z^i_j}^{i+1}$ to Z^i).

In $\text{Set}(Z^i)$, when detecting an event, a node $U_{Z^i_j}^{i+1}$ wants to send packets to Z^i , it first sends a time synchronization packet to Z^i aiming to synchronize the time between them. As shown in Figure 6, the time synchronization packet consists of three fields, including OP_code, node ID, and DR, where OP_code=TSYN indicates that it is a time synchronization packet, NodeID shows that an upstream node of Z^i , i.e., $U_{Z^i_j}^{i+1}$, wishes to send packets to Z^i , and DR is the data rate or data flow sent by $U_{Z^i_j}^{i+1}$ to Z^i per unit of time. At last, the time point at which a time synchronization packet is sent by $U_{Z^i_j}^{i+1}$ ($U_{Z^i_j}^{i+1}$) is denoted by $T_{U_{Z^i_j}^{i+1}}^{cur}$ ($T_{U_{Z^i_j}^{i+1}}^{cur}$).

OP_code=SACK	ReceiverID	T_{Z^i}	ΔT
--------------	------------	-----------	------------

Figure 7: Format of a time synchronization acknowledgement packet (sent by Z^i to $U_{Z^i_j}^{i+1}$).

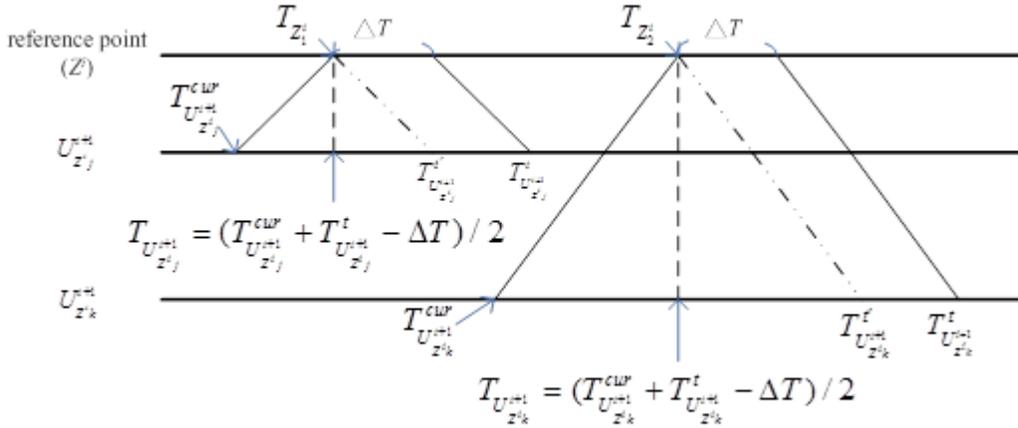


Figure 8: Time synchronization diagram. $T_{U_{Z^i_j}^{i+1}}^{i+1} = T_{U_{Z^i_j}^{i+1}}^t - \Delta T$, $T_{U_{Z^i_k}^{i+1}}^{i+1} = T_{U_{Z^i_k}^{i+1}}^t - \Delta T$, if $T_{Z^i_1} = 1:30:18:001$ (hour: minute: second: mini-second), $T_{U_{Z^i_j}^{i+1}}^{i+1} = 1:30:17:002$, then $\tau_{U_{Z^i_j}^{i+1}}^{i+1} > 0$; if $T_{Z^i_2} = 1:30:19:010$, $T_{U_{Z^i_k}^{i+1}}^{i+1} = 1:30:20:002$, then $\tau_{U_{Z^i_j}^{i+1}}^{i+1} > 0$

When Z^i receives the time synchronization packet, it replies $U_{Z^i_j}^{i+1}$ with a time synchronization acknowledgement packet, which as shown in Figure 7 consists of four fields: OP_code, ReceiverID (i.e., $U_{Z^i_j}^{i+1}$), $U_{Z^i_j}^{i+1}$, T_{Z^i} and ΔT , where T_{Z^i} as illustrated in Figure 8 represents the time when Z^i receives the time synchronization packet, ΔT is the elapsed time duration from the time point when Z^i receives the time synchronization packet to the time when it starts sending the time synchronization acknowledgement packet to $U_{Z^i_j}^{i+1}$, i.e., the time spent in processing the time synchronization packet and preparing the

time synchronization acknowledgement packet. When $U_{Z^i j}^{i+1}$ receives the time synchronization acknowledgement packet, it records the current time as $T_{U_{Z^i j}^{i+1}}^t$ and computes the time difference between $U_{Z^i j}^{i+1}$ and Z^i , denoted by $\tau_{U_{Z^i j}^{i+1}}$, by using Eq. (5).

$$\tau_{U_{Z^i j}^{i+1}} = T_{Z^i} - (T_{U_{Z^i j}^{i+1}}^{cur} + T_{U_{Z^i j}^{i+1}}^t - \Delta T)/2 \quad (5)$$

under the assumption that the bandwidths from Z^i to $U_{Z^i j}^{i+1}$ and vice versa are the same where the first item on the right hand side of Eq. (5), T_{Z^i} , is an instance of T_{Z^i} at Z^i . Let

$$T_{U_{Z^i j}^{i+1}} = (T_{U_{Z^i j}^{i+1}}^{cur} + T_{U_{Z^i j}^{i+1}}^t - \Delta T)/2 \quad (6)$$

which is the time point at $U_{Z^i j}^{i+1}$ when the time point of Z^i is T_{Z^i} . In general, the time difference between $U_{Z^i k}^{i+1}$ and Z^i , i.e., $\tau_{U_{Z^i k}^{i+1}}$, can be computed by using Eq. (7):

$$\tau_{U_{Z^i k}^{i+1}} = T_{Z^i} - (T_{U_{Z^i k}^{i+1}}^{cur} + T_{U_{Z^i k}^{i+1}}^t - \Delta T)/2 \quad (7)$$

also under the similar assumption where T_{Z^i} is another instance of T_{Z^i} at Z^i . Let

$$T_{U_{Z^i k}^{i+1}} = (T_{U_{Z^i k}^{i+1}}^{cur} + T_{U_{Z^i k}^{i+1}}^t - \Delta T)/2 \quad (8)$$

