Computer Networks 53 (2009) 1879–1902

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

An energy-efficient, transport-controlled MAC protocol for wireless sensor networks

Jaesub Kim [*], Kyu Ho Park

Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST), 373-1 Guseong-dong, Yuseong-gu, 305-701 Daejeon, Republic of Korea

A R T I C L E   I N F O

Article history:
Received 9 March 2008
Received in revised form 2 March 2009
Accepted 2 March 2009
Available online 12 March 2009

Responsible Editor: X.S. Shen

Keywords:
Wireless sensor network
Medium access control
Transport control
Congestion control

A B S T R A C T

In wireless sensor networks (WSNs), one major cause of wasted energy is that the wireless network interface is always on to accept possible traffic. Many medium access control (MAC) protocols therefore adopted a periodic listen-and-sleep scheme to save energy, at sacrifice of end-to-end latency and throughput. Another cause is packet dropping due to network congestion, necessitating a lightweight transport protocol for WSNs. In this paper, we suggest a transport-controlled MAC protocol (TC-MAC) that combines the transport protocol into the MAC protocol with the aims of achieving high performance as well as energy efficiency in multi-hop forwarding. Although TC-MAC also works through a periodic listen-and-sleep scheme, it lowers end-to-end latency by reserving data forwarding schedules across multi-hop nodes during the listen period and by forwarding data during the sleep period, all while increasing throughput by piggybacking the subsequent data forwarding schedule on current data transmissions and forwarding data consecutively. In addition, TC-MAC gives a fairness-aware lightweight transport control mechanism based on benefits of using the MAC-layer information. The results show that TC-MAC performs as well as an 802.11-like MAC in end-to-end latency and throughput, and is more efficient than S-MAC in energy consumption, with the additional advantage of supporting fairness-aware congestion control.

© 2009 Elsevier B.V. All rights reserved.

1. Introduction

Advances in the hardware technologies of microprocessors, memory, and wireless communication have made the wireless sensor network (WSN) [1] more popular. The WSN enables a wide range of emerging applications such as environmental monitoring, mobile target tracking, smart space, and ubiquitous computing. However, the most noteworthy limitation of the WSN is that every sensor node is operated by a built-in battery, which is hard to recharge or replace. Reducing unnecessary energy consumption is therefore a very important challenge that has been addressed at each layer of the protocol stack. Our specific focus is the medium access control (MAC) because MAC can control the on and off states of the wireless interface to enhance energy efficiency.

In typical WSN applications, the detection of events occurs sporadically. Hence, each sensor node spends most of its time in an idle listening state to receive possible event packets that are not actually sent. Since this idle listening state consumes as much energy as the receiving state [2,20], the idle listening accounts for most of the wasted energy in a wireless interface. Traditional MACs such as sensor-MAC (S-MAC) [3,8] and Timeout-MAC (T-MAC) [4] reduce idle listening considerably by adopting a periodic listen-and-sleep scheme. However, the periodic listen-and-sleep scheme causes a long latency problem when coupled with multi-hop forwarding, as shown in Fig. 1. Node A forwards data to node B during the first listen period but node B cannot forward data to node C because the listen

* Corresponding author. Tel.: +82 42 869 5425; fax: +82 42 869 8025.
E-mail addresses: jskim@core.kaist.ac.kr (J. Kim), kpark@ee.kaist.ac.kr (K.H. Park).

1389-1286/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2009.03.002

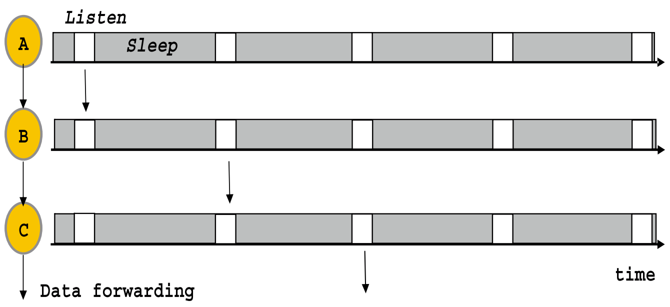


Fig. 1. A long latency problem with a periodic listen-and-sleep scheme.

period is over; hence, node B must wait for the next listen period. This type of delay accumulates at each hop in the forwarding path and causes a very long latency as the forwarding hop count increases. The listen period could be extended as a naive approach to relieve this problem, but that would increase energy consumption of every node. Hence, there is a trade-off between latency and energy savings, and traditional MACs for WSNs have concentrated more on energy savings. The increased latency resulting from this trade-off has the potential to produce a major problem in some applications if an important event fails to arrive at a sink or a base station in time. There are many emerging WSN applications requiring fast response, such as smart hospitals, military surveillance, intrusion detection, disaster monitoring, and real-time control.

To solve the long latency problem without sacrificing energy efficiency, we have suggested a Look-Ahead Scheduling MAC (LAS-MAC) [5]. LAS-MAC separates the channel reservation process and data transmission process of the Carrier Sense Multiple Access (CSMA) protocol, then it makes data transmission schedules across the multi-hop nodes during the listen period and transmits data by following the schedules during the sleep period; the nodes with no schedule go to sleep. LAS-MAC can thus forward data much faster.

Another problem caused by the periodic listen-and-sleep scheme is a decrease in throughput. Although WSNs generally assume very light traffic, the amount of traffic increases when events in densely deployed nodes are generated, when the number of event sources increase their reporting rate, and when events are gathered near the base station. However, a reduced listen period limits the time for data transmission, causing network congestion with queue overflows. Network congestion leads to the waste of communication resources and energy, affecting event detection reliability at the sink. Therefore, the throughput of MAC protocols must also be enhanced without sacrificing energy efficiency.

To mitigate network congestion, in addition, the sources of traffic must help by reducing their data transmission rates, so transport protocols are used. However, the traditional transport protocols used in general computer networks, such as the Transmission Control Protocol (TCP), are inadequate for WSNs. TCP is too heavy for WSNs because it provides the mechanisms such as connection establishment, end-to-end congestion control, and end-to-end error recovery. In general WSNs, a connection establishment mechanism is unnecessary for most event-driven data collection, and the end-to-end congestion control takes such a long time that it leads to high packet dropping in a small-sized buffer. Further, many WSN applications such as environment, habitat, and disaster monitoring do not require strict reliability in each data packet transmission due to data redundancy within a region, but rather require fairness among flows for reliable event detection, so an end-to-end packet error recovery mechanism is also unnecessary. The need for lightweight transport control mechanisms for WSNs therefore seems clear.

In this paper, we present a design for a transport-controlled MAC (TC-MAC) protocol targeted to provide a low end-to-end latency, high throughput, and light-weight congestion control, without sacrificing the energy efficiency earned from the periodic listen-and-sleep scheme. The periodic listen-and-sleep scheme remains one of the best ways to eliminate idle listening, and TC-MAC is also based on it. However, TC-MAC uses the listen period more efficiently by supporting multi-hop channel reservations; it then uses the sleep period for data transmission according to the prior reservations. To utilize the sleep period more efficiently, TC-MAC allows data packets to contain schedule information for subsequent data so that further data can be transmitted. TC-MAC can therefore forward a greater amount of data through more hops as compared to previous MACs using the periodic listen and sleep scheme. In addition, TC-MAC supports the congestion control mechanism in a MAC-layer with a light overhead. The MAC-layer can trace how a shared channel is used by each flow, so TC-MAC traces the channel usage information through a traffic monitor, and then regulates the transmission rate of traffic flow sources in a way that achieves fairness among traffic flows. To mitigate network congestion, TC-MAC uses backpressure for neighbor regulation and an explicit congestion notification for traffic flow source regulation as the COngestion Detection and Avoidance (CODA) [6] protocol does.

The rest of this paper is structured as follows. In Section 2, we survey several related works. In Section 3, we describe our basic design and the operations of TC-MAC. In Section 4, we show a simple queuing analysis of TC-MAC in a multi-hop chain topology. In Section 5, we discuss the scalability of TC-MAC. In Section 6 we present the performance results of TC-MAC and compare them with those of other MAC protocols. Finally, in Section 7 we present our conclusions.

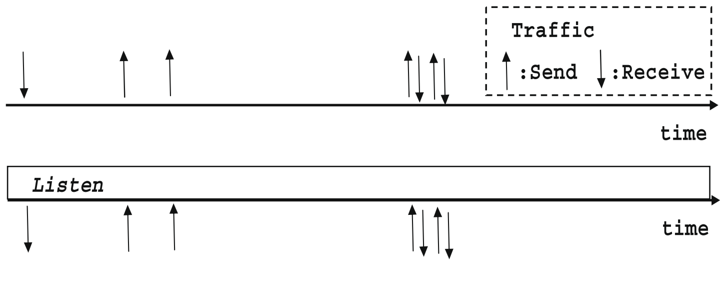
2. Related work

One of the representative wireless MAC protocols is the IEEE 802.11 MAC [7], which is a contention-based protocol that uses the carrier sense multiple access (CSMA) scheme. This protocol is very simple and scalable with large networks but it has a long idle listening time with a light traffic load. As shown in Fig. 2a, the IEEE 802.11 MAC is always awake and responds to traffic immediately, though a considerable amount of energy is wasted by the idle listening. Another popular approach is the Time Division Multiple Access (TDMA) scheme, where data transmissions of all nodes are pre-scheduled. TDMA scheme eliminates the idle listening and the packet collisions because only the nodes that have schedules awaken. However, the TDMA scheme is not scalable because of the complexity of its tasks, such as managing clusters, synchronizing groups, and scheduling dynamic slots. Whereas WSN applications require scalability and energy efficiency, the traditional MAC protocols put more weight on throughput, latency, and fairness.

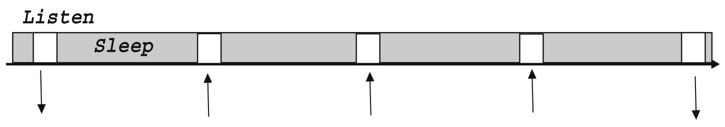
The sensor-MAC (S-MAC) [3], which is a combination of the CSMA an TDMA scheme, was designed for WSNs. S-MAC listens and sleeps periodically to reduce the idle listening time and, as shown in Fig. 2b, it delays its packet transmission until the following listen period. Each node accesses the shared medium by contention during the listen period. However, because the nodes may have different listen-and-sleep schedules, the nodes that need communications must know their neighbor's schedules and wait for them. For easy broadcasting and communication with neighbor nodes, S-MAC suggests a *virtual cluster* scheme. The virtual clusters are organized by the periodic broadcasting of synchronization packets. The nodes in the same virtual cluster listen and sleep together and communicate without waiting.

S-MAC uses a fixed duty cycle of listening and sleeping but Timeout-MAC (T-MAC) [4], as shown in Fig. 2c, uses an adaptive duty cycle to accommodate variable traffic loads. From the beginning of the listen period, the nodes keep listening for the predetermined timeout interval, Time Active (TA), and extend the listening time by TA for some activation events, such as the reception of any data, the sensing of communications, and the ending of a neighbor's data exchange. T-MAC improves the long latency problem in multi-hop forwarding by adaptively extending the listen period; however, it can forward only one hop more than S-MAC per period. Fig. 3 shows that data is forwarded from node A to node B with a limited transmission range of a wireless radio, and that node C can then extend its listen period by overhearing node B's CTS or ACK transmissions. Node C can therefore receive data when node B forwards data to node C. However, in what is called an *early sleep* problem in T-MAC, node D goes to sleep because node D cannot hear node B's transmissions. As a result, node D cannot receive data from node C. To keep node D awake, node C sends the a Future-Request-To-Send (FRTS) packet to node D when it overhears a CTS packet destined for another node. Therefore, T-MAC with FRTS packets can forward two hops more than S-MAC.

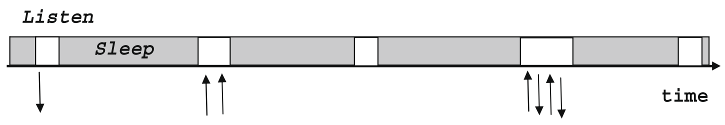
S-MAC with *adaptive listening* [8] also notices the long latency problem with multi-hop forwarding and uses a similar technique as T-MAC to extend the listen period. In S-MAC with adaptive listening, the neighbor nodes know the end of a transmission by overhearing a Request-To-Send (RTS) or a Clear-To-Send (CTS); they then



(a) IEEE 802.11 MAC.



(b) S-MAC.



(c) T-MAC and S-MAC with *adaptive listening*.

Fig. 2. Operations of related works.

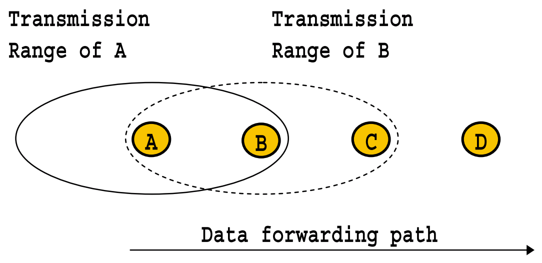


Fig. 3. Multi-hop forwarding with a limited transmission range.

wake up for a short period at the end of the transmission to receive possible traffic. In Fig. 3, node C overhears a CTS from node B and goes to sleep, and that, at the end of node A's data transmission to node B, node C then wakes up and receive data from node B. As a result, S-MAC with adaptive listening can forward one hop more than S-MAC.

Recently, the Routing-enhanced duty-cycle MAC (RMAC) [9] was proposed for low latency and energy efficiency. RMAC also relies on the periodic listen-and-sleep scheme for energy efficiency. For the low latency, RMAC transmits a setup control frame across multiple hops and schedules the upcoming data packet delivery along that route. Each intermediate node along the data packet delivery route sleeps and wakes at a scheduled time to forward data. However, RMAC has a long latency when packet transmission errors occur, because RMAC simply resumes the packet transmission in the next listen period without any immediate packet retransmission.

IEEE 802.15.4 [10] is a simple MAC protocol for lightweight WSNs. It defines two different types of device based on their function: the Full Function Device (FFD) that can be a network coordinator and talk to any other device, and the Reduced Function Device (RFD) that can talk only to the network coordinator, so that data generated from the RFD can be forwarded to the sink through network coordinators. In this scheme, however, the network coordinator also listens and sleeps periodically by sending beacons to save energy, so the long latency problem in multi-hop forwarding persists.

