ECODA: Enhanced Congestion Detection and Avoidance for Multiple Class of Traffic in Sensor Networks

Li Qiang Tao and Feng Qi Yu

Abstract —*Congestion in a Wireless Sensor Network (WSN) can cause missing packets, low energy efficiency, and long delay. Moreover, some applications, e.g. multimedia and image, need to transmit large volumes of data concurrently from several sensors. These applications have different delay and QoS requirements. Congestion problem is more urgent in such applications. To address this challenge, we propose a novel energy efficient congestion control scheme for sensor networks, called ECODA (Enhanced congestion detection and avoidance) which comprises three mechanisms: 1) Use dual buffer thresholds and weighted buffer difference for congestion detection; 2) Flexible Queue Scheduler for packets scheduling; 3) A bottleneck-node-based source sending rate control scheme. Simulations show that ECODA achieves efficient congestion control and flexible weighted fairness for different class of traffic. Therefore it leads to higher energy efficiency and better QoS in terms of throughput, fairness, and delay.*

Index Terms — congestion; service differentiation; Qos; sensor network.

I. INTRODUCTION

Wireless sensor networks (WSNs) have been widely applied to habitat monitoring [4], real-time target tracking [5], environment surveillance [6] and healthcare [11], etc. They are different from traditional wireless networks in several aspects [9]. Commonly, sensor nodes are restricted in computation, storage, communication bandwidth, and, most importantly, energy supply. Extensive studies have been carried out in recent years on the physical layer [10], the media access control (MAC) layer [21-22] [24-26], the network layer[2] [12-13] [27] and transport layer[1][3] [7-8] [14-21] in WSNs.

The event-driven nature of WSNs leads to unpredictable network load. Typically, WSNs operate under idle or light load and then suddenly become active in response to a detected event. When the events have been detected, the information in transit is of great importance. However, the busty traffic that results from the detected events can easily cause congestion in the networks. When congestion happens, the network throughput and coverage fidelity are penalized. So, congestion control is a critical issue in sensor networks.

In WSNs, congestion can be divided into transient congestion and persistent congestion. Transient congestion is caused by link variations, and persistent congestion is caused by source data sending rate. Congestion control mechanism can be classified into end-to-end congestion control and hop-by-hop congestion control. End-to-end congestion control performs exact rate adjustment at source and intermediate nodes according to current QoS level at sink node. The drawback of end-to-end congestion control mechanism is that it heavily relies on round-trip time (RTT), which results in slow response and low convergence. In contrast, hop-by-hop congestion control has faster response.

Currently, there are extensive studies to address congestion problems in WSNs [1] [3] [7-8] [14-21]. Some papers [7-8] [20] provide reliable end-to-end data delivery from every sensor to a sink and hop by hop congestion control [8] [15-19] at every intermediate node on the path from source to sink. However, how to ensure weighted fairness for multiple class of traffic among sensors is not well address by previous research. Congestion Detection and Avoidance (CODA) [8] by jointly sampling the channel loading during every epoch and monitoring buffer length of being filled to judge if congestion happens or not. For transient congestion, the node sends explicit backpressure messages to its neighbors which further propagate the message to upstream source nodes depending their local buffer occupancy or channel loading, every node receive the backpressure messages will lower down their sending rate except the designed node which has the priority to access channel. However, the backpressure message may intensify congestion due to high channel loading when congestion happens. For persist congestion, CODA needs explicit ACK from sink, if insufficient ACK reaches the source, the source will lower down its sending rate. However the explicit ACK waste much energy and the loss the ACK due to link quality will give a false congestion signal to the source and affect the network throughput. CODA can't differentiate bottleneck link either.

As sensor network software and hardware mature, applications which transfer image, video, and structure monitoring data in WSNs are becoming increasingly possible. These applications have different QoS requirements and

[1]This work was partially funded by the Science and Technology Project of Guangdong Province, China, project number 2009A080207008.

Liqiang, Tao is with Institute of Computing Technology, Chinese Academy of Science, Beijing 100080 (e-mail: lq.tao@siat.ac.cn).

Liqiang, Tao is with Graduate University of Chinese Academy of Sciences, Beijing 100049.

FengQi,Yu is with Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences/The Chinese University of Hong Kong, Shenzhen, Guangdong province, 518067,China(e-mail: fq.yu@siat.ac.cn).

Contributed Paper
Original manuscript received 05/28/10
Revised manuscript received 06/25/10
Current version published 09/23/10
Electronic version published 09/30/10.

should be serviced accordingly. We need to consider the fairness issue for packets with different importance. In this paper, we propose an efficient congestion detection and avoidance protocol for multiple class of traffic (ECODA), which has a distributed mechanism operating at the network and MAC layer. In ECODA, we incorporate the following key ideas:

1) ECODA adopts a technique to measure congestion, which uses dual buffer thresholds and weighted buffer difference for congestion detection. The method is different from traditional single buffer threshold method [7-8] [15-18], it could differentiate congestion level and dealt with them correspondingly.