It means at some moment of time if the time of $U_{Z^i j}^{i+1}$ (i.e., $T_{U_{Z^i j}^{i+1}}$) is slower than that of Z^i (i.e., T_{Z^i}), i.e., $T_{Z^i} > T_{U_{Z^i j}^{i+1}}$, e.g., as shown in Figure 8, in which $T_{Z^i} = 1:30:18:001$ (hour: minute: second: mini-second) and at the same moment $T_{U_{Z^i j}^{i+1}} = 1:30:17:002$, then $\tau_{U_{Z^i j}^{i+1}} > 0$. On the other hand, if the time of $U_{Z^i j}^{i+1}$ is faster than that of Z^i , e.g. $T_{Z^i} = 1:30:19:010$ and $T_{U_{Z^i j}^{i+1}} = 1:30:20:002$, $\tau_{U_{Z^i j}^{i+1}} < 0$. If $\tau_{U_{Z^i j}^{i+1}} = 0$, the time of Z^i and that of $U_{Z^i j}^{i+1}$ are synchronized. An upstream node, e.g., $U_{Z^i j}^{i+1}$, needs to calculate and record the time difference, i.e., $\tau_{U_{Z^i j}^{i+1}}$. Then $T_{U_{Z^i j}^{i+1}}^{start} + \tau_{U_{Z^i j}^{i+1}}$ is the synchronized starting time of $U_{Z^i j}^{i+1}$ where $T_{U_{Z^i j}^{i+1}}^{start}$ is the transmission starting time allocated to it by Z^i . At last, $U_{Z^i j}^{i+1}$ uses the allocated time period $D_{U_{Z^i j}^{i+1}}$ to send detected data packets to Z^i , which in turn resends them to the BS.

3.3 Bandwidth/time adjustment phase

When a node $U_{Z^i j}^{i+1}$ detects that there is an event or the end of an event, or the data rates of some of its upstream nodes are changed, it sends a change-of-datarate packet to its downstream node Z^i . As shown in Figure 9, this packet consists of four fields, including OP_code, SenderID, ReceiverID, and $\pm\Delta DR$, where OP_code=FCH indicates change of data rate of the SenderID node (i.e., $U_{Z^i j}^{i+1}$), ReceiverID represents the ID of the target node (i.e. Z^i) of this packet, and $\pm\Delta DR$ shows the increment(+) or decrement(-) of the data rate DR. When Z^i receives a change-of- datarate packet from an upstream node $U_{Z^i j}^{i+1}$, if originally $Z_{Z^i j} = 0$, meaning the value of the $\pm\Delta DR$ field is a positive. If $\pm\Delta DR$ is negative, and after subtraction, $Z_{Z^i j} \leq 0$, it means that the event detected by $U_{Z^i j}^{i+1}$ or its upstream nodes has been over or handled.

When an upstream node detects an event, it estimates how many packets it is going to send per sec-

OP_code=FCH	SenderID	ReceiverID	±ΔDR
-------------	----------	------------	------

Figure 9: Format of the change-of-datarate packet (sent by $U_{Z^i}^{i+1}$ to Z^i)

ond and accordingly calculates its data rate. The data rate can also be preset by the administrator of the WSN when sensor nodes were deployed. This option can be chosen depending on the real requirements. In the bandwidth/time adjustment phase after time synchronization, as mentioned above, for every D_{BS}^0 , Z^i re-allocates bandwidth/time to the upstream node $U_{Z^i}^{i+1}$ and Z^i itself for the $\text{Set}(Z^i) = Z^i, U_{Z^i}^{i+1}, U_{Z^i}^{i+2}, \dots, U_{Z^i}^{i+m}$, based on the data rates currently allocated to Z^i and $U_{Z^i}^{i+1}$, denoted by DR_{Z^i} and $DR_{U_{Z^i}^{i+1}}$, respectively, $1 \leq j \leq m$. Generally, the adjustment is performed outward from BS until level-k nodes.

$$B_{Z^i} = \frac{\min(B_{U_{Z^i}^{i+1}, default'} \frac{DR_{U_{Z^i}^{i+1}}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D)}{\frac{DR_{Z^i}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D + \sum_{j=1}^m \min(B_{U_{Z^i}^{i+1}, default'} \frac{DR_{U_{Z^i}^{i+1}}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D)} \times B_D \quad (9)$$

$$B'_{Z^i} = \frac{\frac{DR_{Z^i}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D}{\frac{DR_{Z^i}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D + \sum_{j=1}^m \min(B_{U_{Z^i}^{i+1}, default'} \frac{DR_{U_{Z^i}^{i+1}}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D)} \times B_D \quad (10)$$

$$B_{Z^i} = \frac{\min(B_{U_{Z^i}^{i+1}, default'} \frac{DR_{U_{Z^i}^{i+1}}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D)}{\frac{DR_{Z^i}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D + \sum_{j=1}^m \min(B_{U_{Z^i}^{i+1}, default'} \frac{DR_{U_{Z^i}^{i+1}}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D)} \times D_{Z^i} \quad (11)$$

$$B'_{Z^i} = \frac{\frac{DR_{Z^i}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D}{\frac{DR_{Z^i}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D + \sum_{j=1}^m \min(B_{U_{Z^i}^{i+1}, default'} \frac{DR_{U_{Z^i}^{i+1}}}{DR_{Z^i} + \sum_{k=1}^m DR_{U_{Z^i}^{i+1}}} \times B_D)} \times D_{Z^i} \quad (12)$$

