Upstream Congestion Control in Wireless Sensor Networks Through Cross-Layer Optimization

C. Wang, *Member, IEEE*, B. Li, *Senior Member, IEEE*, K. Sohraby, *Senior Member, IEEE*, M. Daneshmand, *Member, IEEE*, and Y. Hu

***Abstract*— Congestion in wireless sensor networks not only causes packet loss, but also leads to excessive energy consumption. Therefore congestion in WSNs needs to be controlled in order to prolong system lifetime. In addition, this is also necessary to improve fairness and provide better quality of service (QoS), which is required by multimedia applications in wireless multimedia sensor networks. In this paper, we propose a novel upstream congestion control protocol for WSNs, called Priority-based Congestion Control Protocol (PCCP). Unlike existing work, PCCP innovatively measures congestion degree as the ratio of packet inter-arrival time along over packet service time. PCCP still introduced node priority index to reflect the importance of each sensor node. Based on the introduced congestion degree and node priority index, PCCP utilizes a cross-layer optimization and imposes a hop-by-hop approach to control congestion. We have demonstrated that PCCP achieves efficient congestion control and flexible weighted fairness for both single-path and multi-path routing, as a result this leads to higher energy efficiency and better QoS in terms of both packet loss rate and delay.**

***Index Terms*— Congestion control, cross-layer optimization, wireless sensor networks.**

I. Introduction

A WIRELESS sensor network (WSN) [1] consists of one or more sinks and large number of sensor nodes scattered in an area. With the integration of information sensing, computation, and wireless communication, sensor nodes can sense the physical phenomenon, (pre-)process the "raw" information, share the processed information with neighboring nodes, and report information to the sink. The downstream traffic from the sink to the sensor nodes usually is an one-to-many multicast. The upstream traffic from sensor nodes to the sink is a many-to-one communication. The upstream traffic can be classified into four categories: event-based, continuous, query-based, and hybrid. In event-based delivery, a sensor node does event reporting if and only if target events occur. The sensor data for the event usually has very small size. Sensor nodes may need to periodically report to the sink and generate continuous data transmission in some cases. This is continuous delivery. In query-based delivery, sensory data is stored inside network and is queried by and then transmitted to the sink on demand. Practical applications might trigger hybrid data delivery including event-based, continuous, and query-based. For example, an application would be not only interested in all temperature changes (continuous delivery), but also interested in some specific temperature change (below zero degree, etc; event-based delivery) as well as querying temperature at specific time (query-based delivery). Due to the convergent nature of upstream traffic, congestion more probably appears in the upstream direction. In addition, upstream traffic could have high bit rate with the introduction and development of wireless multimedia sensor networks (WMSNs) [2]. Such high speed upstream traffic is prone to cause congestion which will impair QoS of multimedia applications in WMSNs. This paper studies upstream congestion control for WSNs, especially for multimedia applications of WMSNs.

Two types of congestion could occur in WSNs [3] (see Fig. 1). The first type is node-level congestion that is common in conventional networks. It is caused by buffer overflow in the node and can result in packet loss, and increased queuing delay. Packet loss in turn can lead to retransmission and therefore consumes additional energy. For WSNs where wireless channels are shared by several nodes using CSMA-like (Carrier Sense Multiple Access) protocols, collisions could occur when multiple active sensor nodes try to seize the channel at the same time. This can be referred to as link-level congestion. Link-level congestion increases packet service time, and decreases both link utilization and overall throughput, and wastes energy at the sensor nodes. Both node-level and link-level congestions have direct impact on energy-efficiency and QoS.

Therefore congestion must be efficiently controlled. Congestion control protocol efficiency depends on how much it can achieve the following objectives: (i) energy-efficiency requires to be improved in order to extend system lifetime. Therefore congestion control protocols need to avoid or reduce packet loss due to buffer overflow, and remain lower control overhead that consumes less energy; (ii) it is also necessary to support traditional QoS metrics such as packet loss ratio,

Manuscript received May 15, 2006; revised November 29, 2006. An earlier version of this paper was published at the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC2006), June 5-7, 2006, Taichung, Taiwan. The research at the University of Arkansas was supported in part by AT&T Labs Research. B. Li's work was supported in part by grants from RGC under the contracts HKUST6104/04E, HKUST6165/05E, and HKUST6164/06E, and by grants from NSF China under the contracts 60429202 and 60573115.