2) The flexible queue scheduler can dynamically select the next packet to send. Moreover, it adopts a novel method to filter packets according to channel loading and packets priority when congestion happens.

3) Transient congestion and persistent congestion are differentiated and are dealt with differently. For transient congestion, hop-by-hop implicit backpressure manner is used. For persistent congestion, bottleneck node based source sending rate control and multi-path loading balancing are proposed. Unlike [8], this method does not need explicit ACK from sink. Using the method, bottleneck nodes can be identified and source sending rate can be dynamically adjusted more accurately.

The rest of this paper is organized as follows: Section II presents the most relevant work. Section III-IV elaborately describes ECODA. Section V presents the simulation results. Finally, the paper is concluded in Section VI.

II. RELATED WORK

In the literature, many works have been conducted on congestion control, congestion mitigation, and reliable transmission in WSNs. Existing works can generally be classified into three categories.

The first category consists of transport protocols that provide end-to-end reliability without congestion control. Reliable multisegment transport (RMST) [14] is an example of these protocols. RMST is a hop-by-hop reliable transport protocol specially designed to run on top of directed diffusion in which packet loss is recovered hop by hop using caches in the intermediate nodes. RMST guarantees reliability but is designed for more capable sensor nodes. In addition, in RMST, the node's transmit rate is manually set by a system administrator.

The second category consists of protocols with centralized congestion control schemes. ESRT [7] classifies a network into five regions. It adjusts source packet data sending rate such that network stays in a state where sufficient number of packets arrive at a sink without producing congestion. ESRT's rate allocation is centrally computed i.e., the base station periodically counts the number of received sensor readings and retasks the sensors by broadcasting a new transmission rate. Due to the drawbacks of centralized scheme, ESRT cannot deal with transient congestion efficiently. RCRT [20] is another centralized transport protocol in which all functionalities including congestion detection, rate adaptation, and rate allocation are implemented at sink node. Although RCRT's performance is good, it can't differentiate flows unconstrained in bottleneck regions. Also, RCRT's convergence is too slow when the network has highly varying RTTs.

The third category consists of protocols with distributed congestion control schemes. Fusion [16] uses hop-by-hop flow control, rate limiting, and prioritized MAC to alleviate congestion. With this combination, Fusion achieves higher goodput and better fairness with heavy loads than previous schemes. Congestion detection and avoidance (CODA) in sensor networks [8] is another congestion mitigation strategy, it provides a comprehensive discussion on congestion control and proposes an open-loop hop-by-hop backpressure mechanism and closed-loop multi-source regulation scheme. For transient congestion, each sensor monitors channel utilization and buffer occupancy level to detect congestion. For persistent congestion, source requires sink's feedback to maintain its data rate. Unlike Fusion, CODA doesn't explicitly focus on per-source fairness. IFRC [19] and CCF [15] are both congestion control protocols to ensure fairness. In IFRC, every node adopts multi-level buffer thresholds. When a node's buffer exceeds the threshold, it asks its neighbors to decrease data sending rate and maintain its buffer utilization less than a predefined level. In CCF, two schemes are proposed to ensure fairness: probabilistic selection and epoch based proportional selection. However, IFRC and CCF both try to ensure absolute fairness among every node.

III. PROTOCOL OVERVIEW

A. Use dual buffer thresholds and weighted buffer difference for congestion detection

Current approaches usually adopts single buffer threshold to measure congestion [7-8][15-18], In [19] multiple thresholds is adopts to measure congestion, In their approach, different buffer thresholds are used to maintain neighborhood nodes' data sending rate be identical to each other and ensure fairness, this is different from our design.

In our approach, we define buffer state as "accept state", "filter state" and "reject state". We use two thresholds to border them. Different strategies to accept or reject packets are adopted in different states. Using this method, congestion level could be identified efficiently and dealt with them separately.

In some applications, data priority is important in congestion control. However it has not been paid enough attention yet. Our solution to the problem is adopting weighted buffer difference in the neighborhood as the congestion resolving sequence basis.

B. Flexible Queue Scheduler and weighted fairness.

The tree structure in a conventional sensor network is unfairness in bandwidth allocation for nodes at different depth of the network. Nodes near a sink have more priority to send their own generated packets to the sink. While nodes far from the sink have to send their packets through many hops and experience a long delay. To resolve unfairness problem, we propose a Flexible Queue Scheduler, which differentiates locally generated traffic and route-through traffic. Every packet has two kinds of priorities: static priority and dynamic priority, which will be defined shortly. Packet static priority is represented as an integer and the lowest static priority *SP* (packet) =0. Dynamic priority changes with the number of hops and delay. When congestion happens, two parameters (δ_1, δ_2) are used to limit buffer increasing rate and filter packets. In section IV, we deduce δ_1 and δ_2 through experiment.