where $B_D(D_{Z^i})$ is the bandwidth (time) duration currently allocated to Z^i by its downstream node D, $B_{Z^i}(B'_{Z^i})$ is the bandwidth Z^i allocated to $U_{Z^i}^{i+1}$ (to itself), $D_{Z^i}(D'_{Z^i})$ is the time duration Z^i allocates to $U_{Z^i}^{i+1}$ (to itself) according to the data rate that $U_{Z^i}^{i+1}(Z^i)$ currently declares to generate, i.e., $DR_{U_{Z^i}^{i+1}}(DR_{Z^i})$, where $DR_{U_{Z^i}^{i+1}}, DR_{Z^i} \leq B_D$, $j=1,2,\dots,m$, and $B_{U_{Z^i}^{i+1}, default'}$ is the maximum transmission bandwidth (i.e., the bandwidth of the NIC, also called the default bandwidth, i.e., BWdefault in Figure 1) between $U_{Z^i}^{i+1}$ and Z^i . Furthermore,

$$D_{Z^i} = D'_{Z^i} + \sum_{j=1}^m D_{Z^i} \quad (13)$$

When Z^i receives a change-of-datarate packet, in principle it needs to reallocate bandwidth for all

nodes in $\text{Set}(Z^i)$. But to reduce loads of the network and nodes, if the variation of the total data rate does not exceed a default threshold, e.g., 10%, no adjustment will be performed even if D_{BS}^0 expires. Z^i will postpone adjustment of bandwidth/time for its immediate upstream nodes until receiving the next change-of-data-rate packet such that the accumulative variation of data rate exceeds the threshold. After finishing bandwidth/time duration allocation, Z^i notifies its upstream nodes by sending the bandwidth/time duration allocation packet shown in Figure 4. Upon receiving the packet, the upstream node will reply Z^i with the bandwidth/time duration allocation reply packet shown in Figure 5 and send packets to Z^i according to the allocated bandwidth/time duration. Take the network illustrated in Figure 3 as an example. According to Eq.(1), the BS divides an allocation time duration D_{BS}^0 , e.g., 5 minutes, into D_D , D_F , and D_I as shown in Figure 10 and allocates them to level-1 nodes D, F, and I, respectively. The BS then notifies node D of the start transmission time T_D^{start} , and its allocated time duration D_D . Thus, D can transfer packets to BS from T_D^{start} to $T_D^{last} = (T_D^{start} + D_D)$. T_D^{last} is also the time point for F to start transfer, i.e., $T_F^{start} = T_D^{last} = T_D^{start} + D_D$. Likewise, F is notified with the allocated time duration D_F such that the time $T_F^{last} = T_F^{start} + D_F$. Time allocated to node I is calculated by the similar method. Note that BS does not allocate time duration to itself.

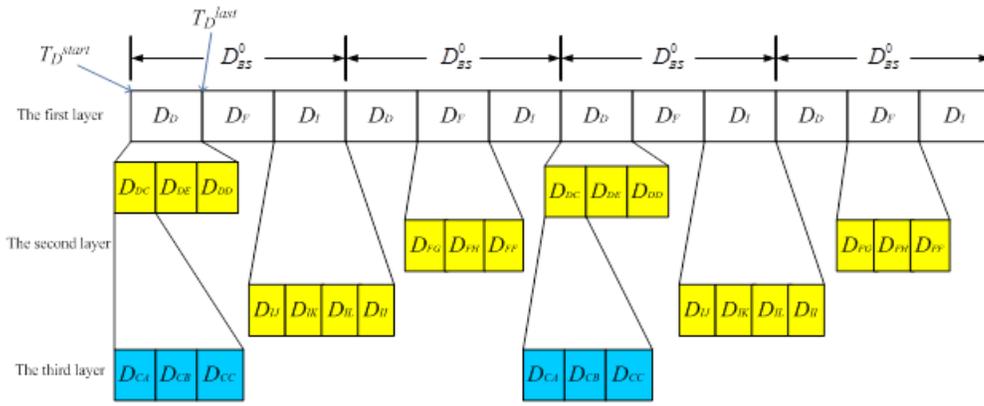


Figure 10: Time allocation diagram for nodes according to network architecture shown in Figure 3.

The method of allocating packet transmission time durations to level-2 nodes is as follows. Taking node D in Figure 3 as an example, D allocates the time duration allocated to it, i.e., D_D , to upstream nodes C, E and D itself as D_{DC} , D_{DE} , and D_{DD} , respectively. Let the time for C (E) to start transferring packets to D be T_C^{start} (T_E^{start}). Since the allocated time duration for C (E) is D_{DC} (D_{DE}), the time to end transfer for C (E) will be $T_C^{last} = T_C^{start} + D_{DC}$ ($T_E^{last} = T_E^{start} + D_{DE}$), where $T_E^{start} = T_C^{last}$. Similarly, $T_D^{start} = T_E^{last}$ and $T_D^{last} = T_D^{start} + D_{DD}$. Transfer time durations for the remaining nodes in level-1 are calculated and allocated by the similar method. In the second cycle of time duration of D_{BS}^0 , this proposed system as shown in Figure 10 substitutes $T_D^{start} \leftarrow T_D^{start} + D_{BS}^0 = T_D^{last} + D_F + D_I$ and repeats all the above time allocation procedures again. If the spanning tree rooted at BS is called Q, and the height of Q is k, the time to start transfer and time durations allocated to nodes in level-n, $n=2,3,4,\dots,k$, are computed by using the similar procedures. Also assume that in any node set, e.g., $\text{Set}(Z^i)$, a time duration $D_{Z^i}^{set}$ required for Z^i to allocate time duration to node $U_{Z^i}^{i+1}$ of the set needs to satisfy Eq.(14).

$$D_{BS}^0 > \sum_{i=0}^{(n-n_{leaf})-1} D_{Z^i}^{set} \quad (14)$$

where Z^0 is BS, n is the number of nodes in Q (including the BS), and n_{leaf} is the number of leaf nodes in Q. Since each internal node of Q needs to allocate the time duration allocated to it by its downstream node to its upstream nodes, if the inequality in Eq. (14) is not satisfied, when D_{BS}^0 expires

forcing BS to start the next bandwidth reallocation, some upstream nodes may not have synchronized. In this case, if any unsynchronized node detects an event or needs to continue sending packets for some former detected events, packet collision may occur.