There are several transport protocols for WSNs [11,22,21]. COngestion Detection and Avoidance (CODA) [6], as the most related work in terms of congestion control, is an energy efficient congestion control protocol which contains congestion detection, open-loop hop-by-hop backpressure, and closed-loop end-to-end multi-source regulation. Channel loading information is sampled for more accurate congestion detection. Backpressure targets for local congestion, whereas the closed-loop regulation targets persistent congestion. However, CODA did not consider much about fairness in wireless networks.

3. Transport-controlled medium access control protocol (TC-MAC)

3.1. Motivation

In WSNs, each node detects events and forwards them to a single or multiple base stations. As the scale of WSNs increase, the amount of long-distance multi-hop forwarding and the amount of traffic increase considerably. Traditional MAC protocols [3,8,4] for WSNs concentrated only on energy efficiency and paid scant attention to their operations with multi-hop forwarding, so they encountered the long latency problem shown in Fig. 1. As a solution for the long latency problem, we have suggested LAS-MAC [5], which reserves multi-hop channels in advance using the short listen period and then transmits data using a relatively long sleep period. However, LAS-MAC did not address the increased traffic in large WSNs. Such increased traffic requires a congestion control scheme as well as high throughput. Therefore, we now suggest design for a transport-controlled MAC (TC-MAC) protocol.

To enhance throughput in the periodic listen-and-sleep scheme, the sleep period must be used more efficiently. However, nodes cannot reserve schedules for subsequent data forwarding because the relaying nodes in the forwarding path are sleeping after the execution of their schedules. In WSNs, however, most traffic goes to the same destination due to a limited number of base stations, so one flow's reserved multi-hop channel schedules can be reused with a time shift if subsequent data has the same destination. Taking advantage of this traffic characteristic to increase throughput, TC-MAC piggybacks subsequent data channel schedule onto data transmission if the subsequent data is headed for the same destination.

In WSNs, nodes are scattered in a wide geographical area, and the sink wants to get a fair amount of event data from all the sensor nodes without network congestion. However, due to a lack of precise congestion indication and coordination among flows competing for the shared wireless medium, the traditional transport protocols fail to allocate resources efficiently and fairly among flows [12,23–25]. Fig. 4 shows a situation where node C has congestion. Traditional data networks often used queue management of a single node for congestion control; in a shared wireless medium, however, congestion in an area involves multiple nodes. Nodes B, I, and D are in transmission range of node C, so traffic flowing through those nodes competes with node C even though some traffic flows do not pass through node C. Thus, to control this congestion, traffic sources which pass through the nodes interfering with node C must also be reduced. A MAC-layer can easily get this shared channel usage information and use it for fair congestion control with cross-layer design and information such as the length of an interface queue (IFQ) and the original source addresses of received data packets. We therefore added essential mechanisms for transport control to the MAC layer in an energy-efficient way, which are explained in Section 3.4.

3.2. Basic operations

This section introduces basic operations of TC-MAC for low end-to-end latency and high energy efficiency. TC-MAC works through the periodic listen-and-sleep scheme, while it reserves multi-hop channel schedules during the listen period and forwards data through multiple hops during the sleep period. We assume that the radio has a fixed coverage, and that sensor nodes are in the same virtual cluster [3], so they listen and sleep together.

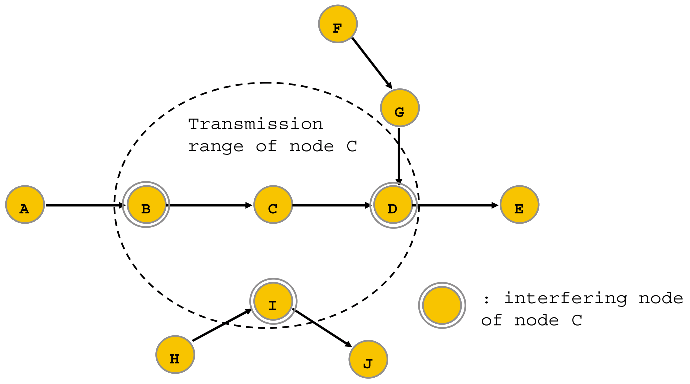


Fig. 4. Interfering nodes in a shared wireless medium.

3.2.1. LAS(Look-Ahead Schedule)-RTS

Previous approaches based on the IEEE 802.11 MAC [3,8,4] use the RTS and CTS exchange for the channel reservation of one-hop nodes. However, TC-MAC requires channel reservations of multi-hop nodes; thus, TC-MAC uses the LAS-RTS instead of the RTS and the CTS. The LAS-RTS, as depicted in Fig. 5, contains a destination and a schedule fields in addition to the previous RTS. The destination field contains a final destination address for forwarding the data and the schedule field (send-time) contains when to send the data. As the LAS-RTS is forwarded, the next hop to forward is determined from the destination with cross-layer routing information, and the schedules of action are reserved on the basis of the schedule and duration information. The *duration* field originally represents the total time required to transmit data, including data and control packets. In TC-MAC, however, the duration only represents the duration of a data packet transmission because TC-MAC omits an explicit ACK packet in multi-hop data forwarding. This is explained in Section 3.2.2.

Our protocol requires synchronization among nodes in the forwarding path to execute the data forwarding schedules. To be robust to synchronization errors, the send-time in the LAS-RTS is relative rather than absolute and set to the value within current cycle. Therefore, clock drift between the LAS-RTS transmission and the execution of the corresponding schedule is negligible.

3.2.2. Multi-hop channel reservation and data transmission

At the beginning of the listen period, nodes that have data to forward transmit LAS-RTS with a final destination address and a send-time of corresponding data. The next forward node that receives the LAS-RTS reserves future channel schedules corresponding to that of the previous node. For example, the next node must reserve a *Receive* schedule corresponding to a *Send* schedule of the previous node. The LAS-RTS is then forwarded to its next node with a newly calculated send-time of data, and the previous node hears this LAS-RTS and regards this LAS-RTS as an acknowledgment of its schedules, which substitutes for the CTS. Due to this dual function of the LAS-RTS, the LAS-RTS can be forwarded more hops in a given short listen period. The LAS-RTS is forwarded until the end of listen period or until the LAS-RTS reaches its final destination. The multi-hop channel reservations for a period are then finished.

The traditional MACs based the IEEE 802.11 MAC reserve channels only for one-hop forwarding by exchanging the RTS/CTS; the duration field of the RTS/CTS contains the total time necessary for one-hop forwarding. However, TC-MAC needs a different unit of a channel reservation for multi-hop forwarding. Therefore, we made an efficient multi-hop channel schedule unit (MCSU) by observing the behavior of multi-hop forwarding. Nodes that receives the LAS-RTS reserve multi-hop schedule unit.

Fig. 6 shows required multi-hop channel schedule for data to be forwarded from node A to node G. When node D forwards data to node E, node D should be in the *Send* schedule and node E should be in the *Receive* schedule. At the same time, by using the broadcasting characteristic of wireless radio, node C should be in the *ACK* schedule to hear node D's data transmission, which is then regarded as the ACK. Node B and node F should be in the *Nav* (Network Allocation Vector [7,19]) schedule because they are the hidden terminals of node D. As data is forwarded from nodes E, F and G, the same pattern is repeated, and each node in the forwarding path has the same pattern of schedules, namely the sequence of N (*NAV*), R (*Receive*), S (*Send*), A (*ACK*). TC-MAC consequently uses $\{N, R, S, A, N\}$ as the MCSU, whose duration is five times of data packet trans-

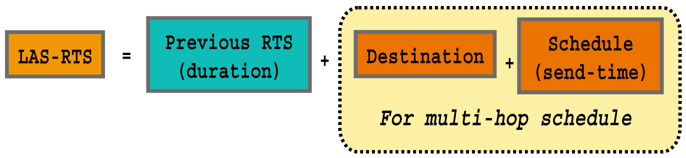


Fig. 5. LAS(Look-Ahead Schedule)-RTS.

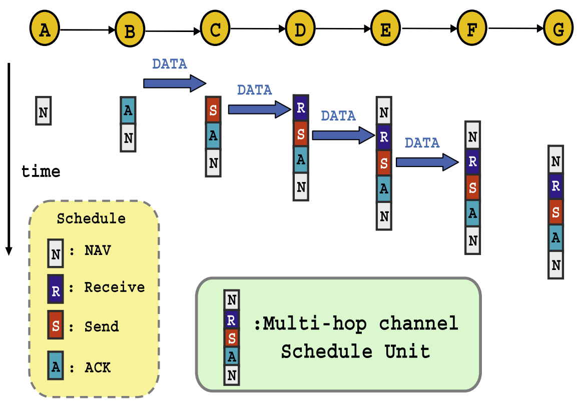


Fig. 6. A multi-hop channel schedule unit for a multi-hop channel reservation.

mission time. TC-MAC reserves the MCSU on the basis of the schedule information, the send-time of data, in the LAS-RTS. The start time of the *R* (*Receive*) schedule in $\{N,R,S,A,N\}$ should correspond to the previous node's send-time of data.

Fig. 7 depicts the process of the multi-hop channel reservation and data transmission in TC-MAC. The vertical bars on each node represent the flow of time and the corresponding schedules. During the listen period, the starting node of the multi-hop forwarding, node A, decides the future send-time of the data, *S* (*Send*), and reserves $\{S,A,N\}$ based on its data packet duration, which includes a margin time of radio propagation delay. Nodes with data to send can determine the send-time of data within the current period, but there is a minimum value for the send-time of the data, $3 * T_{LAS-RTS}$, where $T_{LAS-RTS}$ is the duration of the LAS-RTS transmission. This minimum value is used to avoid collisions with the preceding LAS-RTS. Thus, the data is not transmitted until the LAS-RTS is more than three hops away.

Node A then forwards the LAS-RTS with the schedule (send-time), destination (node E), and data duration to its next forward node, which is determined by the destination with cross-layer routing information. After then, node A waits for its next forward node's LAS-RTS transmission to confirm its schedules. The next node reserves the MCSU ($\{N,R,S,A,N\}$) from the received LAS-RTS, whose *R* (*Receive*) schedule should correspond to the node A's *S* (*Send*) schedule, and then forwards the LAS-RTS with its new send-time of the data. At the same time, the previous node overhears the LAS-RTS as the CTS and confirms its forwarding schedules. Nodes other than the starting node reserve the MCSUs from the LAS-RTS, and their MSCUs are shifted by one data duration as the LAS-RTS is forwarded. This procedure is repeated with the LAS-RTS forwarding. The last node, node E, needs to transmit an explicit LAS-RTS as the CTS and reserve $\{N,R,A\}$ instead of the MCSU. A node withdraws its forwarding schedule by changing its MCSU to $\{N,R,A\}$ when the LAS-RTS reaches LAS-RTS's final destination, when the LAS-RTS cannot be forwarded since the listen period is over, or when a node cannot receive the LAS-RTS as the CTS and cannot confirm its forwarding schedule. As a result, all the reserved schedules across multi-hop nodes are shown in Fig. 7, which enable pipelined data forwarding.

Additionally, numerous neighbor nodes exist around a forwarding path, and they must not reserve schedules disturbing other forwarding traffic. Thus, if the neighbor nodes receive the LAS-RTS that is not destined to them, they must reserve $\{N,N,N\}$ corresponding to the send-time on the LAS-RTS. Namely, they have to be in the *N* schedules for neighbor node's *R*, *S*, and *A* schedules.

After the multi-hop reservation process is finished, the nodes wake up at their scheduled time and do the corresponding data forwarding jobs. This is the data transmission process. The nodes go to sleep except when it needs essential operations such as *S*, *R*, and *A*. During the *N* schedule, the nodes also go to sleep because they have nothing to do. During the *A* schedule, the nodes are awake only for

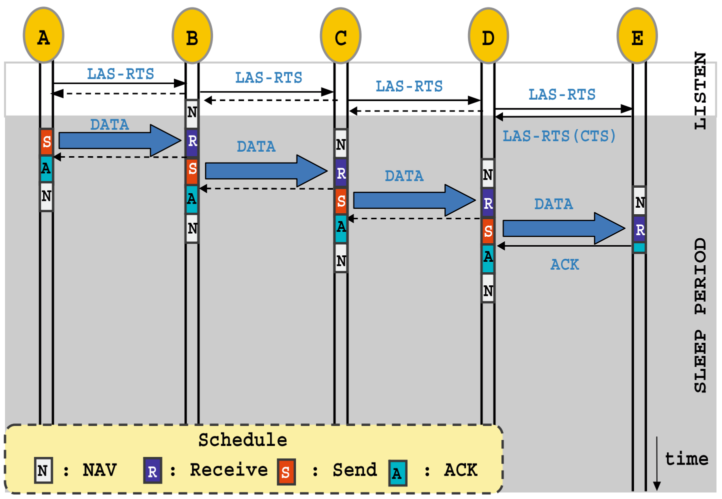


Fig. 7. Multi-hop channel reservation and data transmission in TC-MAC.

as long as they need to confirm the next forward node's data transmission as an ACK, or to send an explicit ACK packet in case of the last node of multi-hop forwarding.

In this way, TC-MAC can forward many hops during a period, so the end-to-end latency is reduced. Also, TC-MAC achieves energy efficiency by reducing unnecessary control packets such as the CTS and ACK packets and by trying to sleep except when essential operations are needed.

3.2.3. Schedule Collision in LAS-RTS transmission

With multiple traffic flows, the forwarding paths of them can cross over; they are called cross traffic. As the number of traffic sources increases, the amount of cross traffic also increases, and TC-MAC therefore needs ways of coping with cross traffic. At the beginning of the channel reservation process, the starting nodes of traffic flows freely determine their send-time of data and forward the LAS-RTS. However, some traffic flows may cross over at certain nodes and request over-lapping schedules. This conflict of schedules is called a *schedule collision*.

Fig. 8 shows how TC-MAC solves a schedule collision with cross traffic. One traffic flow goes from node A to node E, and an another cross traffic flow goes from node F to node G. The two traffic flows cross over at node D. First, node F of the cross traffic flow transmits the LAS-RTS to node D, and then node D reserves the MCSU ($\{N,R,S,A,N\}$) and forwards the LAS-RTS to node G. At this point, the neighbors of node D, node C and node E, overhear this LAS-RTS and reserve $\{N,N,N\}$. Node A also forwards the LAS-RTS and reserves channel schedules, but a schedule collision occurs at node C with the prior schedules ($\{N,N,N\}$) which have been reserved by node D. Node C then signals a failure of schedule reservations through LAS-Negative-Acknowledgement (LAS-NAK), which is identical to the LAS-RTS except that the schedule field is filled with a *bitmap schedule*. The bitmap schedule is a summary of all schedules already reserved at a node. Each bit takes charge of the fixed quantum of time (sleep period/bitmap size) and is set when a corresponding schedule exists. The reason a bitmap schedule is transmitted to node B is to make node B reschedule without additional schedule collisions. As soon as node B receives LAS-NAK, node B withdraws its forwarding schedule by changing $\{N,R,S,A,N\}$ to $\{N,R,A\}$. Node B then retransmits the LAS-RTS with a newly arranged schedule ($\{S,A,N\}$) from the bitmap schedule, thereby avoiding a data transmission collision.