C. Bottleneck-node-based source sending rate control and multi-path loading balancing.

Although the explicit or implicit backpressure messages of distributed hop-by-hop congestion control can finally reach the
source node and source data sending rate is adjusted to mitigate congestion which happens in the downstream (sink side). It cannot accurately adjust source data sending rate. We propose a method, every node can determine routing path (from itself to sink) status. The forwarder can find better path to forward data and the source data sending rate can be adjusted more accurately and efficiently.

Below are some definitions and rules used in the paper.

Definition 1: Packet dynamic priority is defined as:

$$DP(packet) = \frac{\alpha * hop + SP(packet)}{1 + \beta * delay} \quad (1)$$

Where α and β are two parameters for tuning system performance; *hop* is the number of hops to sink; *SP* is the packet static priority; *delay* is the time from the packet generation to current.

Definition 2: If buffer incoming rate is *BI* and buffer outgoing rate is *BO*, then buffer changing rate *R=BO-BI* and weighted buffer changing rate is

$$WR = DP(BO) - DP(BI) \quad (2)$$

Where both *BI* and *BO* are determined by EWMA techniques. *WR* is the weighted buffer changing rate.

Definition 3: Weighted queue length is defined as $WQ = \sum_{j=1}^{N} DP(\text{packet}_j)$, where *N* is the total number of packets in the buffer.

Rule1: For a series of packets $\{p_1, p_2, p_3 \ldots\}$, their *SP* and *DP* are respectively denoted as $SP(p_1), SP(p_2), SP(p_3), \ldots$ $DP(p_1), DP(p_2), DP(p_3), \ldots$ If $SP(p_1) < SP(p_2) < SP(p_3) \ldots$ then the following relationship holds: $DP(p_1) < DP(p_2) < DP(p_3) \ldots$

Combining Def. 1 and Rule1, we can conclude that packets with same *SP* can have different *DP*, and packets with higher *SP* have higher *DP*.

IV. ECODA PROTOCOL DESIGN

A. Congestion Detection

In order to precisely measure local congestion level at each node, we propose dual buffer thresholds and weighted buffer difference for congestion detection. Buffer is defined as three states, "accept state", "filter state" and "reject state", as Fig. 1 indicate. Two thresholds Q_{min} and Q_{max} are used to border different buffer states. Different buffer states reflect different channel loading, corresponding strategy is adopted to accept or reject packets in different states. It is necessary to point out that, in this paper, "reject state" not means to reject all incoming packets, but it means that most of packets will be rejected because buffer utilization is too high.

Every node which has data to send monitors its buffer and piggybacks its *WR* and *WQ* in its outgoing packets. If a node's buffer occupancy exceeds a certain threshold and its data has higher priority among neighborhood, the corresponding congestion level bit in the outgoing packet header is set.

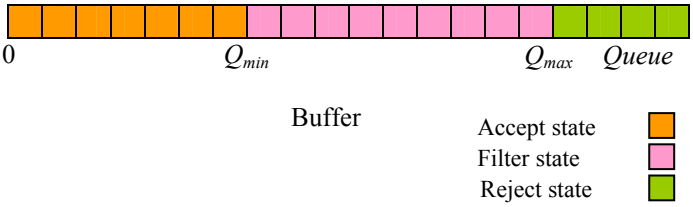


Fig. 1 Buffer state

For a weighted buffer with length *WQ (t)*, after Δt, it becomes

$$WQ(t+\Delta t) = WQ(t) + WR * \Delta t \quad (3)$$

The weighted buffer difference at time $t+\Delta t$ is

$$WQD_{nodei}(t+\Delta t) = \sum_{j=1}^{N} DP(packet_j) - Max(WQ_k(t+\Delta t)) \quad (4)$$

Where $k \in$ neighbor ($node_i$); *N* is the total number of packets in the buffer.

If $WQD_{nodei}(t+\Delta t) \geq 0$, it means that the data of $node_i$ is the most important among its neighbors. If congestion happens, other nodes should lower down their data sending rate to mitigate $node_i$'s congestion.

B. *Flexible Queue Scheduler and Weighted Fairness*

When congestion occurs, packets are dropped to alleviate congestion. Most of the queue schedulers, both in wired and wireless networks, drop packets from the tail rather than any position in the queue. But tail-dropping does not work well. For instance, if the queue in a sensor node is nearly full and dominated with low priority packets, when a high priority packet arrives, it is better to drop a low priority packet rather than the high priority packet. With tail-dropping, the high priority packet may be dropped due to queue overflow.

As indicated in Fig. 2, there are two sub-queues. One is for local generated traffic, and the other is for route-through traffic. In the route-through traffic queue, packets are grouped by sources. For every source, packets are sorted by their dynamic priority from high to low. When sending next packet, a round robin algorithm is adopted. To ensure fairness, the algorithm scans the route-through traffic queue from head to tail. One packet from one source is sent from route-through traffic queue, and then a local generated packet is sent.