OP_code=DATA	SourceID	Data
--------------	----------	------

Figure 11: Format of a data packet.

Figure 11 shows a data packet which is employed by nodes to send their detected environmental data. The packet consists of three fields, including OP_code, SenderID, and Data, where OP_code=DATA indicates that it is a data packet, SourceID represents ID of the source node that detects the event and Data are the transmitted environmental data.

4 Simulation Results and Discussion

We use ns-2 as the simulation tool to evaluate TMCoS and compare it with some state-of-the-art systems, including MUCOM [13], PCCP [21] and HCCP [18]. The specifications of our test bed are listed in Table 2, in which the values may be changed when necessary. The sink node is located at (50, 50) of the 100m x 100m sensing field, i.e., the center of the field. 49 sensor nodes were randomly deployed to sense their surrounding environments and relay packets. The sink node only collects data packets sent by other sensor nodes. Four experiments were performed in this study. The first evaluated three QoS parameters including throughputs, end-to-end delays and packet drop rates given different packet sizes. The second, the third and the fourth also study the three QoS parameters but given different packet rates, and different numbers of events, and different event-lasting times, respectively.

Parameter	Value
Sink node	1
Number of sensor nodes	49
Max bandwidth of a link	250Kbps = 31.25KB
Packet rate of a source node	10 pkts/sec
Packet size	1 KB/pkt
Number of events occurs	5
Event lasting time	25 sec

Table 2: Parameters of the experimental environment

4.1 The first experiment-Different Packet Sizes

In the first experiment, the given packet sizes range from 1KB to 5KB, rather than 1KB listed in Table 2. Hence, the data rates are between 10KB/sec (=10pkts/sec * 1KB/pkt) and 50KB/sec (=10pkts/sec * 5KB/pkt). Figure 12 illustrates the throughputs for these systems. Before bandwidth of a link is saturated, throughputs are almost proportional to packet sizes since a longer packet carries much more data, no matter which scheme was tested. But when bandwidth is gradually saturated, i.e., when packet size is larger than 150 KB/sec (≈ 31.25 KB/sec), throughputs approach toward flat. Because TMCoS does not use the flooding prevention mechanism [13], its throughput on packet size = 5KB is not better than that of the MUCOM. The HCCP lowers its data rate to reduce the amount of packets transmitted to the

sink. Due to less number of packets to be delivered in a time unit, even though its channel competition time is shorter, but throughputs are also lower than those of other tested schemes. Figure 13 shows the end-to-end delay. It is clear that longer packets consume longer transmission time. Owing to regulating data rates of upstream nodes when the corresponding downstream link is congested, the end-to-end delay of TMCoS is lower than those of other tested schemes. Basically, MUCOM utilizes frequency division approach which distributes bandwidth of a link to its upstream nodes. This will narrow the out-bandwidth of an upstream node, consequently delaying packet transmission, especially when events intermittently occur and data packets generated are huge and sudden. This is the reason why its delay time is longer than that of TMCoS. From the TMCoS viewpoint, it uses TDMA to transmit data. During the allocated time slot, the link bandwidth can be totally used by the node. So data can be quickly transferred, resulting in shorter delay time. Figure 14 shows the packet drop rates. Basically throughputs are higher when packet loss rates are smaller. HCCP reduces its data rates to lower the probability of packet loss. So its packet loss rate is the lowest among the tested schemes.

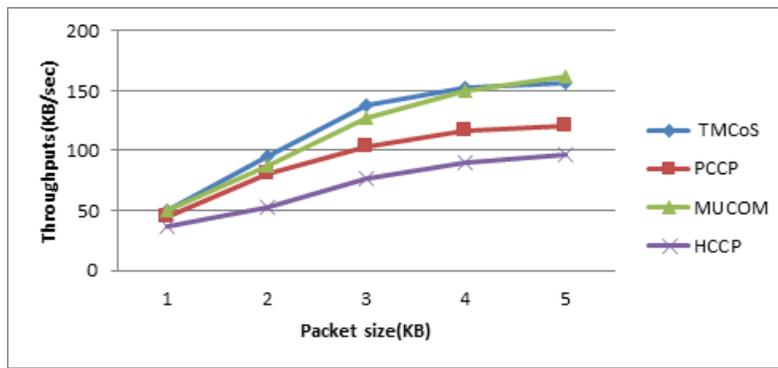


Figure 12: Throughputs at the sink on different packet sizes.

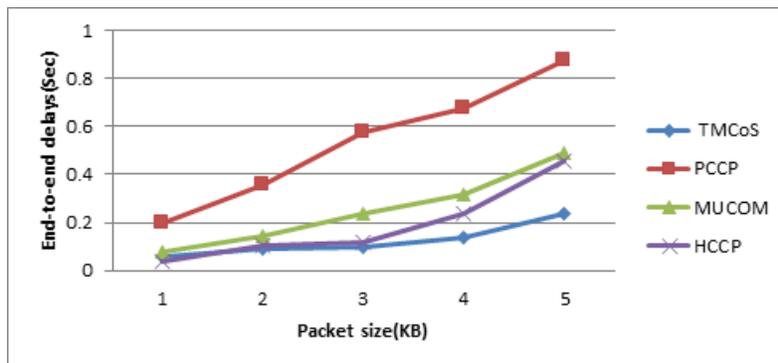


Figure 13: End-to-end delays on different packet sizes.