3.2.4. Transmission error recovery

TC-MAC reserves schedules and forwards data in a pipelined way. Therefore, if there is a transmission error in an intermediate node, all the following schedules in the forwarding path will fail.

To avoid chain errors of data transmission, we suggest a *schedule shift* mechanism for the data transmission error recovery, as shown in Fig. 9. The basic idea of the schedule shift mechanism is that the failures of pipelined schedules from a data transmission error can be recovered by shifting schedules in the following nodes by the same amount of time; that is, $T_{shift} = T_{data} + T_{ack}$, where T_{data} is the duration of the data transmission and T_{ack} is the duration of ACK transmission. The T_{shift} is the minimum time necessary for both the sender and the receiver to notice the failure of data transmission.

At the left-hand side of Fig. 9, before t_1, we can see that node A is sending data to node B at t_0, but a transmission error occurs. Node B finds this error by receiving the corrupted packet or nothing at the R schedule; it then comes back to the beginning of the R schedule and shifts its schedule by T_{shift}. Next, node C finds an error at the beginning of the R schedule because it cannot receive anything; it then

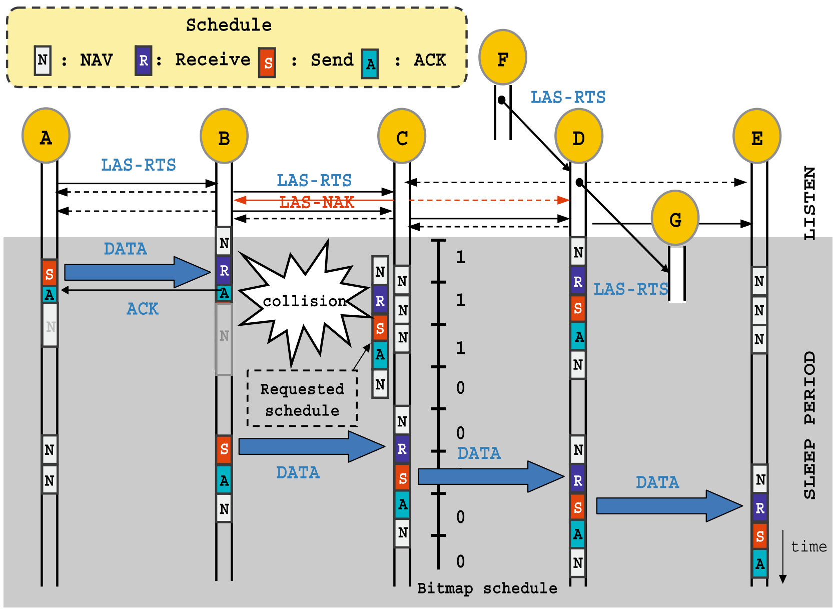


Fig. 8. Schedule collision in LAS-RTS transmission at node C during the listen period.

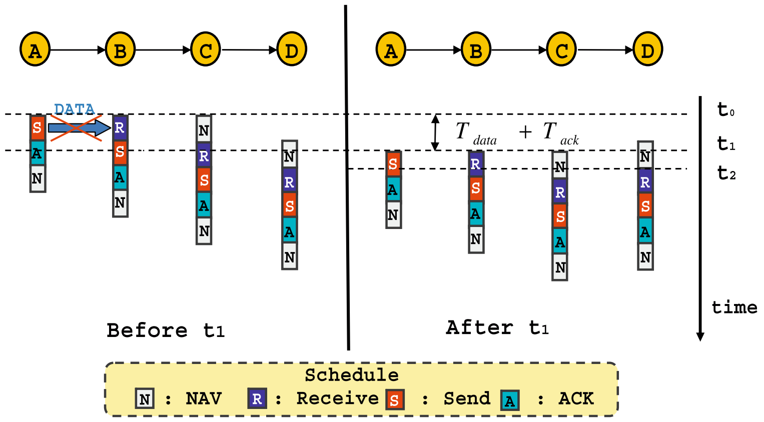


Fig. 9. Data transmission error recovery by means of *Schedule shift* mechanism.

comes back to the *R* schedule and shifts its schedule. For this schedule shift at the beginning of the R schedule, a time-out mechanism is necessary. There is no specific value for the time-out, but a short time-out value is preferred for energy saving. If TC-MAC does not receive anything for a short time interval from the beginning of the R schedule, it shifts its schedules and goes to sleep. At t_1, node A finds its data transmission error when it cannot receive an ACK, the next forward node's data transmission, during the T_{ack} time of the *A* schedule; it then shifts its schedules.

As a result, after t_1, the shifted schedules are represented at the right-hand side of Fig. 9, and the data is retransmitted at node A. Likewise, node D shifts its schedule at t_2 because no data is being received in the *R* schedule, and all the following nodes shift their schedules consecutively for the same reason. In this way, multi-hop schedules can be reused without a new expensive channel reservation process. However, too many schedule shifts consume the extra energy because all the following nodes have to shift their schedules; thus, the number of the schedule shift is limited. The schedule shift is canceled if the shifted schedules reach the other schedules (a schedule collision), or the end of current period; the data transmission then resumes at the next period.

3.2.5. State transition diagram of the basic operations

To summarize, we represented a brief state transition diagram of the basic operations of TC-MAC, as shown in Fig. 10. The left side represents the multi-hop channel reservation process and the right side represents the data transmission process. Basically, TC-MAC repeatedly alternates between sleep and listen states as sleep or listen timers expire, and goes to a listen state when a Look-Ahead Scheduling (LAS) timer which manages the reserved

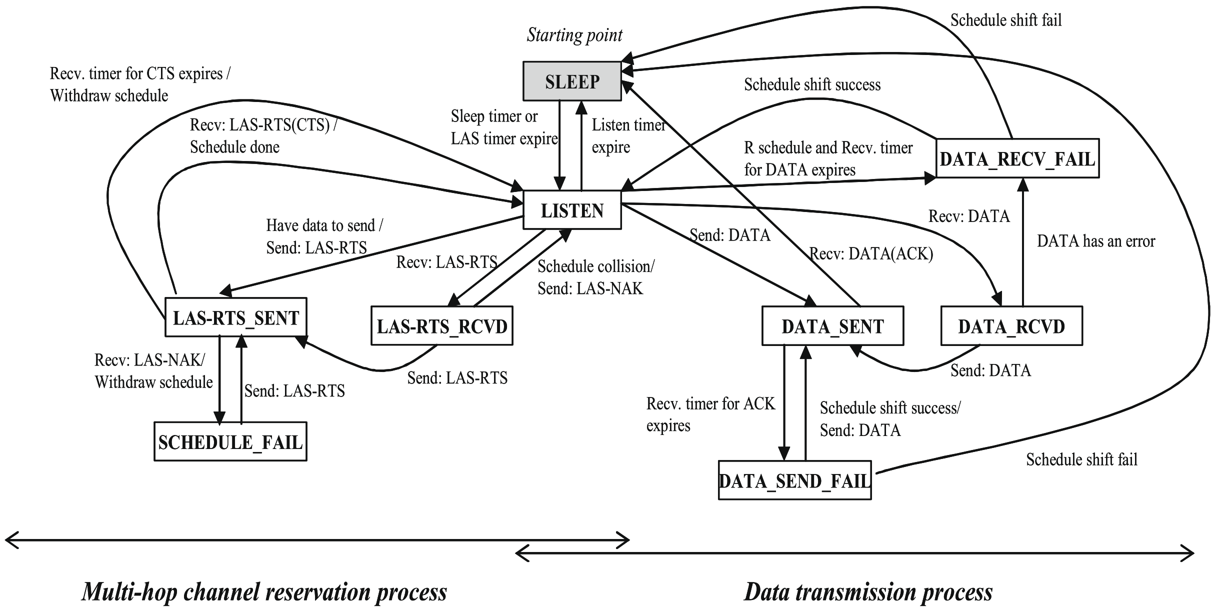

Starting point
SLEEP
LISTEN
LAS-RTS_SENT
LAS-RTS_RCVD
SCHEDULE_FAIL
DATA_SENT
DATA_RCVD
DATA_RECV_FAIL
DATA_SEND_FAIL
Schedule shift fail
Schedule shift success
Recv. timer for CTS expires / Withdraw schedule
Recv: LAS-RTS(CTS) / Schedule done
Sleep timer or LAS timer expire
Listen timer expire
R schedule and Recv. timer for DATA expires
Have data to send / Send: LAS-RTS
Recv: LAS-RTS
Schedule collision/ Send: LAS-NAK
Send: DATA
Recv: DATA(ACK)
Recv: DATA
DATA has an error
Recv: LAS-NAK/ Withdraw schedule
Send: LAS-RTS
Send: LAS-RTS
Send: DATA
Recv. timer for ACK expires
Schedule shift success/ Send: DATA
Schedule shift fail
Multi-hop channel reservation process
Data transmission process

Fig. 10. A brief state transition diagram of the basic operations.

schedules expires. A receive timer is used for a timeout of expected packets, such as the LAS-RTS as CTS or the ACK.

3.3. Throughput enhancement mechanisms

TC-MAC reserves the channels in multi-hop nodes during the listen period to reduce latency. However, TC-MAC with only the basic operations explained in 3.2 does not have high throughput because it reserves the forwarding channels only during the listen period. Our TC-MAC, therefore, piggybacks the channel reservation information of subsequent data on current data transmissions to reserve multi-hop channels even during the sleep period.

3.3.1. Channel reservation in data transmission

The WSN generally assumes that one or a small number of base stations exist to collect sensed data. This means a lot of traffic goes through the same routing paths to the same base stations, and data destined for the same destination can be forwarded by using the multi-hop schedules of the previous data transmission. By exploiting this, TC-MAC reserves channels for subsequent data even during the sleep period so long as subsequent data are destined for the same destination.

Fig. 11 shows channel reservation in data transmission. Before forwarding the first data, node A finds that subsequent data in the interface queue (IFQ) is also bound for the same destination. Then, node A reserves schedule for subsequent data by cloning the first data's schedule, and forwards the first data to node B with subsequent data's schedule information (a send-time of the data). Node B then clones its previous forwarding schedule based on the received schedule information and forwards its data with its subsequent data's schedule. This procedure is repeated to node E, and repeated so long as data in IFQ are destined for the same destination. In this way, TC-MAC can forward a large amount of data in one period without an explicit channel reservation in the listen period.

To reserve multi-hop channels, by the way, information such as duration, destination, and schedule (send-time) of the corresponding data is necessary as the LAS-RTS has; thus TC-MAC added these information fields to the header of the data packet.

3.3.2. Channel reservation sharing

As traffic goes close to the base station, many traffic flows are joined and compete for their channel reservations during the listen period. Even in a single flow, data can be stored in the middle nodes of the forwarding path; thus, the multiple nodes with data to forward compete for their channel reservation. However, if their data are bound for the same destination, they can share their channel schedules instead of reserving their own. Such channel reservation sharing avoids unnecessary channel competition and energy consumption.

As an example, Fig. 12 shows two traffic flows destined to node E. At the beginning of the listen period, *flow1* forwards the LAS-RTS first and reserves multi-hop channels to node E, and then *flow2* forwards the LAS-RTS to node E but finds the already-reserved multi-hop channels to node E at node B. At this point, node B no longer needs to forward the LAS-RTS because data from *flow2* can be queued at node B and forwarded by using *flow1*'s reserved channels in the way explained in Section 3.3.1.

In more detail, let us assume that *flow1* and *flow2* in Fig. 12 have one data packet to forward with the same LAS-RTS forwarding explained above. If *flow2*'s data is scheduled to arrive at node B before *flow1*'s, as shown in Fig. 13, node B stores *flow2*'s data to its IFQ first and then stores *flow1*'s. On the first *S* schedule of node B, node B sends the first data in the IFQ, *flow2*'s data, and reserves the next channel starting from node B for *flow1*'s data in the IFQ; that is, the middle nodes of the schedule path can reserve their own schedule starting from them if the previous nodes do not make further reservations. Conversely, if *flow1*'s data is scheduled to arrive before *flow2*'s,

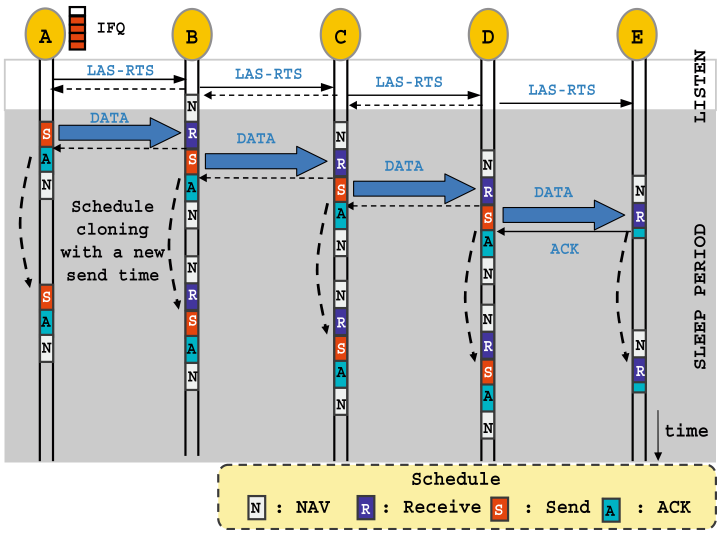


Fig. 11. Channel reservation in data transmission.