C. Wang is with the Department of Electrical Engineering, University of Arkansas, Fayetteville, AR 72701 USA (e-mail: cgwang@ieee.org).

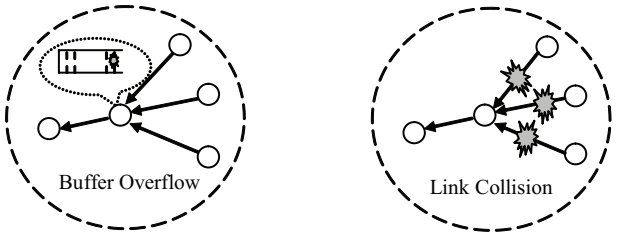
B. Li is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong, China (e-mail: bli@cse.ust.hk).

K. Sohraby is with the Department of Electrical Engineering, University of Arkansas, Fayetteville, AR 72701 USA (e-mail: sohraby@uark.edu).

M. Daneshmand is with AT&T Labs Research, Florham Park, NJ 07932 USA (e-mail: daneshmand@att.com).

Y. Hu is with the College of Information, South China Agricultural University, Guangzhou, China (e-mail: ymhu@scau.edu.cn).

Digital Object Identifier 10.1109/JSAC.2007.070514.



(a) node-level congestion (b) link-level congestion

Fig. 1. Congestion in wireless sensor networks.

packet delay, and throughput. For example, multimedia applications in WMSNs [2] require not only packet loss guarantee but also delay guarantee; (iii) fairness needs to be guaranteed so that each node can achieve fair throughput. Most of the existing work [3] [4] guarantees simple fairness in that every sensor node obtains the same throughput to the sink. In fact, sensor nodes might be either outfitted with different sensors or geographically deployed in different place and therefore they may have different importance or priority and need to gain different throughput. Therefore weighted fairness is required.

There are two general approaches to control congestion: network resource management and traffic control. The first approach tries to increase network resource to mitigate congestion when it occurs. In wireless network, power control and multiple radio interfaces can be used to increase bandwidth and weaken congestion. For example, virtual sinks in Siphon [5] have two radio interfaces: one primary low-power mote radio with smaller bandwidth and another long-rage radio with larger bandwidth. When congestion occurs, the long-rage radio is used as a shortcut or "siphon" to mitigate congestion. With this approach, it is necessary to guarantee precise and exact network resource adjustment in order to avoid over-provided resource or under-provided resource. However this is a hard task in wireless environments. Unlike the approaches based on network resource management, traffic control implies to control congestion through adjusting traffic rate at source nodes or intermediates nodes. This approach is helpful to save network resource and more feasible and efficient when exact adjustment of network resource becomes difficult. Most existing congestion control protocols belong to this type. According to the control behavior, there are two general methods for traffic control in WSNs: end-to-end and hop-by-hop. The end-to-end control can impose exact rate adjustment at each source node and simplify the design at intermediate nodes; however, it results in slow response and relies highly on the round-trip time (RTT). In contrast, the hop-by-hop congestion control has faster response. However, it is usually difficult to adjust the packet forwarding rate at intermediate nodes mainly because packet forwarding rate is dependent on MAC protocol and could be variable.