4.2 The second experiment-Different Packet Rates

In the second experiment, the given packet rates range from 10pkts/sec to 50pkts/sec, rather than 10pkts/sec listed in Table 2. Hence, the data rates are between 10KB/sec (=10pkts/sec * 1KB/pkt) and 50KB/sec (=50pkts/sec * 1KB/pkt). Figure 15 illustrates the throughputs for these tested systems. When

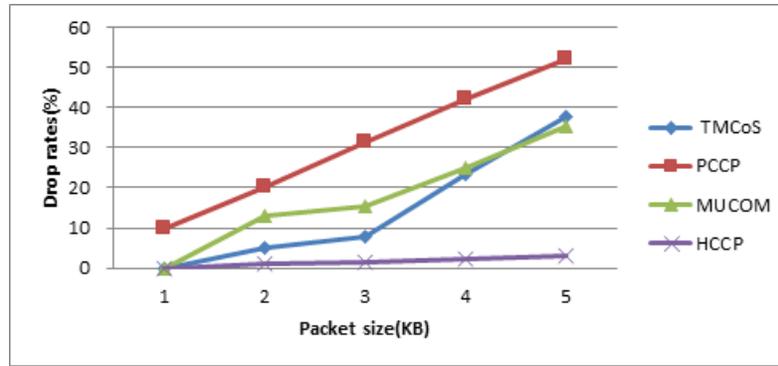


Figure 14: Packet drop rates on different packet sizes.

packet rate is lower than 30KB/sec, throughputs are almost proportional to packet rates since only a small number of packets are lost. But when bandwidth is gradually saturated, like that in the first experiment, throughputs approach flat. Because TMCoS does not use flooding prevention mechanism, its throughput on packet rate = 50(pkt/sec) is not better than that of the MUCOM. The reason is that when a larger number of packets is transmitted, the probability of channel contention collision is then higher, resulting in lower throughputs.

Figure 16 shows the end-to-end delay. When packet rates are higher, due to buffer overflow, the probability with which a packet is dropped is also increased, causing longer packet transmission time, especially when dropped packets need to be retransmitted. But owing to regulating upstream-node data rates, the end-to-end delay of TMCoS is lower than those of other tested schemes.

Figure 17 shows the packet drop rates. Basically, when packet loss rates are smaller, throughputs will be higher. HCCP as mentioned above, due to reducing packet rates, its packet loss rates are then lower than those of other tested schemes.

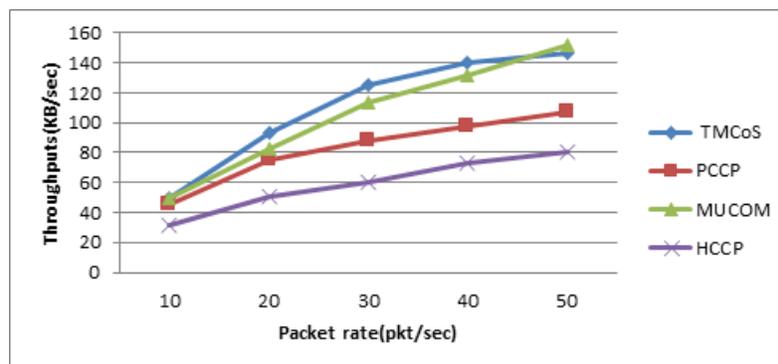


Figure 15: Throughputs at the sink on different packet rates.

4.3 The third experiment-Different Numbers of Events

In the third experiment, different numbers of events, including 5, 10, 15, 20 and 25, were given. Figure 18 illustrates the throughputs of these systems. When numbers of events increase, the throughputs are also higher since many packets are generated and delivered per second. All tested schemes have the

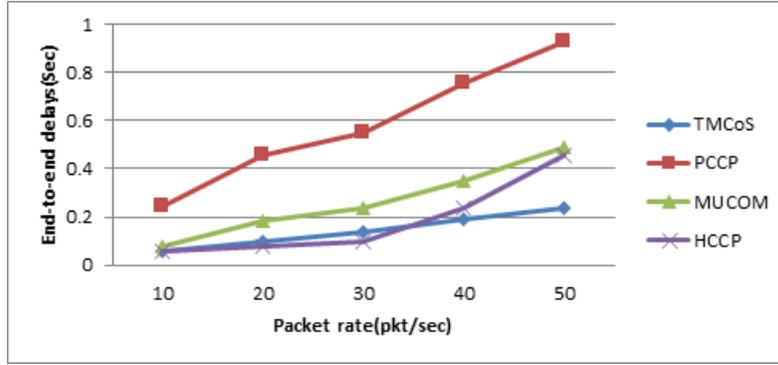


Figure 16: End-to-end delays on different packet rates.

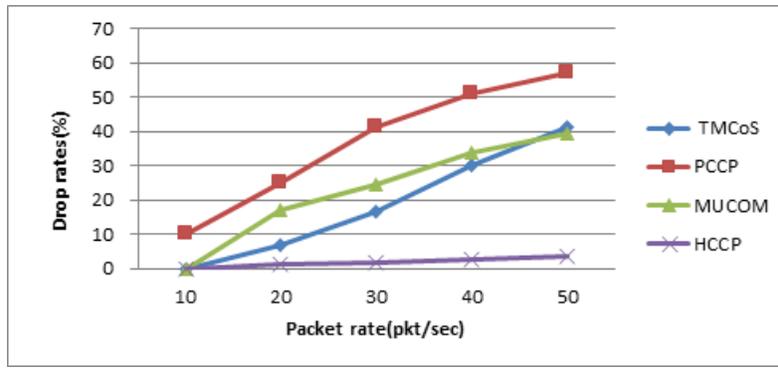


Figure 17: Packet drop rates on different packet rates.

same phenomenon. As mentioned above, the throughput of the TMCoS on number of events = 25 is not better than the MUCOM. But it outperforms the other schemes and the MUCOM when number of events is less than 25. Figures 19 and 20, respectively, show the end-to-end delay and packet drop rates. Their experimental results are similar to those of the two previous experiments. We do not redundantly describe them here.