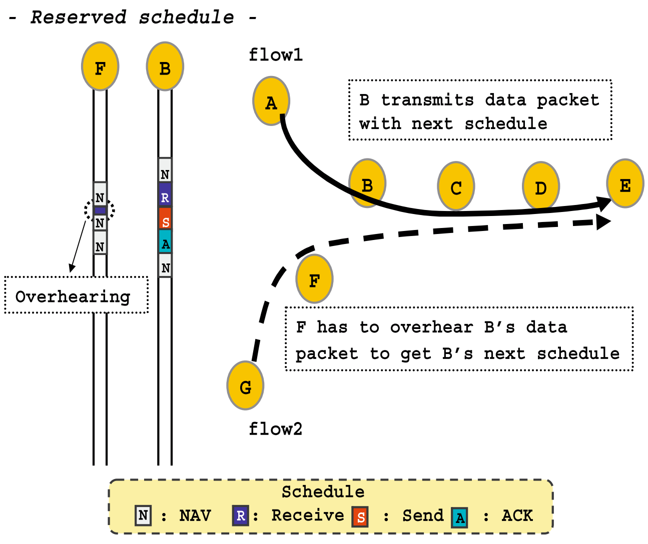

- Reserved schedule -
F
B
N
R
S
A
Overhearing
flow1
A
B transmits data packet with next schedule
B
C
D
E
F
F has to overhear B's data packet to get B's next schedule
G
flow2
Schedule
N : NAV R : Receive S : Send A : ACK

Fig. 12. Channel reservation with other traffic.

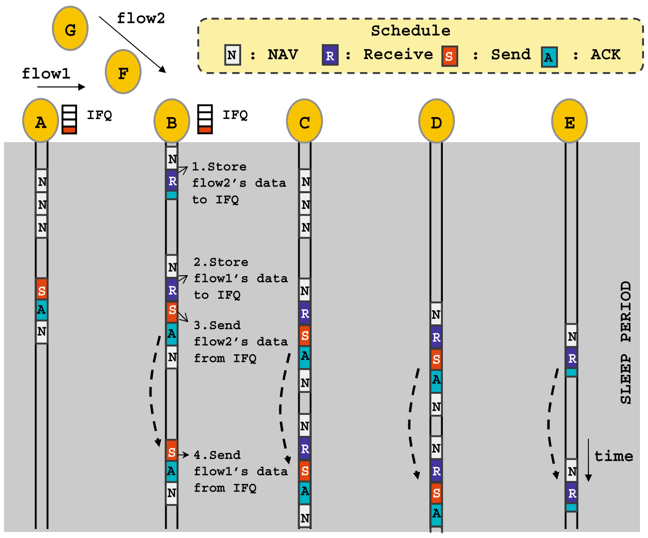

G
flow2
flow1
F
Schedule
N : NAV R : Receive S : Send A : ACK
A
IFQ
B
IFQ
C
D
E
1.Store flow2's data to IFQ
2.Store flow1's data to IFQ
3.Send flow2's data from IFQ
4.Send flow1's data from IFQ
SLEEP PERIOD
time

Fig. 13. Channel reservation sharing with other traffic.

node B finds no data in the IFQ at *flow1*'s data transmission, and then cannot reserve the channel schedule for *flow2*'s. To address this problem, we define a *virtual packet* which represents a packet that is scheduled to arrive but has not yet arrived, and make the nodes reserve channel schedule for the next data whenever there are virtual packets as well as packets in the IFQ. Therefore, node B can find *flow2*'s virtual packet at *flow1*'s data transmission and reserve the channel for *flow2*. Node B must reserve the channel schedule for *flow2* later than the arrival schedule of the virtual packet.

With more than one data packet in *flow1* and *flow2*, node B receives data from both flows and stores them in its IFQ, and then forwards them in order at every *S* schedule with channel reservation of subsequent data.

3.3.3. *Schedule collision in data transmission*

Schedule collisions also occur when multi-hop channel schedules are reserved in data transmissions during the sleep period. If multi-hop schedules are reserved by the LAS-RTS during the listen period, nodes can easily get their neighbors' reserved channel information and use it to avoid packet or schedule collisions. However, if multi-hop schedules are reserved by data transmissions during the sleep period, all the neighbors of data transmitting node are sleeping; thus, the neighbors cannot get the newly reserved schedules for subsequent data forwarding. For this reason, the neighbors of data transmitting node must wake and overhear the channel reservations of the data transmitting node. However, not all neighbors need to wake. Only the neighbors with the subsequent data to send need to wake because they should reserve their schedules for the subsequent data in such a way to avoid packet or schedule collisions.

As shown in Fig. 12, when node B has the *S* schedule, neighbor F has the *N* schedule. If node F has the next data to send, it has to wake up at the second *N* schedule corresponding to the *S* schedule of node B, and overhear node B's data transmission to get the next channel reservation information. At that time, node F only needs to overhear the data header containing the next channel schedule information instead of the whole data packet. However, node F may not be able to get the next channel schedule information of node B since the *N* schedule does not ensure receipt of a packet. In this case, node F gives up its next data transmission, and tries it again in the next period.

When a schedule collision occurs at the channel reservation in data transmission, there is no given time for schedule collision handling. TC-MAC thus uses the *S* schedule for schedule collision handling instead of sending data. Fig. 14 shows how the schedule collision in data transmission is handled. In *Step 1* of Fig. 14, as data is forwarded from node A to node C, each node reserves the next channel schedules, and a schedule collision occurs at node C because it already has $\{N,N,N\}$ from other traffic. To handle the schedule collision, node C sends the LAS-NAK to node B instead of sending data on the *S* schedule. Then, node B withdraws its forwarding schedules and reserves new schedules by sending the LAS-RTS. The transmission of LAS-NAK and LAS-RTS does not cause a packet collision because the neighbors of nodes B and C are set as the *N* schedules for the *A* and *S* schedules. After receiving the new schedule from node B, node C sends data with the new schedule after shifting its schedule, as shown in *Step 2* of Fig. 14. In this way, data can be continuously forwarded with next channel reservation in spite of the schedule collision.

3.3.4. Schedule connection mechanism

The channel reservation in TC-MAC is optimized for multi-hop data forwarding with the pipelined transmission, which eliminates the transmission of unnecessary control packets and brings efficient channel use. If a pipeline of data forwarding schedules is broken and separated, the benefits disappear and performance decreases. The pipeline is often separated when multiple nodes in the same forwarding path try to make their own pipelines and when schedule collisions occur.

For example, Fig. 15a shows data forwarding from node A to node F. In the first period, node A can only forward the LAS-RTS to node D due to the limited listen period. Therefore, data forwarded from node A is stored in node D. In the next period, node D and node A each try to reserve channel schedule. They contend to reserve their forwarding channels at node D even though their data are generated from the same source and bound for the same destination; as a result, they create separated pipelines and inefficient channel usage. To solve this problem, TC-MAC uses a *schedule connection* mechanism that connects separated pipelines. If pipelines are headed to the same destination, they can be connected into one pipeline as depicted in Fig. 15b. When node D tries to reserve the next channel schedule, node D checks whether there is a schedule such as $\{N,R,A\}$ that ends at node D with the same destination. If so, node D changes the $\{N,R,A\}$ schedule to $\{N,R,S,A,N\}$ to connect the pipelines: the $\{N,R,A\}$ schedule in Fig. 15a is changed to $\{N,R,S,A,N\}$ in Fig. 15b. Then, when node E

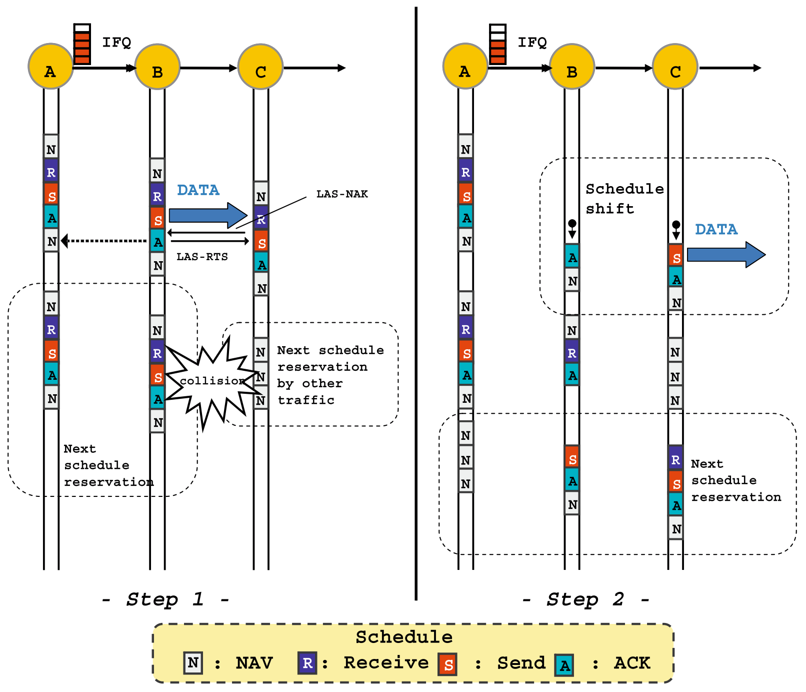
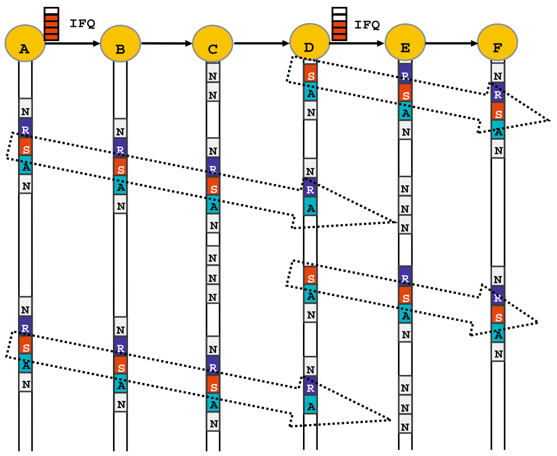
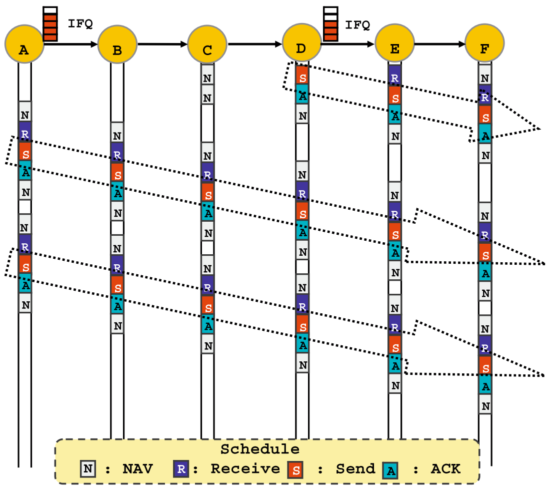


Fig. 14. Schedule collision in data transmission during the sleep period.



(a) Channel reservation without a schedule connection mechanism.



(b) Channel reservation with a schedule connection mechanism.

Fig. 15. The schedule connection mechanism.

receives the connected schedule, it discovers that the schedule collides with its prior {*N*,*N*,*N*} schedule in Fig. 15a. However, node E can know that those colliding schedules are reserved from the same node D, and then treats it as if node D wants to connect pipelines. Node E thus replaces the {*N*,*N*,*N*} schedule with the {*N*,*R*,*S*,*A*,*N*} schedule. As a result, the separated pipelines are connected, as shown in Fig. 15b, and the channels can be used more efficiently

3.4. Congestion control mechanisms

The network congestion in WSNs causes large queuing delays and high packet losses which lead to the waste of considerable energy and network resources, decreasing throughput.

To design a congestion control mechanism for our protocol, we considered two characteristics of WSNs. First, end-to-end congestion control is inadequate in WSNs because it takes such a long time that a small-sized queue easily overflows, thus wasting a lot of network resources. Moreover, that type of congestion control burdens the base station with congestion feedback because WSNs have many-to-one communication patterns. Therefore, a distributed congestion control mechanism is more appropriate. Second, the transport-layer approaches for congestion control are inadequate because WSNs operates on the wireless radio. The transport-layer only knows very abstract information such as queue size, and does not how the wireless medium is used and shared by different traffic flows; thus, it causes unfairness among flows [12,23–25]. On the other hand, the MAC-layer can easily trace this information.

Considering the above characteristics, TC-MAC implements a congestion control mechanism at the MAC-layer with a *Traffic Monitor* which traces how the wireless medium is shared by traffic flows. TC-MAC judges the network congestion in a distributed way based on the queue sizes (IFQ sizes) in the middle nodes, and controls the congestion implicitly and explicitly with the traffic monitor, through *backpressure* and *explicit congestion notification*. The data transmission rates of the immediate neighbors are controlled by backpressure, and the rate of original source nodes are controlled by explicit congestion notification (ECN) based on the traffic monitor. The ECN mechanism is not an end-to-end congestion control scheme since any middle congested node notifies its congestion to the responsible source nodes. Fig. 16 shows an overview of congestion control mechanisms used in TC-MAC.

3.4.1. Backpressure

One fast way for nodes to gain relief from the buffer overflows is to make neighbors stop transmitting data. TC-MAC thus uses backpressure to signal individual node's congestion when the IFQ size exceeds a certain threshold. For energy efficiency, backpressure is transferred implicitly by adding and setting a congestion bit in the header of control and data packets. If the adjacent nodes receive backpressure, they stop transmitting data to the source of backpressure by not reserving more channel schedule to it. Backpressure is simple and effective for local congestion. However, it is inadequate for the sort of persistent congestion which requires regulation of the original traffic sources, and it does not consider fairness among different flows.

3.4.2. Traffic monitor

For efficient and fair congestion control, TC-MAC uses a traffic monitor which traces the channel usage of traffic flows and helps to find the most dominant traffic flow. The traffic monitor manages flow sources with their channel usages, as shown in Fig. 17, and this information can be acquired from the header of the data packet that has source and channel reservation information; the channel usage is estimated by counting all the reserved channel schedules, i.e. the *R*, *S*, *A*, and *N* schedules, during a fixed interval at a node. As explained in Section 3.3.3, nodes with data to send overhear their neighbors' data packets, so their traffic monitor can easily get that information. A congested node mostly has data to send; thus, it mostly overhears neighbor's data packets.