This paper investigates the problem of upstream congestion control in WSNs through traffic control. We believe that, in WSNs, every packet might contain useful information, which can be utilized through packet-based computation, and furthermore utilized to enhance congestion control. The packet-based computation could be practical for WSNs considering: 1) a WSN generally has small packet forwarding rate and therefore the unit time to forwarding one packet could be long enough for a sensor node to perform certain computation even though its computation capability is limited; 2) although WSNs, at most time, are deployed with limited energy and lacks recharging approach. The packet-based computation is still preferred for a sensor node since it is generally known that the computation consumes less energy than communication [1]. Moreover the packet-based computation could provide useful information to reduce or avoid useless communication and in turn compensate the energy it consumes. We address both node-level and link-level congestion in this paper and propose a new Priority-based Congestion Control Protocol (PCCP), which employs packet-based computation to optimize congestion control for a WSN. Our contribution includes:

1) PCCP enables cross-layer optimization. Specifically we propose to exploit packet inter-arrival time and packet service time to produce a measure of congestion. We observe by incorporating information of packet inter-arrival time and the packet service time, it can directly reflect level of activity at the node or at the link, both types of congestion can be captured through a single parameter, referred to as congestion degree, which is defined as the ratio of service time over inter-arrival time.
2) PCCP, as a congestion control protocol, is designed to work under both single-path routing and multi-path routing scenarios. None of existing congestion control protocol for WSNs considers the support of multi-path routing.
3) We argue that each sensor node should have different priority since sensor nodes might be installed with different kinds of sensors in an environment. Through priority-based congestion control, PCCP realizes priority-dependent weighted fairness which allows sensor nodes to receive priority-dependent throughput. This model has not been
4) PCCP results in lower buffer occupancy, which can avoid and/or reduce packet loss. It achieves high link utilization and low packet delay.

The remainder of this paper is organized as follows: Section II briefly introduces related work. Section III describes system models considered in this paper. Section IV presents PCCP in detail. Section V provides simulation results. Finally, Section VI concludes the paper.

II. Related Work

In recent years, several new congestion control protocols have been proposed for WSNs [6]. Among of them, CCF (Congestion Control and Fairness) [3] uses packet service time to deduce the available service rate and therefore detects congestion in each intermediate sensor node. Congestion information, that is packet service time in CCF, is implicitly reported. CCF controls congestion in a hop-by-hop manner and each node uses exact rate adjustment based on its available service rate and child node number. CCF guarantees simple fairness. That means each node receives the same throughput. However the rate adjustment in CCF relies only on packet service time which could lead to low utilization when some

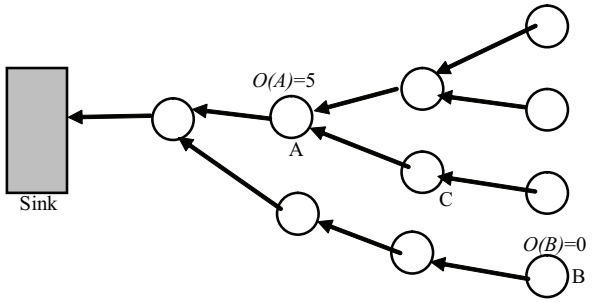


Fig. 2. Network mode-a logical topology established by routing protocols.

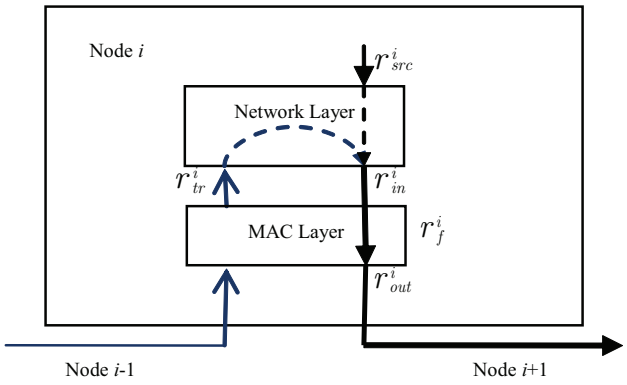


Fig. 3. General node mode in WSNs.

sensor nodes do not have enough traffic or there is a significant packet error rate (PER).