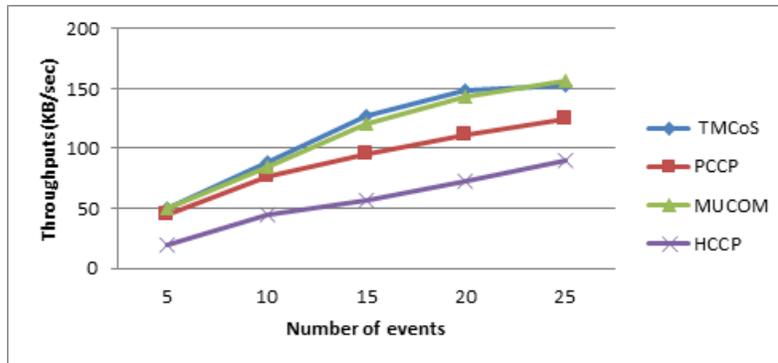


Figure 18: Throughputs at the sink on different numbers of occurred events.

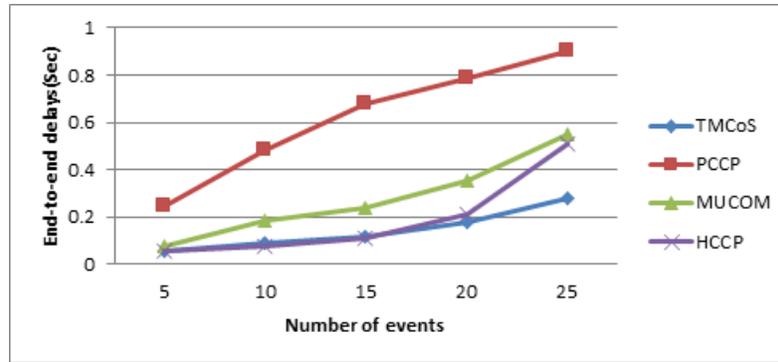


Figure 19: End-to-end delays on different numbers of occurred events.

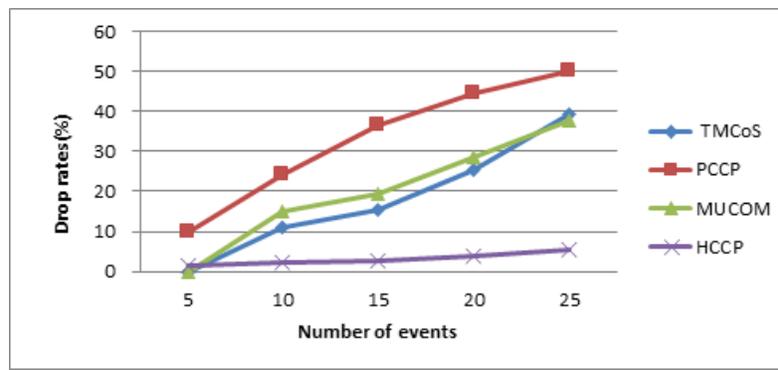


Figure 20: Packet drop rates on different numbers of occurred events.

4.4 The fourth experiment- Different Event-lasting Times

In the fourth experiment, the event-lasting times range between 25 and 200 sec instead of 25 sec listed in Table 2. When an event-lasting time is longer, since with the same packet rate, many more packets are generated and sent per second, the probability of channel contention collision will be higher, so the throughputs are also lower.

Figure 22 shows the end-to-end delay, as lasting time is increased, the probability of channel contention congestion is higher, thus increasing the delay time.

Figure 23 shows the packet drop rates. HCCP as mentioned above, due to reducing packet rates, its packet loss rates are then lower.

Figure 24 shows the difference of energy consumed by the TMCoS which is a TDMA approach, and the MUCOM which is a FDMA scheme. It is clear that the MUCOM consumes more energy than the TMCoS does. The reason is that with TDMA, nodes can enter their sleep mode during the time periods out of their assigned time slots. But nodes of FDMA need to be always in their active mode. They can enter sleep mode sometimes. But the time period is not fixed depending on how much data needed to be delivered. Of course, the price of TDMA is that an upstream node needs to periodically synchronize with its downstream node.

From the above experiments, we can learn that when bandwidth of a link is fixed, and transmission capacity is high, the link will gradually be saturated, hence increasing the end-to-end delays and packet loss rates. When many more events occur, the probability of channel contention and congestion will

be more serious. Throughputs are then reduced. However, reducing the packet size can result in lower probability of the channel contention and congestion. Although HCCP has a relatively low drop rate, its throughputs are relatively low. The MUCOM disperses its data flow to neighbor nodes and adopts congestion prevention mechanism. Its throughputs are relatively higher than those of PCCP and HCCP, and its end-to-end delays are a little longer than those of the two schemes. From these experiments, we dare to say that the TMCoS outperforms the other three tested schemes.

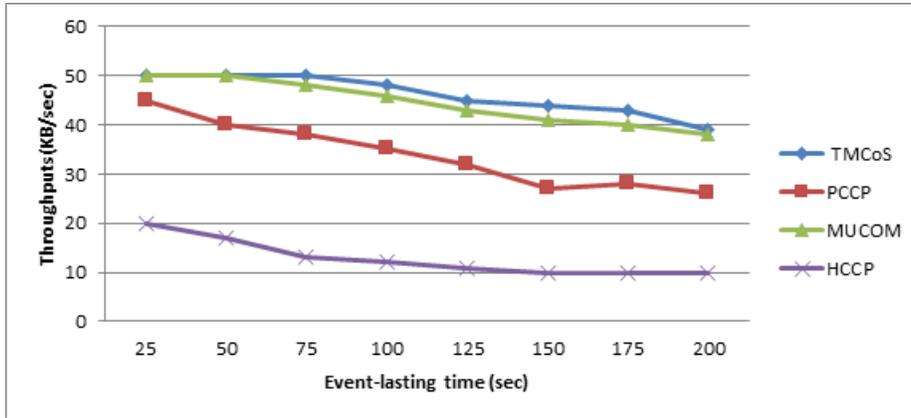


Figure 21: Throughputs at the sink.