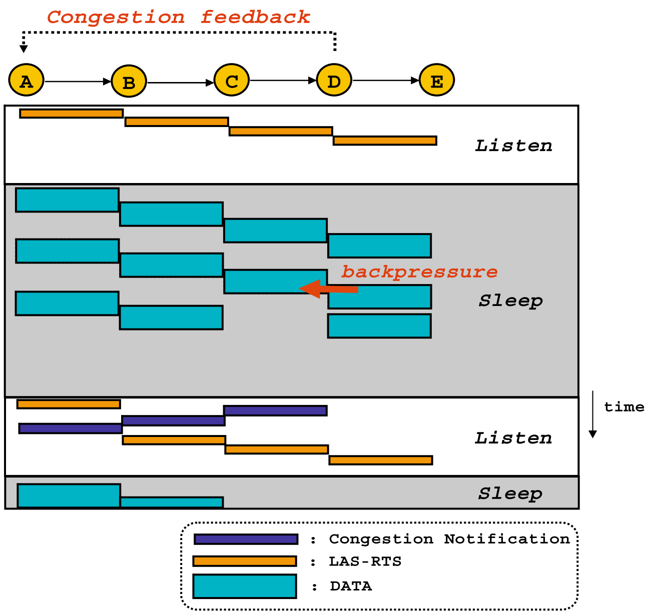
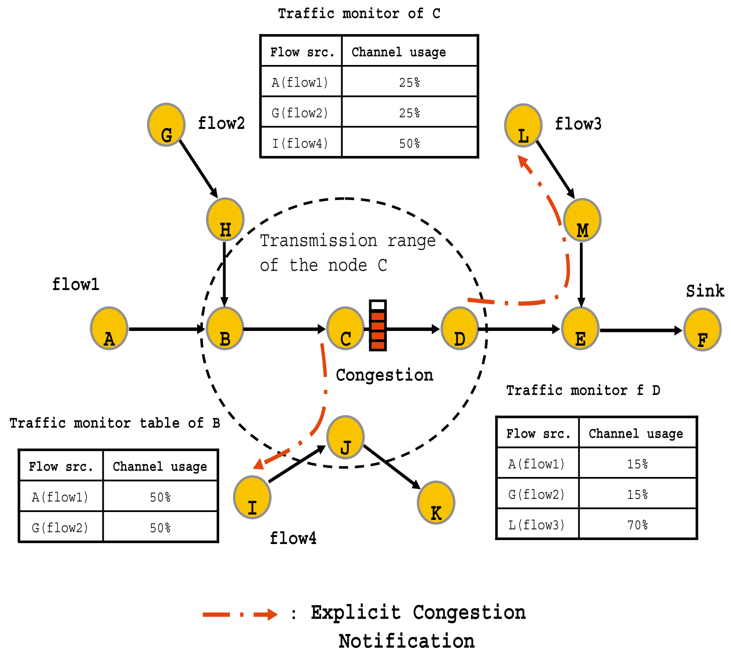


Fig. 16. Congestion control mechanisms.


flow2
flow3
flow1
Sink
G
H
L
M
A
B
C
D
E
F
J
I
K
Transmission range of the node C
Congestion
flow4
: Explicit Congestion Notification

Traffic monitor of C

Flow src.	Channel usage
A(flow1)	25%
G(flow2)	25%
I(flow4)	50%

Traffic monitor table of B

Flow src.	Channel usage
A(flow1)	50%
G(flow2)	50%

Traffic monitor f D

Flow src.	Channel usage
A(flow1)	15%
G(flow2)	15%
L(flow3)	70%

Fig. 17. Explicit congestion notification with the traffic monitor.

Fig. 17 shows some traffic monitors. In the traffic monitor at node C, it includes not only *flow1* and *flow2* passing through, but also *flow4* not passing through because the middle node J of *flow4* shares the channel with node C. When node J reserves forwarding schedules, node C reserves the corresponding *N* schedules by overhearing data packet of node J, and thus counts the channel usage of *flow4*.

3.4.3. Explicit congestion notification

Backpressure can efficiently control local congestion with little overhead. However, backpressure is insufficient for the type of persistent congestion which requires the regulations of the original sources, because it propagates too slowly. To relieve persistent congestion, TC-MAC uses an explicit congestion notification (ECN) mechanism which throttles the sending rate of traffic sources. The ECN mechanism follows the max-min fairness in allocation of the shared channel. When a node detects congestion, it finds the most channel-consuming flow from its traffic monitor and then sends a congestion notification (CN) packet to the source of the flow. The source that receives the CN packet decreases its sending rate by half; otherwise it increases its rate additively as an additive increase multiplicative decreases (AIMD) scheme operates. If the congestion persists after a certain time interval, the ECN mechanism sends another CN packet to the source of the current most channel-consuming flow. Therefore, the high rate flows receive CN packets based on their contribution to the congestion and reduce their sending rates, but the low rate flows do not. The CN packet is almost equal to the LAS-RTS packet in terms of packet size and included fields, and forwarded during the listen period for fast congestion feedback, as shown in Fig. 16. The CN packet forwarding may disturb the LAS-RTS forwarding, so the CN packet is not forwarded frequently.

The amount of channel usage is usually proportional to the sending rate of a traffic flow, but not always because of various network topologies and forwarding paths. Therefore, the fair channel allocation does not always mean the fair rate allocation of traffic flows.

In Fig. 17, for example, node C detects congestion by finding that its IFQ size exceeds a certain threshold, which is higher than that of the backpressure. Node C then sends a CN packet to the source of *flow4*, node I, because *flow4* occupies the most channel of node C. Another reason for node C's congestion is that node C's immediate neighbors can be blocked by their own neighbors; node C therefore cannot send data to its neighbors. In Fig. 17, node C cannot send data to node D if node D is blocked by node E which sends *flow3*. However, because of the limited wireless radio range, node C cannot know that the congestion is also caused by *flow3*, but node D can know this fact since it can overhear the channel reservations of both node C and node E. Therefore, if immediate neighbors of a congested node overhear a CN packet, they must check whether there are flows not passing through them with the $\{N,N,N\}$ schedules and consuming the most of their own channel. If such flows are found, the nodes must send the CN packets to the sources of such flows. In Fig. 17, node D overhears the CN packet of node C and finds, from its traffic monitor, that node D's channel is mostly used by *flow3* which is not passing through node D. Node D thus sends a CN packet to the source of *flow3*, node L. In this way, the congested node can regulate the sending rate of potentially interfering flows.

4. Queuing analysis

This section suggests a queuing analysis of TC-MAC in a multi-hop chain topology shown in Fig. 18. First, we denote the listen period by T_{listen}, the sleep period by T_{sleep}, and the offset between the schedules (MCSUs) by T_{offset}. The traffic generation rate from a node i itself is assumed to be Poisson with a rate being λ^i_{self}. As traffic is forwarded to the destination, the traffic rate of the node is the sum of traffic generation rates of the node itself and each relaying node. Then the total traffic rate on a sensor node i (λ_i) is represented as

$$\lambda_i = \lambda^i_{self} + \sum_{j=1}^{N_{relay}} \lambda^j_{self}, \qquad (1)$$

where N_{relay} is the number of relaying nodes to node i.

The link transmission rate (service rate) of TC-MAC can be derived from Fig. 18. We assume that, during the listening period of T_{listen}, the multi-hop channel schedules in the forwarding path are successfully reserved with the

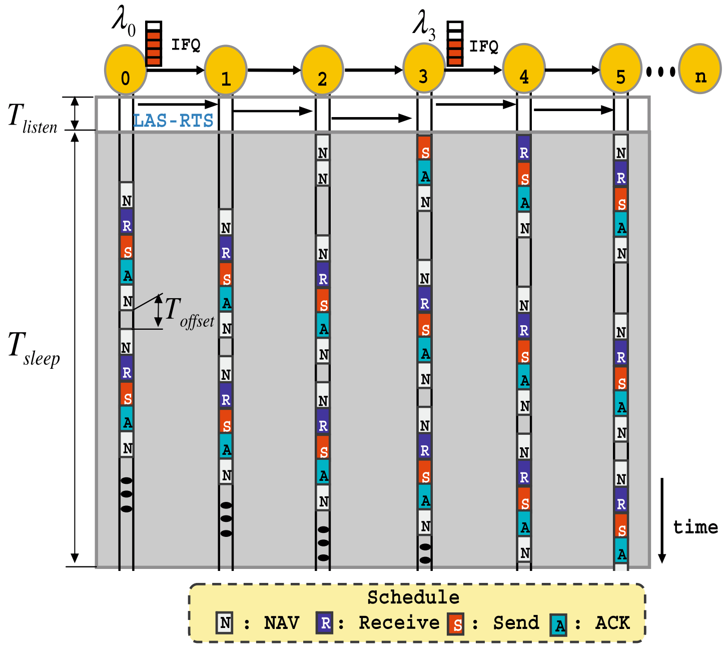


Fig. 18. Queuing network model for multi-hop forwarding.

cross-layer routing information and channel reservation sharing, and without the schedule collision. Then, traffic in the forwarding nodes can be continuously forwarded to the destination during T_{sleep}. To forward one data packet, a node needs $5T_{data} + T_{offset}$, so the link transmission rate of TC-MAC is given as

$$\mu_{TC\_MAC} = \frac{\frac{T_{sleep}}{5T_{data}+T_{offset}}}{T_{listen} + T_{sleep}} \quad (\text{packets/s}). \tag{2}$$

Traffic intensity ρ is then stated as

$$\rho = \frac{\lambda}{\mu}. \tag{3}$$

In the M/M/1/B queuing model, the probability of zero packets in the system (p_0) is given by the equation

$$p_0 = \begin{cases} \frac{1-\rho}{1-\rho^{B+1}} & \text{if } \rho \neq 1, \\ \frac{1}{B+1} & \text{if } \rho = 1. \end{cases} \tag{4}$$

And, the probability of 'B' packets in the system is stated as

$$p_B = p_0 \rho^B. \tag{5}$$

The average queuing delay can be calculated from the duration of time spent by the packet in the queue. The average number of packets in the queue E[N] is stated as

$$E[N] = \sum_{i=0}^{B} i p_i = \begin{cases} p_0 \sum_{i=0}^{B} i \rho^i & \text{if } \rho \neq 1, \\ \frac{B}{2} & \text{if } \rho = 1. \end{cases} \tag{6}$$

And, the effective arrival rate is defined as the rate of the packet actually entering the system and is given by

$$\lambda' = \lambda(1 - p_B). \tag{7}$$

The average waiting time is then given by the equation

$$W = \frac{E[N]}{\lambda'}. \tag{8}$$

In addition, the throughput in M/M/1/B queuing model is represented as

$$Throughput = (1 - p_0)\mu. \tag{9}$$

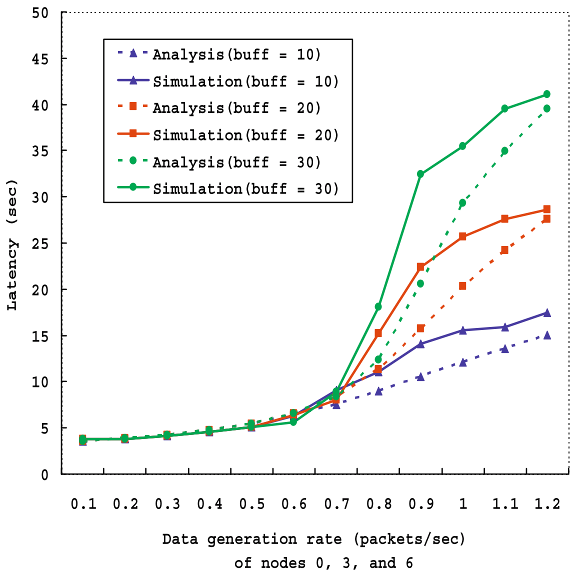
The traffic generated by a source encounters a queuing delay, a MAC delay, and a transmission delay during its transmission to the destination. We assume that MAC and transmission delay is small compared to queuing delay. Considering only queuing delay of each node, the end-to-end latency in a multi-hop forwarding path is represented as

$$T_{E2E} = \sum_{\forall i \in \text{ forwarding-path}} W_i, \tag{10}$$

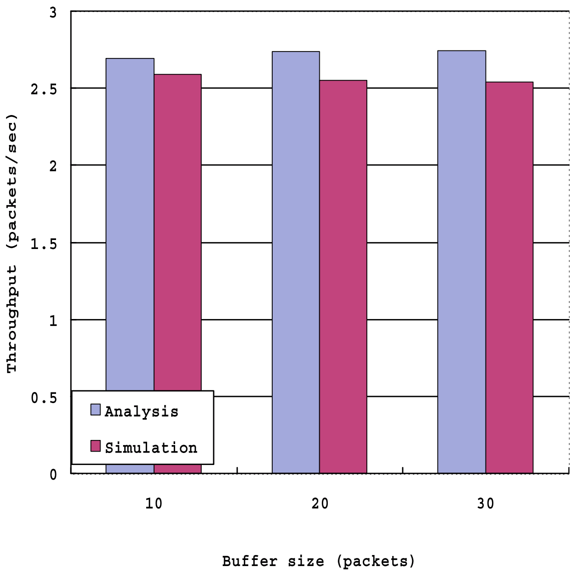
where W_i is the average waiting time on a node *i*. The throughput in a multi-hop forwarding path is determined by the node with the highest traffic density.

To validate the queuing model of TC-MAC, we analyzed and simulated the end-to-end latency and throughput in a chain topology (hop-count $(n) = 9$) shown in Fig. 18. We used the simulation parameters that will be introduced in Section 6.1 and simulated with various IFQ sizes. We generated traffic destined for node 9 at nodes 0, 3, and 6 at the same rate, and measured the average end-to-end latency of traffic generated from node 0. The throughput is measured by counting successfully received packets at node 9.

As shown in Fig. 19a, the end-to-end latencies of analysis and simulation below the data generation rate of 0.7 packets/s are almost similar. However, above the data generation rate of 0.7 packets/s, the end-to-end latencies of simulation is longer than those of analysis, because the MAC delay which is not considered in analysis increases. With high traffic load, there are many schedule collisions, channel reservation failures, and packet retransmissions, which increase the MAC delay and decrease the effective



(a) End-to-end latency.



(b) Throughput.

Fig. 19. Comparison of end-to-end latency and throughput in multi-hop forwarding.

link transmission rate. For the same reason, the throughput of simulation is slightly lower than the throughput of analysis, as shown in Fig. 19b.

5. Scalability

TC-MAC consists of various recovery mechanisms such as the schedule collision handling and transmission error recovery, and throughput enhancement mechanisms such as the channel reservation sharing and schedule connection. Those mechanisms are designed considering the multi-hop forwarding schedules of other nodes, but the nodes that need those mechanisms only use their local knowledge. Therefore, the overhead of those mechanisms does not increase considerably as the size of the network grows. The traffic monitor for congestion control also traces local channel usage information; though it consumes some extra energy to get the channel usage of other traffic flows. We reduce this overhead by only allowing the nodes with data to send to overhear the data packet headers (not whole data packets) of other traffic flows at the second N schedule of $\{N, N, N\}$, as shown in Fig. 12.

In a real implementation of TC-MAC, the various proposed mechanisms will consume some extra energy for computation. However, their computations are carried out locally at each node, and the computation cost is often several orders of magnitude less than the communication cost in WSNs [13]. Therefore, TC-MAC also achieves good scalability.