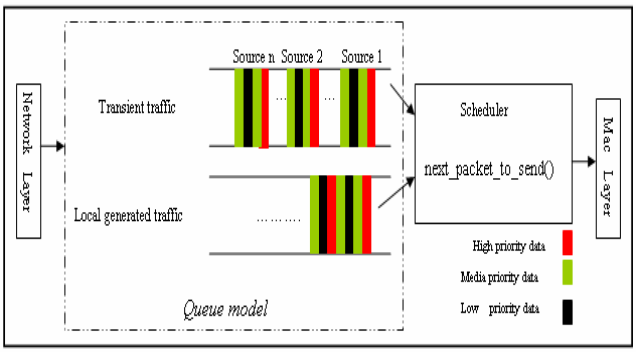


Fig. 2 Queue model

Suppose that the total buffer size of a node is Q. In the queue, the dynamic priorities of packets are respectively denoted as DP_1, $DP2$... DP_n, where n is the total number of priorities. The total number of packets are $N=N_{DP1}+N_{DP2}+N_{DP3}\ldots\ldots+N_{DPn}$. Two thresholds Q_{min} and Q_{max} are used, as shown in Fig. 1, for handling packets with different strategy:

1) If $0 \le N \le Q_{min}$, all incoming packets are buffered, because queue utilization is low.

2) If $Q_{min} \le N \le Q_{max}$, some packets with low dynamic priority are dropped or overwritten by subsequent packets with high dynamic priority, then the expected average buffer length increases at a rate of $R \le \delta_1$.

3) If $Q_{max} \le N \le Q$, some packets with high dynamic priority is dropped or overwritten, then the expected average buffer length increases at a rate of $R \le \delta_2$.

δ_1 and δ_2 are two variables that can be tuned to achieve optimal system performance. When Q_{min} is reached, the traffic around the node is slightly congested. Some packets with low dynamic priority are dropped to limit $R \le \delta_1$. When Q_{max} is reached, some packets with high dynamic priority are dropped to limit $R \le \delta_2$.

To validate (δ_1, δ_2) and its impact on system performance, we setup a simple experiment, where there are two source nodes (S1 and S2), a sink (S), and a node to forward packets (F) ,as Fig.3 indicated. Total buffer length is set 20, Q_{min}= (1.0/3.0)*20 and Q_{max}= (2.0/3.0)*20. Fig. 4 shows the simulation of buffer changing rate and packet dropping rate of the node F for different (δ_1, δ_2) scenarios.

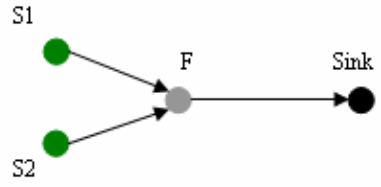
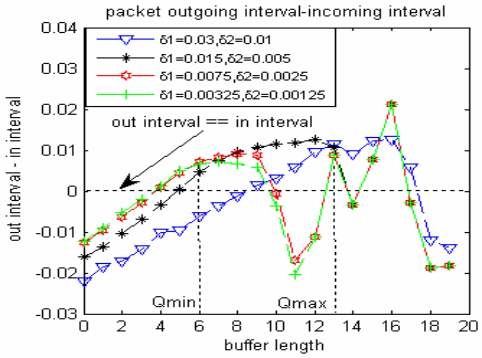
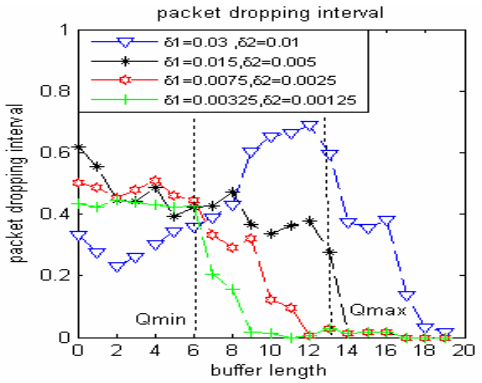


Fig. 3 Topology for testing (δ_1, δ_2)

One can see the experiment results in Fig.4, when buffer length is within [Q_{min}, Q_{max}], the bigger the δ_1 is, the faster the buffer length increases and the lower packet dropping rate. When δ_1=0.03, buffer length increases at the fastest rate and packet dropping rate is the smallest (packet drop interval is the biggest). When δ_1 lowers down to 0.015, buffer increases slowly; when δ_1 lowers down to 0.0075 and 0.00325, buffer length decreases and packet dropping rate becomes fast.



(a) Buffer changing rate.



(b) Packet dropping rate.

Fig. 4 The relation of (δ_1, δ_2) and buffer changing rate.

δ_2 has the same trend as δ_1. When buffer length is within [Q_{max}, Q], buffer utilization is high. So δ_2 should be less than δ_1 to reject more packets to lower down buffer increasing rate.