In Fusion [4], congestion is detected in each sensor node based on measurement of queue length. The node that detects congestion sets a CN (congestion notification) bit in the header of each outgoing packet. Once the CN bit is set, neighboring nodes can overhear it and stop forwarding packets to the congested node so that it can drain the backlogged packets. This non-smooth rate adjustment could impair link utilization as well as fairness, although Fusion has a mechanism to limit the source traffic rate and a prioritized MAC algorithm to improve fairness. Siphon [5] also infers congestion based on queue length in intermediate nodes, but it uses traffic redirection to weaken congestion. There is no rate adjustment in Siphon.

Congestion Detection and Avoidance (CODA), [7], detects congestion based on buffer occupancy as well as wireless channel load. CODA designs both open-loop and closed-loop rate adjustment, and the algorithm used to adjust traffic rate works in a way like additive increase multiplicative decrease (AIMD) and inevitably leads to the existence of packet loss similar to the traditional TCP protocol.

Adaptive Rate Control (ARC), [8], is an LIMD-like (linear increase and multiplicative decrease) algorithm. In ARC, if an intermediate node overhears that the packets it sent previously are successfully forwarded again by its parent node, it will increase its rate by a constant α. Otherwise it will multiply its rate by a factor β where $0 < \beta < 1$. ARC does not use explicit congestion detection or explicit congestion notification and therefore avoids use of control messages. However the coarse rate adjustment could result in tardy control and introduce packet loss.

Those existing congestion control protocols for WSNs have two primary limitations. First, they only guarantee simple fairness, which means that the sink receives the same throughput from all nodes. However, sensor nodes may have different priority or importance due to either their functions or the location at which they are deployed. Second, most current techniques only support single-path routing. However multi-path routing [9] is generally used to improve resilience, energy-efficiency, and reliability.

III. System Models

This section describes network and node models, as shown in Figs. 2 and 3, respectively.

A. Network Model

This paper addresses upstream congestion control for a WSN that supports multi-path routing. For ease of discussion, we first consider single path routing as shown in Fig. 2. Later, we will extend it to multi-path routing. The network model to be investigated in this paper is depicted in Fig. 2, where sensor nodes are supposed to generate continuous data, for example multimedia flows produced by sensor nodes installed with digital camera, and form many-to-one convergent traffic in the upstream direction. They are assumed to implement CSMA-like MAC protocol. Each sensor node could have two types of traffic: source and transit. The former is locally generated at each sensor node, while the latter is from other nodes. Therefore each sensor node can be a source node and/or intermediate node. When a sensor node has offspring nodes and transit, it is a source node as well as an intermediate node. On the other hand, it is only a source node if it has no offspring nodes, and therefore only has source traffic. The offspring node of a particular ancestor node is defined as the node whose traffic is routed through this particular ancestor node. If an offspring node directly connects to its ancestor node, this offspring node is called child node and its ancestor node is called parent node. For example in Fig. 2, node A has 5 offspring nodes and therefore it plays the role of a source node as well as an intermediate node, simultaneously. Node C is the child node of node A, which in turn is the parent node of node C. However node B has zero offspring node and is only a source node. For a particular sensor node i, we use $O(i)$ to denote the total number of its offspring nodes. In the remainder of this paper, when we refer to sensor nodes, we mean that they act as both a source node and an intermediate node unless otherwise indicated.

B. Node Model

Fig. 3 presents the queuing model at a particular sensor node i with single-path routing. The transit traffic of node $i(r^i_{tr})$ is received from its child nodes such as node $i-1$ through its MAC layer. The source traffic is locally generated with the rate of r^i_{src}. Both the transit traffic and the source traffic converge at the network layer before being forwarded to node $i+1$, which is the parent node of node i. Packets could be queued at the MAC layer if total input traffic rate $(r^i_m = r^i_{src} + r^i_{tr})$

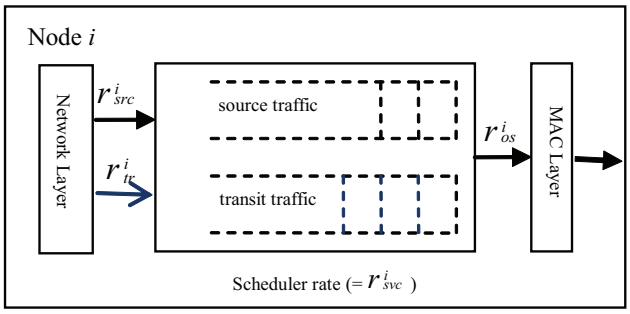


Fig. 4. Node mode in PCCP protocol.