6. Performance evaluation

This section presents a performance evaluation of TC-MAC. To show the benefits of TC-MAC, we compared its latency, throughput, fairness, and energy consumption with the corresponding values of related protocols.

6.1. Simulation environment

We implemented and evaluated the whole operations of TC-MAC in the ns-2 (ver. 2.29) simulator [14]. For the comparison, we chose S-MAC, S-MAC-with adaptive listening (S-MAC-ADAPT), and an 802.11-like MAC without sleeping (802.11-LIKE). T-MAC is not included because S-MAC-ADAPT is similar to T-MAC in that the listen period can be adaptively extended by overhearing the transmissions of the neighbors.

The sizes and transmission times of packets used in the simulations are represented in Table 1. For TC-MAC, S-MAC and S-MAC-ADAPT, we used a 10% duty cycle (143 ms for the listen period and 1290 ms for the sleep period). For the multi-hop channel reservation of TC-MAC, the send-time of the data for the LAS-RTS transmission is set to $10T_{LAS\text{-}RTS}$ and that for data transmission is set to $10T_{data}$ ($T_{offset} = 7T_{data}$), which are used by the first node of multi-hop forwarding. The maximum number of schedule shift for TC-MAC is 3.

Table 1
The size and transmission latency of packets.

Packet	Size (bytes)	Transmission time (ms)
RTS/CTS/ACK	10	11.0
LAS-RTS/CN	14	14.2
DATA	50	43.0

We simulated a sensor node with the Two Ray Ground radio propagation model in the air, a single omni-directional antenna, and 20 kbps wireless interface with the 250 m transmission range and the 550 m carrier sensing range, which works in a similar manner to the 914 MHz Lucent WaveLAN direct sequence spread spectrum (DSSS) radio interface. To evaluate the energy consumption, we used an energy model with $P_{transmission} = 24$ mW, $P_{receive} = 13$ mW, and $P_{idle} = 13$ mW, where P represents *power* and these parameters are referenced from the model TR1000 of RF Monolithics Inc. [15].

For the routing protocol, we used the greedy perimeter stateless routing (GPSR) [16,26] with an additional *route_query* interface that is called when TC-MAC queries the next forward hop to the routing-layer. The simulation results were measured after the routing path was set up and cached to exclude the effect of the routing protocol on performance.

6.2. Performance of basic operations

To evaluate the performance of multi-hop forwarding, we used a multi-hop chain topology with a CBR (constant bit rate) flow from node 0, as shown in Fig. 20. The nodes are equally spaced such that only a sender's immediate neighbors can receive packets; that is, the range of the sender's radio transmission covers only the sender's immediate neighbors.

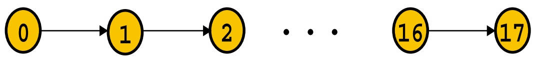


Fig. 20. A multi-hop chain topology.

6.2.1. Latency

Fig. 21 shows the average latency of multi-hop forwarding as the hop count increases. The latencies of S-MAC, S-MAC-ADAPT, and 802.11-LIKE increase linearly as the hop count increases, though the slopes of S-MAC and S-MAC-ADAPT are much higher than that of 802.11-LIKE. This phenomenon occurs because S-MAC and S-MAC-ADAPT use the periodic listen-and-sleep scheme and wait for the next listen period to forward data at each hop. The latency of S-MAC is proportional to the length of the listen-and-sleep cycle; the latency is increased by 1433 ms (143 ms + 1290 ms) at each hop. On the other hand, owing to the adaptive listening scheme, the latency of S-MAC-ADAPT is almost three times lower than that of S-MAC. In Section 2, we mentioned that S-MAC-ADAPT could forward only one more hop more per period than S-MAC, though results show that S-MAC-ADAPT can forward two hops more. In examining this phenomenon, we found that the transmission range of a sender was up to one-hop neighbors, though the carrier sensing range of a sender was up to two-hop neighbors [17]. In S-MAC and S-MAC-ADAPT, if the nodes are sending or receiving

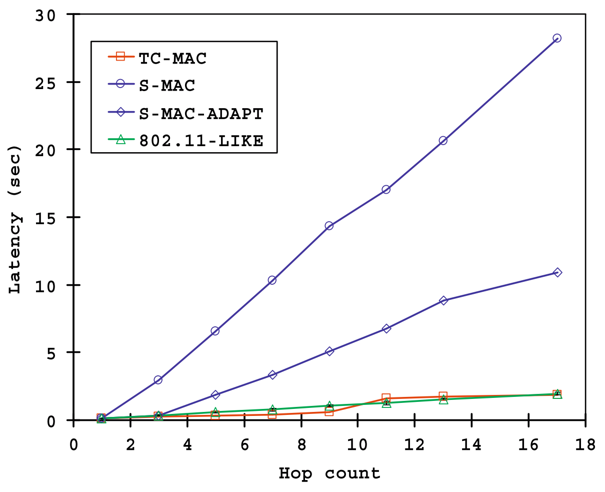


Fig. 21. Latency as the hop count of forwarding increases.

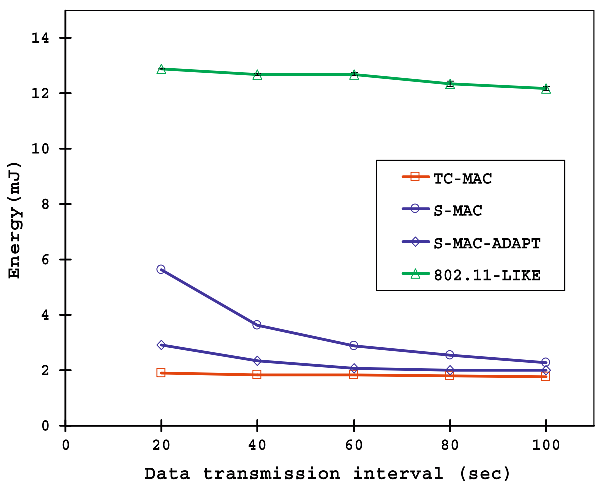


Fig. 22. Energy consumption per node (hop-count = 9).

something at the end of the listen period, they do not sleep for the entire sleep period because they are designed to run on an event-driven operating system called TinyOS [18]; thus, they determine whether the nodes go to sleep or not only at the events of listen timer expiration. Therefore, the node that is two hops away from a sender can sense the radio signal and does not sleep; it can then receive data from its immediate neighbor. T-MAC also extends its listen period by sensing its neighbor's transmissions; thus its performance is similar to that of S-MAC-ADAPT.

The latency of TC-MAC is enhanced considerably more than that of S-MAC and S-MAC-ADAPT and as low as that of 802.11-LIKE. S-MAC-ADAPT makes only the nodes in two hops away from a sender awake and forward data per period, whereas TC-MAC makes the nodes in more than two hops awake and forward data by means of multi-hop scheduling. Furthermore, when the hop count is less than 11, TC-MAC is even faster than 802.11-LIKE; This comes from the pipelined data forwarding without unnecessary control packets. However, the latency of TC-MAC is increased abruptly at the hop count of 11 because the amount of multi-hop reservations is limited by the duration of the listen period. In this result, the LAS-RTS can be forwarded and can reserve multi-hop channels of 10 hops during the listen period; the remainder of the forwarding hops are scheduled and transmitted in the next listen period.

6.2.2. Energy consumption

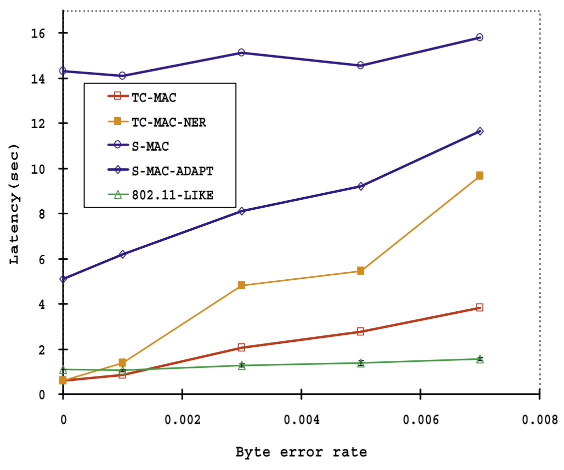
Fig. 22 shows the energy consumption of the MAC protocols as the traffic load changes. We represented the average energy consumption per node during a fixed time interval. The hop count of the multi-hop forwarding was nine, and the traffic load was changed by varying the data transmission interval. Regardless of the data transmission interval, 802.11-LIKE consumes the most energy because it is always awake and most of the energy is consumed by the idle listening. On the other hand, the other MAC protocols consume much less energy because they reduce the idle listening by means of the periodic listen-and-sleep scheme.

S-MAC-ADAPT achieves better energy efficiency than S-MAC. S-MAC determines whether the node goes to sleep or not only at the end of the listen period; thus, once S-MAC decides to be awake at the reception of neighbor's packets, the node remains awake for the entire sleep period and consumes considerable energy. However, S-MAC-ADAPT additionally tries to determine whether the node goes to sleep whenever packet transmissions are completed; thus S-MAC-ADAPT largely reduces the overall wake-up time. TC-MAC is the most energy efficient because it reduces the unnecessary control packet transmissions and is awake only for the scheduled time.

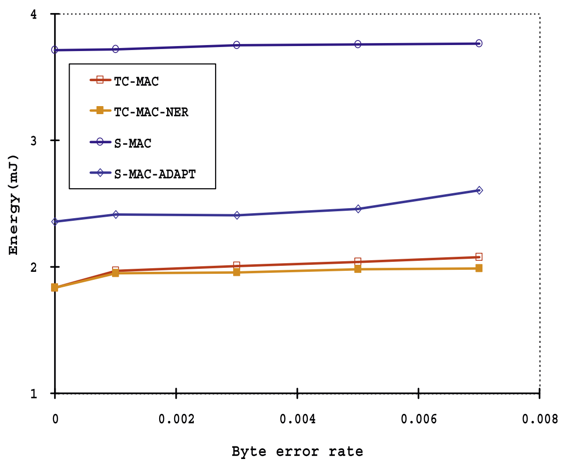
In Fig. 22, with the short data transmission intervals, there are more data packet transmissions of neighbors, so S-MAC and S-MAC-ADAPT are more likely to be awake unnecessarily and consume more energy although the data packets are not destined for the node. TC-MAC consumes slightly more energy due to the increased data packet transmissions. With the long data transmission intervals, most energy is consumed by the fixed listen periods, so the energy consumption is similar for TC-MAC, S-MAC, and S-MAC-ADAPT. TC-MAC consumes a little less energy than other protocols since it sends less control packets. The 10% duty cycle MACs, namely TC-MAC, S-MAC, and S-MAC-ADAPT, consume more than 10% of energy consumed by 802.11-LIKE because they periodically transmit the synchronization packets for the virtual clusters.

6.2.3. Performance of multi-hop forwarding in an error-prone environment

Fig. 23a plots the average latency of multi-hop forwarding as the byte error rate varies. TC-MAC with no transmission error recovery mechanism (TC-MAC-NER) is added to evaluate the effect of the transmission error recovery mechanism. The latencies of all protocols increase as the byte error rate increases but their slopes are different. The latency of 802.11-LIKE increases at the slowest rate because it is always awake and handles errors immediately. For S-MAC, the latency grows at a relatively slow speed. When a data transmission has error, it tries to recover the error with immediate retransmission without going to sleep.



(a) Latency with byte errors (Hop-count = 9).



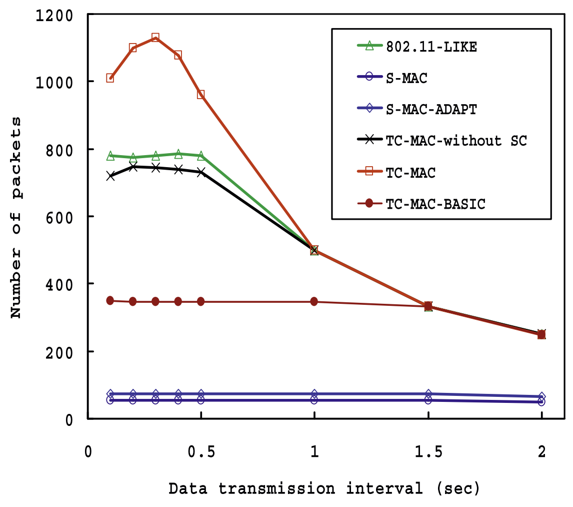
(b) Energy consumption per node with byte errors (Hop-count = 9, Data-transmission-interval = 40 Seconds).

Fig. 23. Performance of multi-hop forwarding in error-prone Environments. TC-MAC-NER represents TC-MAC with no transmission error recovery mechanism.

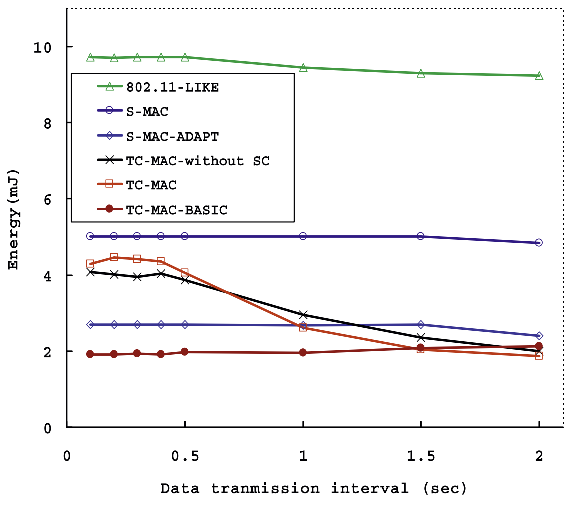
The latency of S-MAC-ADAPT increases rapidly compared to that of S-MAC because the adaptive listening scheme of S-MAC-ADAPT does not work well with data transmission errors. At the end time of previous node's transmission, the next forward node wakes up for a fixed time but cannot receive data from its previous node because the previous node's error; the end time of the previous node's transmission can be changed by retransmissions.

The latency of TC-MAC-NER also increases steeply because TC-MAC-NER sends data in a pipelined way; thus a failure in the middle node makes all the subsequent schedules useless. However, as shown in the graph, TC-MAC alleviates this problem considerably with the error recovery mechanism, namely the schedule shift. With a relatively small byte error rate (less than 0.002 byte error), TC-MAC is as fast as 802.11-LIKE.