Based on the experiments and discussion above, tradeoff can be made to optimize system performance. When incoming packets have high dynamic priority, δ_1 and δ_2 are set relatively large to accept important packets. Otherwise δ_1 and δ_2 are set relatively small to reject less important packets. If we let buffer increasing rate be R, we have

$$\delta_1 = R*DP/MP \quad (5)$$

$$\delta_2 = 0.5*\delta_1 \quad (6)$$

DP is current packet dynamic priority; MP is the maximum priority. Using this scheme, packets' increasing rate in the buffer will be proportional to its DP (Dynamic priority), this optimize the packets acceptation in the buffer, when congestion happens.

C. Bottleneck node based source data sending rate control

Each source node S_i (i=1, 2, 3.....) maintains its data sending rate G_i.It periodically updates its data sending rate according to its neighbor's congestion level. Owing to many to one characteristic, when source nearby nodes send data smoothly, congestion could still happen in the downstream (sink side). To solve the problem, we propose a method which is called bottleneck node based source data sending rate control. It includes: 1) Determine routing path status from a certain node to sink; 2) Bottleneck node detection and source data sending rate control. Using this scheme, source data sending rate can be regulated more accurately.

C1. Determination of routing path status from a certain node to sink

For node i whose next hop is sink, its data forwarding delay is piggybacked in the data packets' header. Its child node overhears this information and compares its own forwarding delay D (i) with its parent p's data forwarding delay D (p) and does the following calculation:

$$D_{max}(i) = MAX\{D(p), D(i)\} \quad (7)$$

Next time, when this child has data to send, D_{max} (i) will be piggybacked in the packet header, D_{max} (i) is the path status from node i to sink. This process is recursively computed up to the final source node.

C2. Bottleneck node detection and data sending rate control

When source node s overhears data from its parent p, it extracts the delay information piggybacked in the data packets and set its data sending rate Gs as:

$$Gs = 1/D_{max}(p) \quad (8)$$

Like existing protocols[8][15-20], upon receiving a backpressure message, the source node or forward node decreases its data sending rate or adjusts data sending rate for different paths if multiple paths exist. However, unlike existing protocols, if no backpressure message is received, the source node doesn't increase its data sending rate additively. In ECODA, the source data sending rate can be exactly set as $1/D_{max}$ upon receiving a message from downstream. The pseudo code of the rate control algorithm is shown in Fig. 5

```
Event {
   a0: receive backpressure message;
   a1: receive downstream maximum one hop delay
}

Action Set {
b0: additively increase data forwarding rate;
b1: multiplicatively decrease data forwarding rate;
b2: multiplicatively decrease data sending rate;
b3: set data sending rate to be 1/Dmax
b4: adjust data sending rate for different paths
}

Node type {
c0: a node which forward data generated by other nodes
c1: a node which generates data, but does not forward data
c2: a node has c0 and c1 simultaneously
}
```

Operations in forwarder node:

```
01    if ( node type=='c0' && node type! ='c1')   {
02       if (event! ='a0')
03       action='b0';
04  else if (event=='a0') {
05       action='b1';
06  if (exists multiple paths)
    action='b4';  }
}
```

Operations in source node:

```
01  if (node type=='c1' && node type! ='c0') {
02   if (event! =='a0' && event=='a1') {
03       action='b3';  }
04  else if (event=='a0')
05       else action='b2';
06  if (exists multiple paths)
07       action='b4' ;}
```

Fig. 5 The pseudo code of rate control algorithm

V. EVALUATION AND COMPARISON

In this section, we evaluate ECODA and compare its performance with other existing solutions. ECODA has three components, dual buffer thresholds and weighted buffer

difference for congestion detection, Flexible Queue Scheduler for packet scheduling, and Bottleneck node based source data sending rate control. Since CODA [8], are widely accepted congestion control schemes, we compared ECODA with CODA.

Table 1: NS-2 simulation parameters

Area of sensor field	1000×1000 m^2
Number of sensor nodes	35
Radio range of a sensor node	70 m
Packet length	36 bytes
IFQ length	20 packets
Transmit Power	0.660 W
Receive Power	0.395 W

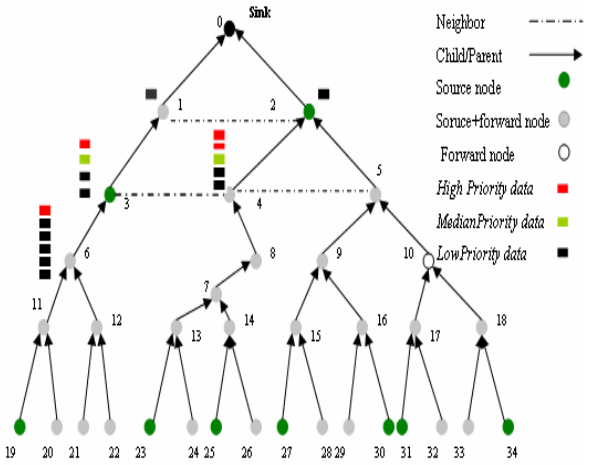


Fig. 6. Network topology for simulation.