exceeds packet forwarding rate at the MAC layer $\left(r_f^i\right)$. The packet forwarding rate r_f^i depends on the MAC protocol itself. With the assumption of CSMA-like protocol, the number of active sensor nodes as well as their traffic density influences r_f^i. In Fig. 3, r_{out}^i is the packet output rate at the node i towards node $i+1$. If r_{in}^i is smaller than r_f^i, r_{out}^i will equal r_{in}^i. Otherwise if $r_{in}^i > r_f^i$, then r_{out}^i will be close to r_f^i. Therefore, $r_{out}^i = \min\left(r_{in}^i, r_f^i\right)$. This property can be utilized to indirectly reduce r_{out}^i through reducing r_{in}^i. In fact, the output traffic at node i is part of transit traffic at the node $i+1$. Therefore reduction of r_{out}^i implies a decrease of r_{tr}^{i+1}.

If packet input rate r_{in}^i exceeds packet forwarding rate r_f^i, then there will be backlogged packets inside node i and node-level congestion takes place. At this time, we need to reduce r_{in}^i and/or increase r_f^i. While r_f^i can be increased through adjusting MAC protocols, it is much easier to lower r_{in}^i through throttling either r_{src}^i, r_{tr}^i or both of them. The source rate r_s^i can be reduced locally by changing sampling (or reporting) frequency. The transit traffic r_{tr}^i can be indirectly reduced through rate adjustment at the node $i-1$.

On the other hand, if there is collision on the link around the node i, then node i and its neighboring nodes should reduce channel access in order to prevent further link-level congestion. Although this task may be performed through the MAC, yet it is easier to reduce r_{in}^i.

This paper designs a novel congestion control approach through flexible and distributed rate adjustment in each senor node as shown in Fig. 4. It introduces a scheduler between network layer and MAC layer, which maintains two queues: one for source traffic and another for transit traffic. The scheduling rate is denoted as r_{svc}^i. A weighted fair queuing (WFQ) or weighted round robin (WRR) algorithm maintaining two queues can be practically used to guarantee fairness between source and transit traffic, as well as among all sensor nodes. The priority index of source traffic and transit traffic, which will be defined in next section, is used as the weight, respectively, for source traffic queue and transit traffic queue. By adjusting the scheduling rate r_{svc}^i, PCCP realizes an efficient congestion control while maintaining the MAC protocol parameters unchanged and therefore works well with any CSMA-like MAC protocol.

IV. PCCP Protocol

PCCP is designed with such motivations: 1) In WSNs, sensor nodes might have different priority due to their function or location. Therefore congestion control protocols need guarantee weighted fairness so that the sink can get different, but in a weighted fair way, throughput from sensor nodes. 2) With the fact that multi-path routing [9] is used to improve system performance of WSNs, congestion control protocols need to be able to support both single-path routing and multi-path routing. 3) Congestion control protocols need to support traditional QoS in terms of packet delivery latency, throughput and packet loss ratio, which is required by multimedia application in WMSNs [2].

PCCP tries to avoid/reduce packet loss while guaranteeing weighted fairness and supporting multi-path routing with lower control overhead. PCCP consists of three components: intelligent congestion detection (ICD), implicit congestion notification (ICN), and priority-based rate adjustment (PRA).

ICD detects congestion based on packet inter-arrival time and packet service time. The joint participation of inter-arrival and service times reflect the current congestion level and therefore provide helpful and rich congestion information. To the best of our knowledge, jointly use of packet inter-arrival and packet service times as in ICD to measure congestion in WSNs has not been done in the past.

PCCP uses implicit congestion notification to avoid transmission of additional control messages and therefore help improve energy-efficiency. In ICN, congestion information is piggybacked in the header of data packets. Taking advantage of the broadcast nature of wireless channel, child nodes can capture such information when packets are forwarded by their parent nodes towards the sink, assuming that there is no power control and the omni-directional antenna is used.