Fig. 23b shows the average energy consumption per node in a fixed time interval with byte errors. Owing to the error recovery, the energy consumption is slightly increased as byte error increases; The energy consumption of 802.11-LIKE is not shown because of the scale of the graph; however, the energy consumption is around 12.7 mJ and it also increases slightly as the byte error increases. The energy consumption of TC-MAC is slightly higher than that of TC-MAC-NER, which means that only minimal energy is consumed for the transmission error recovery mechanism in TC-MAC.



(a) Throughput with high traffic loads.

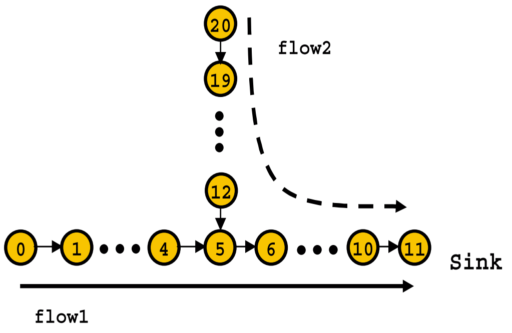


(b) Energy consumption per node.

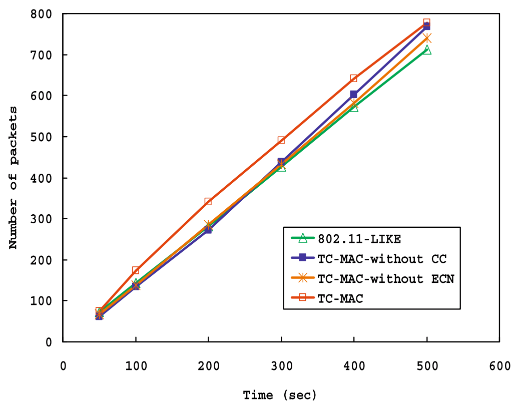
Fig. 24. Throughput and corresponding energy consumption per node. TC-MAC-without SC represents TC-MAC without the schedule connection mechanism; TC-MAC-BASIC represents TC-MAC only with the basic operations.

6.3. Performance of throughput enhancement mechanisms

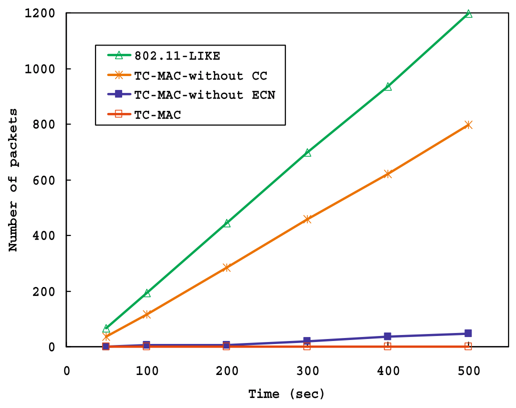
We examined throughput and corresponding energy consumption in the same chain topology as used in section 6.2 with 9 hop-count and a fixed simulation time. The results are shown in Fig. 24. TC-MAC, TC-MAC-without Schedule Connection (SC), and TC-MAC-BASIC are configured not to use the congestion control mechanisms. The effect of the congestion control is shown in Section 6.4. As data transmission interval decreases, the amount of



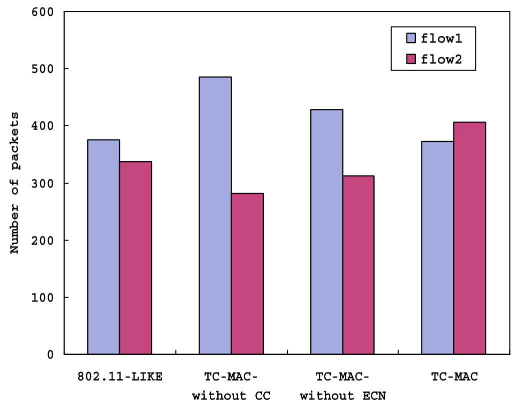
(a) A topology with two flows.



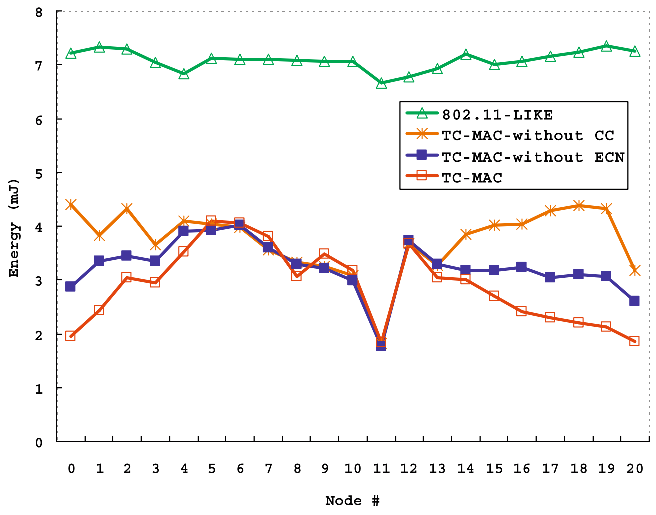
(b) Throughput.



(c) Number of dropped packets.

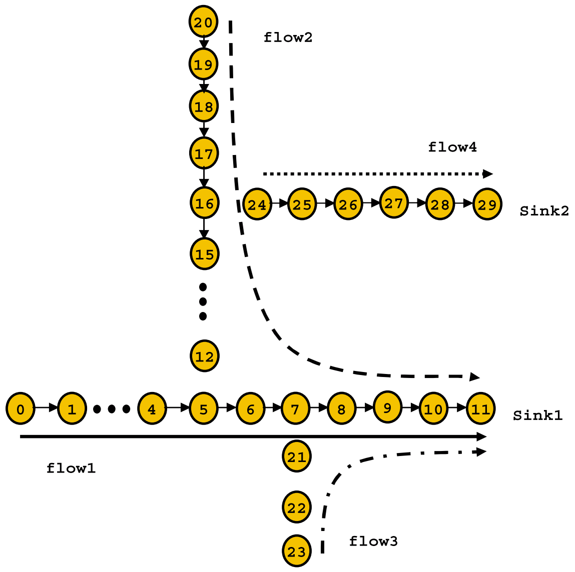


(d) Distribution of received data packets.

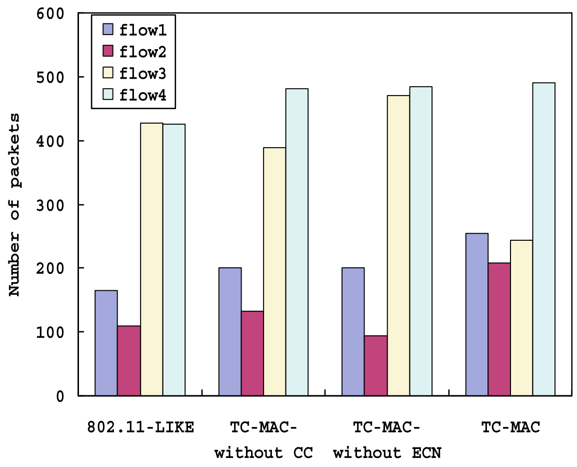


(e) Energy consumption of each node.

Fig. 25. Performance of a topology with two flows. TC-MAC-without CC represents TC-MAC without the congestion control mechanisms; TC-MAC-without ECN represents TC-MAC without the explicit congestion notification mechanism.



(a) A topology with four flows.



(b) Distribution of received data packets.

Fig. 26. Performance of a topology with four flows.

transmitted data increases. However, S-MAC and S-MAC-ADAPT have too low throughput to accommodate the increased amount of data. TC-MAC-BASIC shows moderate throughput which results from multi-hop channel reservation, while TC-MAC without SC shows almost the same performance as 802.11-LIKE due to the channel reservation in data transmission. TC-MAC achieves the best throughput using the throughput enhancement algorithms that lead to the efficient use of sleep period and multi-hop channel. The energy consumption of TC-MAC increases as the amount of forwarding data increases but it is not greater than that of 802.11-LIKE or S-MAC. Therefore, TC-MAC achieves good throughput as well as energy efficiency.

6.4. Performance of congestion control mechanisms

As multiple flows from different geographical regions gather, congestion frequently occur. In this section, we evaluate TC-MAC in multiple-flow congestion scenarios. We set the size of the interface queue to 500 bytes, and configured the system to send backpressure at a queue size of 250 bytes and ECN at the queue size of 350 bytes.

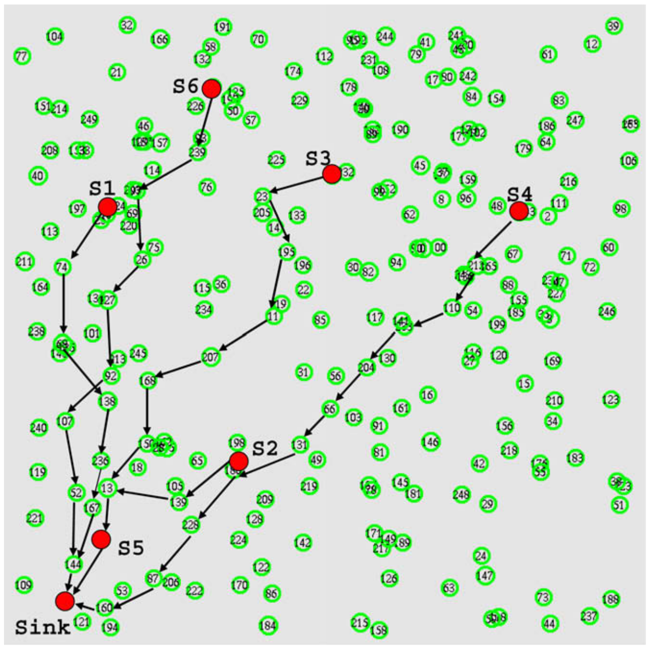


Fig. 27. A realistic topology.

6.4.1. *Topology with two traffic flows*

To evaluate the performance of the congestion control mechanisms, we set the topology shown in Fig. 25a. There are two traffic flows both bound for node 11, and they join together at node 5. The data transmission interval of each CBR flow source is set to 0.5 s.

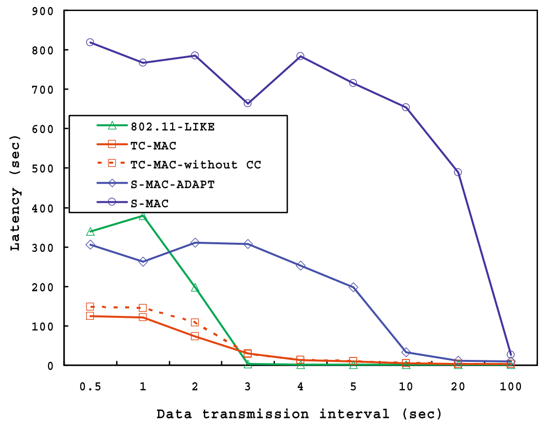
Fig. 25b shows the number of received packets at sink node 11 as simulation time varies. The throughput of the presented protocols is similar except for TC-MAC: TC-MAC's throughput is superior because the congestion control mechanisms regulate the transmission rates of traffic sources so that few packets are dropped. The number of dropped packets in the middle nodes of the forwarding paths is shown in Fig. 25c. Without the congestion control, 802.11-LIKE and TC-MAC-without CC lose more packets than they receive. However, the backpressure mechanism eliminates most of the congestion losses, as TC-MAC-without ECN shows, and the ECN mechanism further reduces packet losses, as TC-MAC shows.

Fig. 25d shows the fairness of the two flows. 802.11-LIKE and TC-MAC ensures the fairness between the two flows while the others do not. In this topology, two flows share the wireless medium around node 5 equally, so 802.11-LIKE can forward nearly the same amount of packets using only the contention mechanism which exchanges the RTS/CTS. However, TC-MAC-without CC and TC-MAC without ECN cause unfairness because once they get the shared medium, they send consecutive packets in the IFQ during the sleep period, rather than contending for the shared medium at every packet transmission like 802.11-LIKE does. Therefore, the ECN mechanism is necessary for the fairness control.

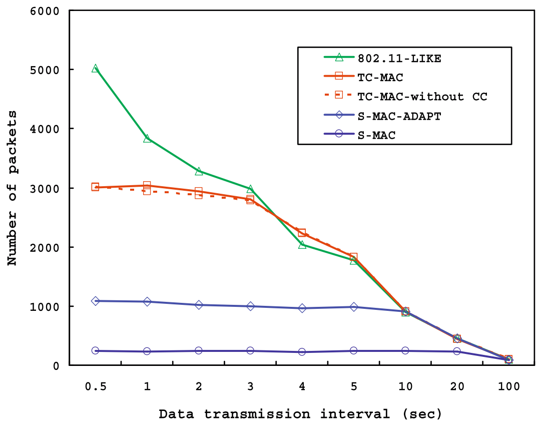
We have also plotted the energy consumption of each node in Fig. 25e. Although 802.11-LIKE and TC-MAC have similar throughput, 802.11-LIKE consumes far more energy than TC-MAC. For 802.11-LIKE, TC-MAC-without CC, and TC-MAC-without ECN, the energy consumption of the nodes between node 0 and crossing node 5 and between node 20 and node 5 increases because these nodes consumes unnecessary energy for dropped packets. However, the congestion control mechanisms of TC-MAC considerably reduces the number of dropped packets, and therefore the energy consumption of these nodes decreases.

6.4.2. *Topology with four traffic flows*

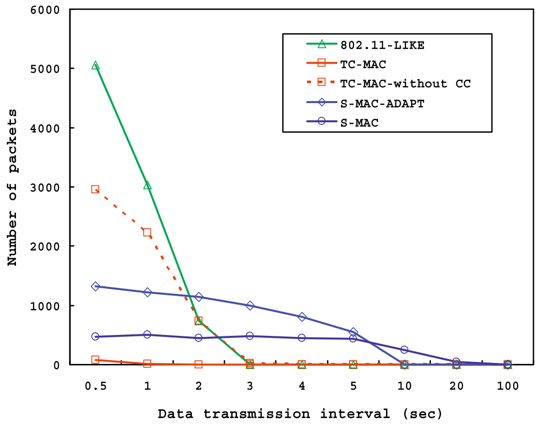
In the topology of Fig. 25a, the effect of the ECN is not shown sufficiently, so we set another topology, as shown in Fig. 26a. There are 4 traffic flows in this topology. *Flow1*



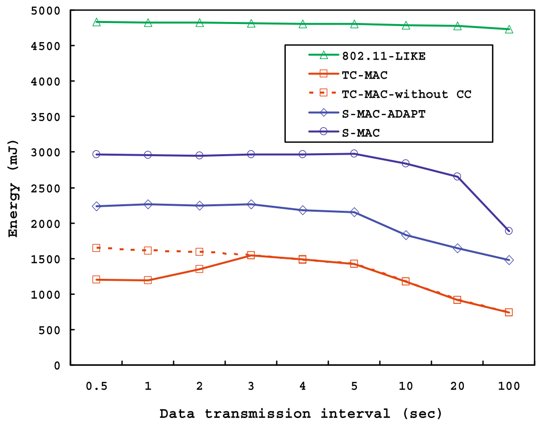
(a) Latency (Hop-count = 9).