The simulations are made in NS2 and parameters are given in Table 1.The parameters are set according to mica2 mote, which is most common sensor network developing environment. Fig. 6 is the topology used for simulation, which is a tree structure, sink is the root, and this is agreed with the sensor network topology pattern. There are 35 nodes and three static priorities are adopted in simulations, which are 0, 1, and 2, respectively. The parameters used in Def. 1: α=0.5, β=0.02, Q_{min}= (1.0/3.0)*20, Q_{max}= (2.0/3.0)*20. The maximum dynamic priority *MP* is set to 3. Three metrics, throughput, end-to-end packet delays, and weighted fairness, are selected to evaluate system performance.

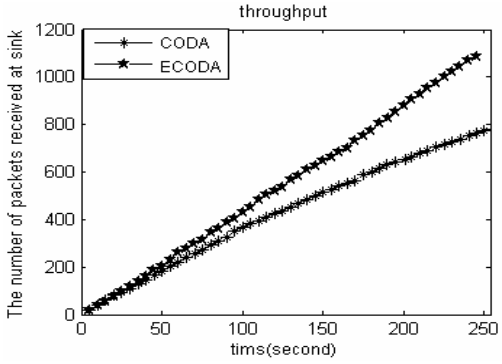


Fig. 7 Throughput comparison.

1) Throughput and end-to-end delay

Throughput and delay are simulated and shown in Fig. 7 and 8 respectively. From Fig. 7, one can see that ECODA has higher throughput than CODA. Since CODA adopts queue-length-based scheme for congestion detection, every time when buffer length exceeds a threshold, congestion is reported. However, when many nodes exceed the buffer utilization threshold simultaneously, nodes which have the highest data priority send congestion information, leading to CODA's low efficiency. When persistent congestion happens, CODA needs feedback from sink to maintain its sending rate, which has two drawbacks: 1) Too many ACKs cause extra energy consumption; 2) ACKs may be dropped due to some reasons (congestion, link variation) and cannot reach the source. ECODA solves this problem by introducing a bottleneck node based source reporting control scheme which is in implicit manner. It costs no extra energy and is more robust than CODA's closed loop multi-source regulation scheme.

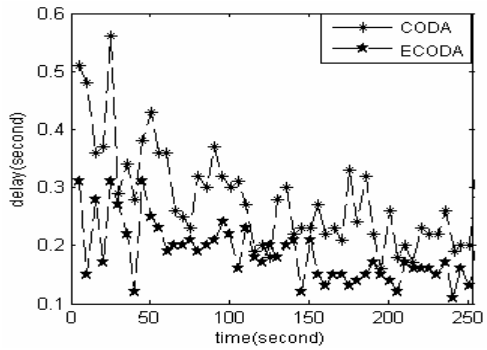


Fig. 8 Delay comparison.

The average delays of CODA and ECODA are presented in Fig. 8. Because CODA resolves persistent congestion inefficiently and cannot deal with the situation that several buffers exceed threshold simultaneously, packets experience much longer delay in CODA than in ECODA. ECODA uses dual buffer thresholds to detect congestion, which measures channel loading indirectly.

2) Weighted fairness

The most important improvement of ECODA is that it provides fairness to different class of traffic. The packet throughput and delay for different packets priorities are simulated and shown in Fig. 9 and 10. According to Rule1 and Def. 1, a packet with static priority 0, 1, or 2, its DP is distributed in [0, 1), [1, 2), or [2, 3), respectively. Since ECODA adopts dual buffer thresholds and weighted buffer difference for congestion detection, only the node which has higher priority data can send congestion information and packets are scheduled according to their priority. As indicated in Fig. 9 and 10, the packet with high priority has higher throughput and lower delay.

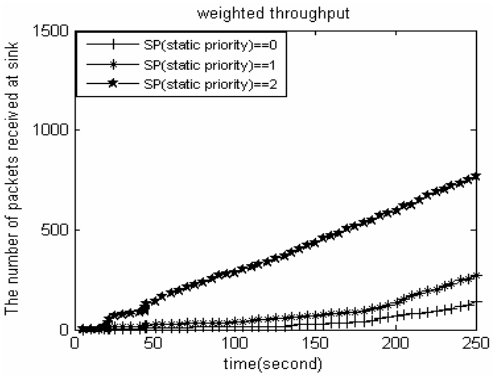


Fig. 9 Throughput for different data traffic.

ECODA provides weighted fairness for different class of traffic. It also provides fairness for packets with same static priority but its source is at different depth in the routing tree of a network. According to Def. 1, packet dynamic priority is partially determined by the number of hops to sink and delay. With the number of hops increasing, packet dynamic priority increases and coverage fidelity is guaranteed.