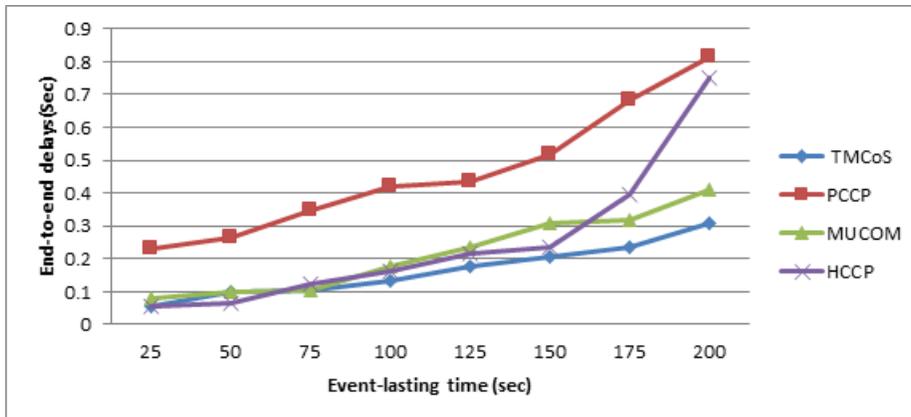


Figure 22: Average end-to-end delays.

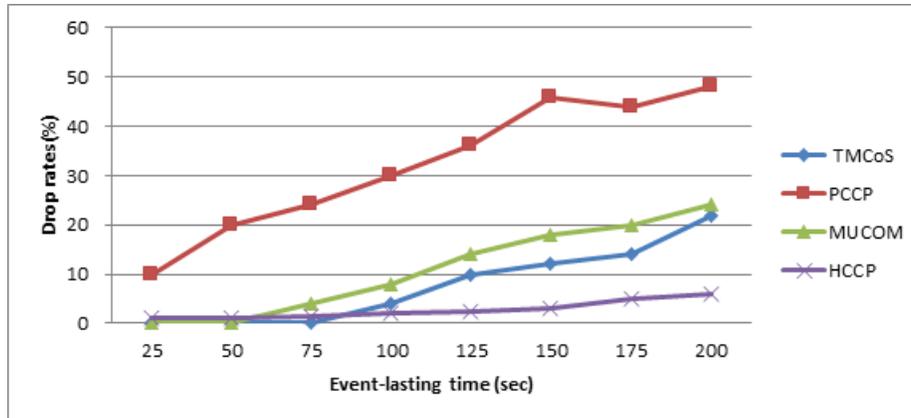


Figure 23: Average packet drop rates.

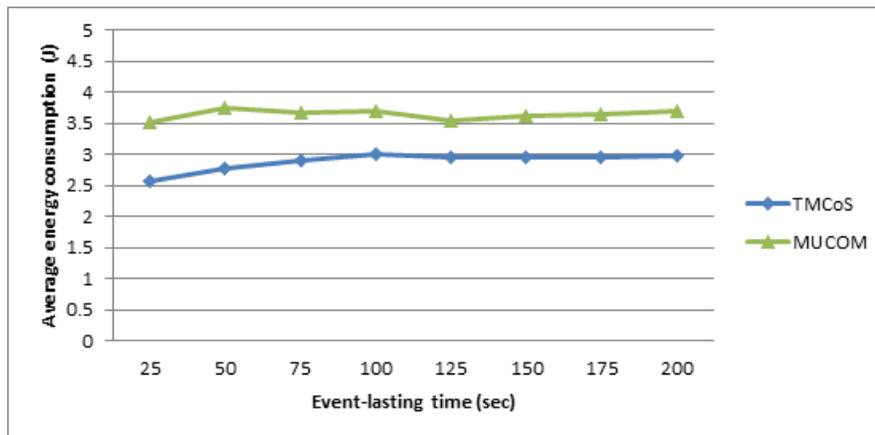


Figure 24: Average energy consumption.

5 Conclusions and Future Work

In this paper, we propose a time division method to solve the congestion-control problem for an event-driven sensor network. When a node is congested, our method adjusts the duration of the time slot allocated to this node and regulates its generated traffic to mitigate the congestion. Experimental results demonstrate that this method can effectively improve a WSN's throughputs, shorten its end-to-end delays and reduce its packet loss rates. In the future, we would like to develop another spanning-tree construction algorithm to create a spanning tree which has higher performance than that of the one we current use. It can also show which node is the bottleneck of data transmission, and regulate the traffic transmitted by all its upstream nodes. We will further derive the TMCoS's behavior model and reliability model so that user can predict their behaviors and reliabilities before using it. Experimental results above also show that the congestion prevention mechanism employed by the MUCOM can effectively improve throughputs. So we will increase the TMCoS with this function to improve its throughputs. These constitute our future studies.