(b) Throughput.



(c) Number of dropped packets.

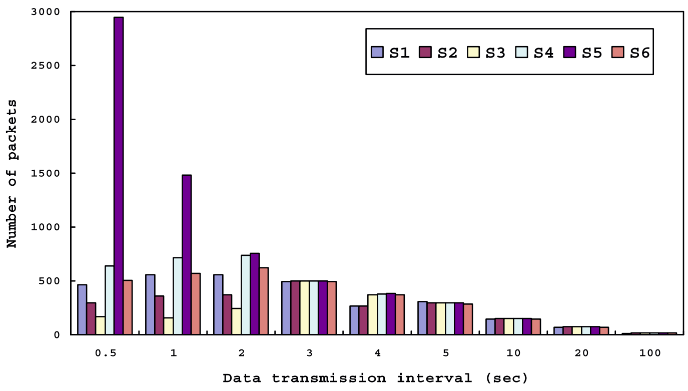


(d) Energy consumption of all the nodes.

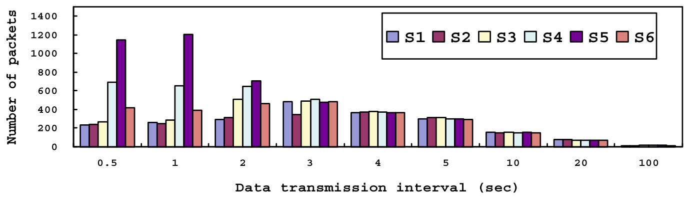
Fig. 28. Performance in a realistic topology. TC-MAC-without CC represents TC-MAC without the congestion control mechanisms.

and *flow2* gather at node 5, and then *flow3* joins to them at node 7; then, they are forwarded to node 11. *Flow4* is an independent flow and is forwarded from node 24 to node 29. We set the data transmission interval to 1 s and ran the simulation for 500 s.

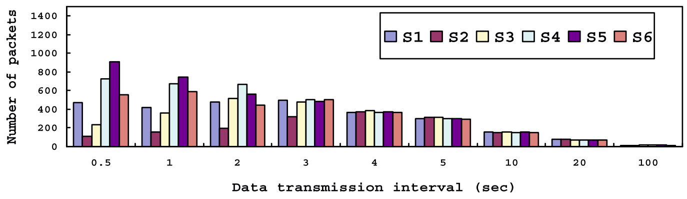
The number of received packets at node 11 and node 29 is shown in Fig. 26b. *Flow4* forwards almost all generated data because there is no congestion around node 16. However, the other flows experience congestion around node 5 and node 7, and congestion losses occur. In 802.11-LIKE,



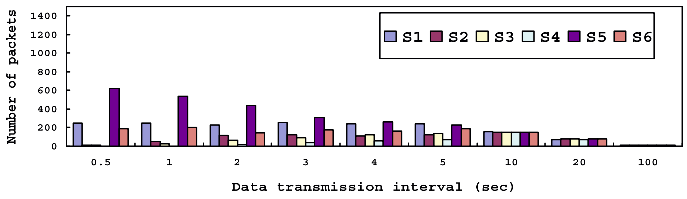
(a) 802.11-LIKE.



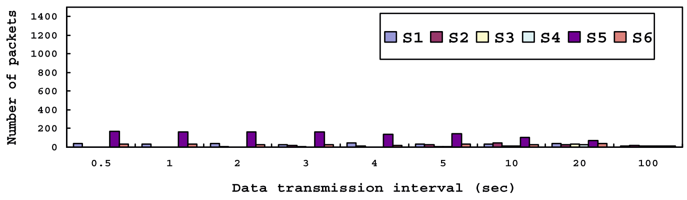
(b) TC-MAC.



(c) TC-MAC-without CC.



(d) S-MAC-ADAPT.



(e) S-MAC.

Fig. 29. Distribution of received data packets.

TC-MAC without CC, and TC-MAC without ECN, *flow3* forwards more data than either *flow1* or *flow2* because *flow3* contends with the conjoined *flow1* and *flow2*. However, TC-MAC achieves good fairness because the traffic monitor traces the wireless channel usage of each flow and helps to throttle the most channel-consuming node; thus, ECN throttles *flow3*.

6.5. Performance in a realistic environment

In this section, we evaluate the performance of TC-MAC in a realistic topology shown in Fig. 27, which is composed of 250 nodes distributed uniform randomly in a 2000 m by 2000 m square area. The sink is located at the bottom left corner of the square, and there are 6 randomly chosen traffic sources (from S1 to S6). We set the size of the interface queue to 1000 bytes, and configured the system to send backpressure at a queue size of 500 bytes and ECN at the queue size of 700 bytes. We used the greedy perimeter stateless routing (GPSR) [16,26], and the routing paths from the traffic sources to the sink are depicted in Fig. 27. We ran the simulation for 1500 s and the performance results are shown in Figs. 28 and 29.

For the latency evaluation, we plot the average latency of 9-hop away data to the sink, as shown in Fig. 28a. As the data transmission interval decreases, the latency of S-MAC increases considerably due to increased channel contention. S-MAC delays its data transmission until the next period whenever it loses at channel contention, causing a very long latency. S-MAC-ADAPT considerably reduces this long latency of S-MAC by the adaptive listening. The latency of 802.11-LIKE also increases with the increased channel contention. However, the latencies of TC-MAC and TC-MAC-without CC are much lower than that of 802.11-LIKE because once they reserve multi-hop channel schedules, the data in the forwarding path can be forwarded without further channel contention. TC-MAC has a lower latency than TC-MAC-without CC due to the reduced network congestion.

Fig. 28b shows the total number of received data packets at the sink during a 1500-s interval, and Figs. 28c and d, and 29 show the corresponding number of dropped packets in the middle nodes of the forwarding paths, energy consumption of all the nodes, and distribution of received data packets, respectively. Even though TC-MAC is a duty-cycle MAC, it shows good throughput with small numbers of congestion losses in the realistic topology. Moreover, TC-MAC consumes the lowest energy among the presented MAC protocols. S-MAC and S-MAC-ADAPT consume more energy than TC-MAC even though they have much lower throughput, since they receive many radio signals of other traffic at the end of the listen period and therefore cannot go to sleep. As we mentioned in Section 6.2.2, S-MAC determines whether the node goes to sleep only at the end of the listen period and S-MAC-ADAPT additionally determines whenever transmissions are completed. TC-MAC consumes less energy than TC-MAC-without CC at short data transmission intervals, because even though TC-MAC and TC-MAC-without CC have similar throughput, TC-MAC has much fewer number of dropped packets.

If there is no congestion with the large data transmission intervals, all the traffic sources can equally transmit data, as shown in Fig. 29. However, as the congestion start to occur, there is a serious imbalance among the packets received from each traffic source. For TC-MAC, the ECN mechanism tries to balance the sending rates of traffic flows sharing the same channel. Therefore, the traffic sources except *S4* and *S5* transmit similar amount of data (over 230 packets). *S4* can transmit more than others (except *S5*) because it transmits data through the remote forwarding path whose channel is shared by fewer traffic flows. Although *S5* is in the hot channel sharing region, *S5* can transmit more packets, because *S5* is on the paths of *S3* and *S2*, and *S5* can therefore transmit its data after the transmissions of *S2* or *S3* if the channel is available, as explained in Section 3.3.2.

7. Conclusion

Many listen-and-sleep MAC protocols have been suggested for WSNs, though they have sacrificed end-to-end latency and throughput to achieve that efficiency. Furthermore, as the scale of WSNs grows, the length of forwarding paths increases, and the traffic load increases where traffic from various areas is gathered. Such increased traffic causes network congestion and wastes considerable energy.

Motivated by these problems, we have presented a transport-controlled MAC (TC-MAC) protocol designed to provide low end-to-end latency, high throughput, and light-weight congestion control without sacrificing energy efficiency. TC-MAC achieves energy efficiency from a periodic listen-and-sleep scheme as other MAC protocols for WSNs do. However, TC-MAC uses the listen period for multi-hop channel reservation and the sleep period for data transmission to encourage low end-to-end latency, while piggybacking the following data's forwarding schedule on data packets for high throughput. For light-weight and fairness-aware congestion control, TC-MAC deploys backpressure and explicit congestion notification schemes with a traffic monitor. Simulation results confirm that TC-MAC achieves its goals. TC-MAC consumes as little energy as S-MAC and performs as well as the 802.11-like MAC without sleeping.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Commun. Mag. 40 (8) (2002) 102–114.

[2] M. Stemm, R.H. Katz, Measuring and reducing energy consumption of network interfaces in hand-held devices, IEICE Trans. Commun. E80-B (8) (1997) 1125–1131.

[3] W. Ye, J. Heidemann, D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in: Infocom '02, IEEE, 2002, pp. 1567–1576.

[4] T. van Dam, K. Langendoen, An adaptive energy-efficient MAC protocol for wireless sensor networks, in: SenSys'03: Proceedings of the First International Conference on Embedded Networked Sensor Systems, ACM Press, New York, NY, USA, 2003, pp. 171–180.

[5] D.P. Jaesub Kim, Keuntae Park, An energy-efficient scheduling MAC protocol for wireless sensor networks, in: Consumer Communications and Networking Conference (CCNC)'07, IEEE, 2007, pp. 655–659.

[6] C.-Y. Wan, S.B. Eisenman, A.T. Campbell, CODA: congestion detection and avoidance in sensor networks, in: SenSys'03: Proceedings of the

First International Conference on Embedded Networked Sensor Systems, ACM, New York, NY, USA, 2003, pp. 266–279.
[7] IEEE, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Standard 802.11, 1999.
[8] W. Ye, J. Heidemann, D. Estrin, Medium access control with coordinated adaptive sleeping for wireless sensor networks, IEEE/ACM Trans. Netw. 12 (3) (2004) 493–506.
[9] S. Du, A.K. Saha, D.B. Johnson, RMAC: a routing-enhanced duty-cycle MAC protocol for wireless sensor networks, in: INFOCOM, 2007, pp. 1478–1486.
[10] IEEE 2003 Version of 802.15.4 MAC and PHY Standard, <http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf>.
[11] B. Hull, K. Jamieson, H. Balakrishnan, Mitigating congestion in wireless sensor networks, in: SenSys'04: Proceedings of the Second International Conference on Embedded Networked Sensor Systems, ACM, New York, NY, USA, 2004, pp. 134–147.
[12] K. Xu, M. Gerla, L. Qi, Y. Shu, TCP unfairness in ad hoc wireless networks and a neighborhood red solution, Wirel. Netw. 11 (4) (2005) 383–399.
[13] G.J. Pottie, W.J. Kaiser, Wireless integrated network sensors, Commun. ACM 43 (5) (2000) 51–58.
[14] Ns-2, The Network Simulator, <http://www.isi.edu/nsnam/ns>.
[15] Ash Transceiver tr1000 Data Sheet, <http://www.rfm.com>.
[16] B. Karp, H.T. Kung, GPSR: greedy perimeter stateless routing for wireless networks, in: MobiCom'00: Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking, ACM Press, New York, NY, USA, 2000, pp. 243–254.
[17] H.L.S.L.L.Z.M.G. Zhenghua Fu, Petros Zerfos, The impact of multihop wireless channel on TCP throughput and loss, in: Infocom '03, IEEE, 2003, pp. 1744–1753.
[18] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister, System architecture directions for networked sensors, in: ASPLOS-IX: Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems, ACM Press, New York, NY, USA, 2000, pp. 93–104.
[19] V. Bharghavan, A. Demers, S. Shenker, L. Zhang, MACAW: a media access protocol for wireless LAN's, in: SIGCOMM'94: Proceedings of the Conference on Communications Architectures, Protocols and Applications, ACM Press, New York, NY, USA, 1994, pp. 212–225.
[20] O. Kasten, Energy consumption. <http://www.inf.ethz.ch/kasten/research/bathtub/energy_consumption.html>.
[21] C. Wang, K. Sohraby, B. Li, M. Daneshmand, Y. Hu, A survey of transport protocols for wireless sensor networks, IEEE Netw. 20 (3) (2006) 34–40.
[22] A. Woo, D.E. Culler, A transmission control scheme for media access in sensor networks, in: MobiCom'01: Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking, ACM, New York, NY, USA, 2001, pp. 221–235.
[23] K. Tang, M. Gerla, Fair Sharing of MAC under TCP in Wireless Ad Hoc Networks, 1999, <http://citeseer.ist.psu.edu/tang99fair.html>.
[24] K. Xu, S. Bae, S. Lee, M. Gerla, TCP behavior across multihop wireless networks and the wired internet, in: WOWMOM'02: Proceedings of the Fifth ACM International Workshop on Wireless Mobile Multimedia, ACM, New York, NY, USA, 2002, pp. 41–48.
[25] S. Xu, T. Saadawi, Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks?, Commun Mag., IEEE 39 (6) (2001) 130–137. <http://dx.doi.org/10.1109%2F35.925681>.
[26] A. Rao, C. Papadimitriou, S. Shenker, I. Stoica, Geographic routing without location information, in: MobiCom'03: Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking, ACM Press, New York, NY, USA, 2003, pp. 96–108.

Jaesub Kim received the BS degree in electrical engineering from Kyungpook National University in 2000 and the MS degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST) in 2002. He is currently a PhD student at KAIST. His research interests include sensor network, internet server systems, and operating systems.

Kyu Ho Park received the BS degree in electronics engineering from Seoul National University, Korea in 1973, the MS degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 1975, and the DrIng degree in electrical engineering from the University de Paris XI, France in 1983. He has been a Professor of the Division of EECS, KAIST since 1983. He was a president of Korea Institute of Next Generation Computing in 2005–2006. His research interests include computer architectures, file systems, storage systems, ubiquitous computing, and parallel processing. Dr. Park is a member of KISS, KITE, Korea Institute of Next Generation Computing, IEEE and ACM.