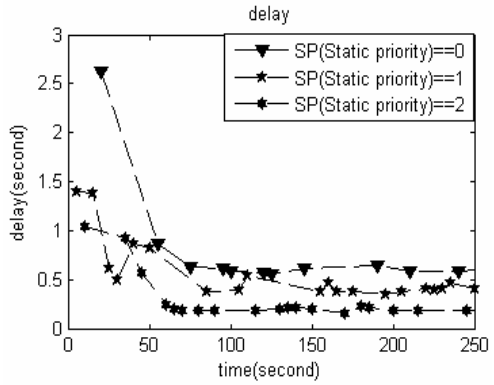


Fig. 10 The delay for different class traffic.

3) Impact of Parameters (δ_1, δ_2)

In this section, we study packet acceptance ratios for different buffer states. Since δ_1 and δ_2 are two parameters that limit buffer increasing rate when congestion happens, queue scheduler accepts or rejects packets according to buffer increasing rate and packet priority.

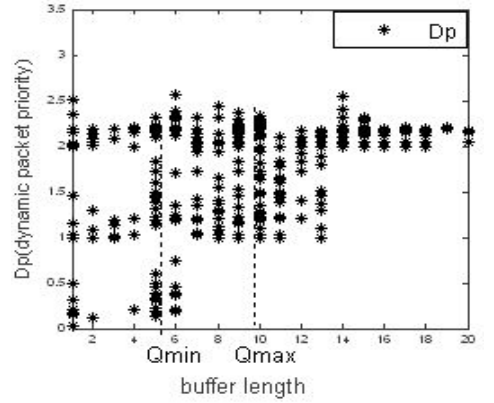


Fig. 11 Buffer occupancy and the DP of accepted packets.

When buffer length is within [*0, Q_{min}*], the buffer is in "accept state". All incoming packets are accepted. Packet priority is distributed between [*0, MP*], where *MP* is the maximum priority, as shown in Fig. 11. However, when buffer length is within [Q_{min}, Q_{max}], the buffer is "filter state". Some packets are dropped or overwritten to make buffer increasing rate less than δ_1. So high priority packets are accepted and low priority packets are dropped or overwritten. From Fig. 11 one can also see that the priority of accepted packets is within [1, 2.6].

When buffer length is within [Q_{max}, Q], the buffer is in "reject state". Some high priority packets is dropped or overwritten to make buffer increasing rate less than δ_2. Because $\delta_2=0.5*\delta_1<\delta_1$, more packets are dropped to alleviate congestion. From Fig. 11, one can see that the packet priority is within [2, 2.6]. Packets with priority less than 2 are dropped or overwritten.

VI. CONCLUSION

In this paper, a congestion control protocol (ECODA) is proposed. It detects congestion jointly using dual buffer thresholds and weighted buffer difference. ECODA has a flexible queue scheduler and packets are scheduled according to their priority. ECODA deals with transient congestion and persistent congestion efficiently. For transient congestion, it adopts hop-by-hop congestion control scheme. For persistent congestion, it adopts bottleneck node based source data sending rate control. Simulations show that ECODA achieves high link utilization and flexible fairness. It can reduce packet loss, improve energy efficiency, and lower delay.

REFERENCES

[1] H.Kim, J. Song, and S. Lee "Energy-Efficient Traffic Scheduling in IEEE 802.15.4 for Home Automation Networks," *IEEE Transaction on Consumer Electronics*, vol.53, issue.2. 369-374, May 2007.

[2] H.Oh, H.Bahn, and K. Chae, "An Energy-Efficient Sensor Routing Scheme for Home Automation Networks," *IEEE Transaction on Consumer Electronics*, vol. 51, issue. 3, pp. 836-839, August 2005.

[3] D.Lee and K.Chung, "Adaptive Duty-cycle Based Congestion Control for Home Automation Networks," *IEEE Transaction on Consumer Electronics*, vol.56, No.1, February 2010.

[4] A.Cerpa, J.Elson, M.Hamilton, and J. Zhao, "Habitat monitoring: Application drive for wireless communications technology," in Proc. *ACM SIGCOMM Workshop Data Commun*. Latin Amer, Caribbean. San Jose, Costa Rica, Apr. 2001.

[5] T.He, et al. Achieving Real-Time Target Tracking Using Wireless Sensor Networks, *ACM Trans. On Embedded Computing System*, 2007.

[6] Holman et al, 3003 R.Holman, J.Stanley and T. Ozkan-Haller, Applying video sensor networks to nearshore environment monitoring, *IEEE Pervasive Computing* 2(4) (2003), pp. 14-21

[7] Y. Sankarasubramaniam, O. Akan, I. Akyildiz, "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks," in Proc. of *ACM MobiHoc* '03.

[8] C.-Y. Wan, S.B. Eisenman, A. T. Campbell, "CODA: Congestion detection and avoidance in sensor networks", in Proc. *ACM SenSys*, Nov.2003.

[9] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E.Cayirci, "A survey on sensor networks," *IEEE Commu. Mag.*, vol.40, no. 8, pp. 102-104, Aug. 2002.