References

- [1] R. Bai, Y. Qu, Y. Guo, and B. Zhao. An energy-efficient TDMA MAC for wireless sensor networks. In *Proc. of the 2nd IEEE Asia-Pacific Services Computing Conference (ASCC'07), Tsukuba Science City, Japan*, pages 69–74. IEEE, December 2007.
- [2] P. Corke, T. Wark, R. Jurdak, H. Wen, P. Valencia, and D. Moore. Environmental wireless sensor networks. *Proceedings of the IEEE*, 98(11):1903–1917, November 2010.
- [3] O. Demigha, W. Hidouci, and T. Ahmed. On energy efficiency in collaborative target tracking in wireless sensor network: A review. *IEEE Communications Surveys & Tutorials*, 15(3):1210–1222, 2013.
- [4] W. Fang, F. Liu, F. Yang, L. Shu, and S. Nishio. Energy-efficient cooperative communication for data transmission in wireless sensor networks. *IEEE Transactions on Consumer Electronics*, 56(4):2185–2192, November 2010.
- [5] N. Farzaneh and M. Yaghmaee. Probability based hop selection approach for resource control in wireless sensor network. In *Proc. of the 6th International Symposium on Telecommunications (IST'12), Tehran, Iran*, pages 703–708. IEEE, November 2012.
- [6] F. Ganhaio, M. Pereira, L. Bernardo, R. Dinis, R. Oliveira, and P. Pinto. Energy per useful packet optimization on a tdma wsn channel. In *Proc. of the 19th International Conference on Computer Communications and Networks (ICCCN'10), Zurich, Switzerland*, pages 1–6. IEEE, August 2010.
- [7] S. Gobriel, D. Mosse, and R. Cleric. TDMA-ASAP: Sensor network TDMA scheduling with adaptive slot-stealing and parallelism. In *Proc. of the 29th IEEE International Conference on Distributed Computing Systems (ICDCS'09), Montreal, Quebec, Canada*, pages 458–465. IEEE, June 2009.
- [8] S. Gobriel, D. Mosse, and R. Cleric. TDMA-ASAP: Sensor network TDMA scheduling with adaptive slot-stealing and parallelism. In *Proc. of the 29th IEEE International Conference on Distributed Computing Systems (ICDCS'09), Montreal, Quebec, Canada*, pages 458–465. IEEE, June 2009.
- [9] J. Kang, Y. Zhang, and B. Nath. Accurate and energy-efficient congestion level measurement in ad hoc networks. In *Proc. of the 2005 IEEE Wireless Communications and Networking Conference (WCNC'05), New Orleans, Louisiana, USA*, volume 4, pages 2258–2263. IEEE, March 2005.
- [10] B. Kasireddy, Y. Wang, and J. Liu. Energy conservation using network coding in grid wireless sensor networks. In *Proc. of the 3rd International Workshop on Intelligent Systems and Applications (ISA'11), Wuhan, China*, pages 1–4. IEEE, May 2011.
- [11] T. Lai, C. Chen, J. Liu, J. Wang, C. Chuang, and J. Jiang. A novel dynamic convergecast tree generator for wsn-based environmental surveillance of orchid plantation. In *Proc. of the IEEE 14th International Conference on High Performance Computing and Communications & 9th International Conference on Embedded Software and Systems (HPCC-ICCESS'12), Liverpool, New York, USA*, pages 1629–1633. IEEE, June 2012.
- [12] M. Lazarescu. Design of a wsn platform for long-term environmental monitoring for iot applications. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 3(1):45–54, March 2013.
- [13] F. Leu, F. Jiang, C. Lien, S. Lai, and S. Chiou. A rate-allocation based multi-path control scheme for event-driven wireless sensor networks on constant event packet rates. *Computer Systems Science and Engineering*, 27(5):295–315, September 2012.
- [14] M. Maggs, S. O'Keefe, and D. Thiel. Consensus clock synchronization for wireless sensor networks. *Journal of IEEE Sensors*, 12(6):2269–2279, June 2012.
- [15] H. Myung, J. Lim, and D. Goodman. Single carrier fdma for uplink wireless transmission. *IEEE Vehicular Technology Magazine*, 1(3):30–38, September 2006.
- [16] D. Patil and S. Dhage. Priority-based congestion control protocol (pccp) for controlling upstream congestion in wireless sensor network. In *Proc. of the 2012 International Conference on Communication, Information & Computing Technology (ICCICT'12), Mumbai, India*, volume 1, pages 1–6. IEEE, October 2012.
- [17] D. Sathian, R. Baskaran, and P. Dhavachelvan. Lifetime enhancement by cluster head cooperative trustworthy energy efficient MIMO routing algorithm based on game theory for wsn. In *Proc. of the IEEE 3rd International Conference on Computing Communication & Networking Technologies (ICCCNT'12), Coimbatore, India*, pages 1–5. IEEE, July 2012.
- [18] J. Sheu, L. Chang, and W. Hu. Hybrid congestion control protocol in wireless sensor networks. *Journal of*

Information Science and Engineering, 25:1103–1119, 2009.

- [19] J. Yackovich, D. Mosse, A. Rowe, and R. Rajkumar. Making wsn TDMA practical: Stealing slots up and down the tree. In *Proc. of the 17th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'11)*, Toyama, Japan, pages 41–50. IEEE, August 2011.
 - [20] F. Yin, Z. Li, and H. Wang. Energy-efficient data collection in multiple mobile gateways WSN-MCN convergence system. In *Proc. of the IEEE 2013 Consumer Communications and Networking Conference (CCNC'13)*, Las Vegas, Nevada, USA, pages 271–276. IEEE, January 2013.
 - [21] M. Zawodniok and S. Jagannathan. Predictive congestion control protocol for wireless sensor networks. *IEEE Transactions on Wireless Communications*, 6(11):3955–3963, November 2007.
-

Author Biography



Fang-Yie Leu received his bachelor, master and Ph.D. degrees all from National Taiwan University of Science and Technology, Taiwan, in 1983, 1986 and 1991, respectively. His research interests include wireless communication, network security, Grid applications and Security. He is currently a professor of Computer Science Department, TungHai University, Taiwan, and one of the editorial board members of at least 7 journals. Prof. Leu now organizes MCNCS and CW ECS international workshops. He is now serving as the TPC member of at least 10 international conferences. He is also an IEEE member.



Hsin-Liang Chen received his bachelor, master and Ph.D. degrees all from National Taiwan University of Science and Technology, Taiwan, in 1983, 1986 and 1991, respectively. His research interests include wireless communication, network security, Grid applications and Security. He is currently a professor of Computer Science Department, TungHai University, Taiwan, and one of the editorial board members of at least 7 journals. Prof. Leu now organizes MCNCS and CW ECS international workshops. He is now serving as the TPC member of at least 10 international conferences. He is also an IEEE member.



Chih-Chung Cheng is an undergraduate student of Computer Science Department, TungHai University, and will receive his bachelor degree in 2015. He is now preparing to apply the master program of this university.