[10] E.Shih,S. Cho, and N. Ickes, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks," in Proc. *ACM MobiCom*, Rome, Italy, Jul.2001, pp. 272-287.
[11] L. Schwiebert, S. Gupta, and J. Weinmann, "Research challenges in wireless networks of biomedical sensors", in Proc, *ACM MobiCom*,Rome,Italy,Jul.2001,pp.151-165.
[12] T.He, J. Stankovic, C.Lu, and T. Abdelzaher, "Speed: A stateless protocol for real-time communication in sensor networks," in Proc. *IEEE Int. Conf. Distrib.Comput. Syst., Providence, RI*, May 2003, pp.46-55.
[13] C.Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann,and F.Silva, "Directed Diffusion for wireless sensor networking," *IEEE/ACM Trans. Netw.*,vol. 11, no.1,pp.2-16,Feb.2002.
[14] F.Stann and J. Herdemann, "RMST: Reliable data transport in sensor networks," in Pro. *1st IEEE Workshop SNPA*, Anchorage, AK, Nov. 2003, pp. 102-112.
[15] C.-T. Ee R. Bajcsy, "Congestion control and fairness for many-to-one routing in sensor networks", in Proc. *ACM Sensys*, Nov. 2004.
[16] B. Hull, K. Jamieson, H. Balakrishnan, "Mitigating congestion in wireless sensor networks", in Proc, *ACM Sensys*, Nov. 2004.
[17] S. Kim, R. Fonseca, et al., Flush: A Reliable Bulk Transport Protocol for Multihop Wireless Network, in Proc. *ACM SenSys*, Nov. 2007
[18] C.G. Wang, B Li, K.Sohraby, et al. Upstream congestion control in wireless sensor networks through cross-layer optimization , *IEEE Journal on Selected Areas in Communications* 25 (4) (2007), pp. 786–795.
[19] S. Rangwala, R. Gummadi, R. Govindan, and K. Psounis, "Interference-Aware Fair Rate Control in Wireless Sensor Networks," In Proc. of *ACM SIGCOMM'06*. Page 63-74.
[20] P. Jeongyeup, G. Ramesh "RCRT: Rate-Controlled Reliable Transport for Wireless Sensor Networks," in Proc, *ACM SenSys*, Nov. 2007.
[21] V.Rajendrand, K. Obraczka, and Garcia, "Energy-efficient, collision-free medium access control for wireless sensor networks," in Proc. 1st *ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, Los Angeles, CA USA, Nov, 2003.
[22] A. Woo, D.C. Culler, "A transmission control scheme for media access in sensor networks", in Proc. *ACM Mobicom*, July 2004.
[23] S. Tilak, N. B Abu-Ghazaleh, W. Heinzelman, "Infrastructure Tradeoffs for Sensor Networks," In Proc.*WSNA*, September 2002, Atlanta, GA, USA.
[24] W. Ye, J.Heidemann, D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," In Proc. *INFOCOM*'02, New York, USA, June 2002.
[25] J. Li, C. Blake, D. D. Couto, H. Lee, R. Morris, Capacity of ad hoc wireless networks. In Proc. of *the Seventh Annual International Conference on Mobile Computing and Networking*, Pages 61-69, July 2001.
[26] S. Navrati, R. Abhishek S. Jitae, Dynamic duty cycle and adaptive contention window based Qos-MAC protocol for wireless multimedia sensor networks , *Computer Networks*, 52 (2008), pp. 2532-2542.
[27] Y. Xu, J. Heidemann, D. Estrin, Geography-informed energy conservation for ad-hoc routing, in: *Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Rome, Italy, July 2001.

BIOGRAPHIES

Li Qiang Tao was born in Nan He Province, China. In 2005 and 2007, He received his B.S. degree and M.S. degree separately in Zhen Zhou University and Beijing Institute of Technology, both in computer science. From 2007 to current, he is pursuing his PhD degree in Institute of Computing Technology, Chinese Academy of Sciences. He research interests are in Wireless Multimedia sensor networks, traffic and congestion control, Quality of Services (QoS).

Feng Qi Yu earned his Ph.D. degree in Integrated Circuits and Systems Lab (ICSL) at UCLA. He has more than eight years of US industrial experience and more than three years of Chinese industrial experience in design, development, and implementation of CMOS integrated circuits and systems. He has done several projects in CMOS RF integrated systems, such as wireless LAN, GPS, land mobile phone, CMOS sensor, and RFID. Before joining Shenzhen Institute of Advanced Technology in June 2006 as a full professor, he worked at Rockwell Science Center (USA) as a design engineer, Intel (USA) as an analog circuit design engineer, Teradyne (USA) as a Sr. IC design engineer, Valence (USA) as a Sr. principal engineer, and Suzhou CAS IC Design Center (China) as a VP and RF department director. His R&D interests are in the field of CMOS RF circuit design, CMOS sensor design, wireless sensor networks, wireless communication systems, RFID.