Finally, PCCP designs a novel priority-base algorithm employed in each sensor node for rate adjustment, in order to guarantee both flexible fairness and throughput, called PRA, In PRA, each sensor node is given a priority index. PRA is designed to guarantee that: (1) The node with higher priority index gets more bandwidth and proportional to the priority index; (2) The nodes with the same priority index get equal bandwidth. (3) A node with sufficient traffic gets more bandwidth than one that generates less traffic. The use of priority index provides PCCP with high flexibility in realizing weighted proportional fairness. This paper does not consider how to choose priority index for each sensor node, yet it assumes the priority index has been determined and how to use it to enhance congestion control. Before describing the new congestion control protocol, the following provides three definitions related to the priority index:

Definition 1: Source Traffic Priority ($SP(i)$) – The source traffic priority at sensor node i is used to represent the relative priority of local source traffic at node i. $SP(i)$ is independent of the offspring node number of the node i.

How to determine $SP(i)$ is dependent on the specific applications. For example, if the sink wants to receiver the same number of packets from each sensor node, the same priority index can be set for all nodes. On the other hand, if the sink wants to receive more detailed sensory data from a particular set of sensor nodes, such sensor nodes can be assigned a higher priority index and therefore allocated higher bandwidth. Also if some sensor nodes are outfitted with kinds of sensors, they can be assigned with a higher priority index since kinds of sen-

sors might generate more sensory data than nodes with a single kind of sensor. In this paper, our congestion control protocol is designed based on $SP(i)$ and realize proportional fairness in the way: $SP(i)/SP(j) = Thruput(i)/Thruput(j)$, where $Thruput(i)$ is the throughput of node i.

Definition 2: Transit Traffic Priority ($TP(i)$) – The transit traffic priority at sensor node i is used to represent the relative priority of transit traffic routed through node i. In case of single path routing, $TP(i)$ equals the sum of source traffic priority of each offspring node or $TP(i)$ still equals the sum of global priority of all child nodes of node i, where global priority is defined below. Regarding multi-path routing [8], $TP(i)$ will be impacted by the number of paths and multi-path routing policy. We will discuss multi-path routing in detail later in Section IV-B.

Definition 3: Global Priority ($GP(i)$) – The global priority refers to the relative importance of the total traffic at each node i. Therefore $GP(i) = SP(i) + TP(i)$.

A. Single Path Routing

1) Intelligent Congestion Detection (ICD): In order to precisely measure local congestion level at each intermediate node, we proposes intelligent congestion detection (ICD) that detects congestion based on mean packet inter-arrival $\left(t_a^i\right)$ and mean packet service times $\left(t_s^i\right)$ at the MAC layer. Here packet inter-arrival time is defined as the time interval between two sequential arriving packets from either source or for the transit traffic, and the packet service time is referred to as the time interval between when a packet arrives at the MAC layer and when its last bit is successfully transmitted. t_s^i covers packet waiting, collision resolution, and packet transmission times at the MAC layer. t_a^i as well as t_s^i can be measured at each node i on a packet-by-packet basis.

Based on the t_a^i and t_s^i, ICD defines a new congestion index, congestion degree $d(i)$, which is defined as the ratio of average packet service time over average packet inter-arrival time over a pre-specified time interval in each senor node i as follows:

$$d(i) = t_s^i / t_a^i. \tag{1}$$

The congestion degree is intended to reflect the current congestion level at each sensor node. When the inter-arrival time is smaller than the service time, the congestion degree $d(i)$ is larger than 1 and the node experiences congestion. Otherwise when the congestion degree $d(i)$ is smaller than 1, the incoming rate is below the outgoing rate, and hence congestion abates. Therefore congestion degree can adequately represent congestion condition and provide helpful information in order to realize efficient congestion control. The congestion degree $d(i)$ can inform the child nodes about the traffic level to be increased or decreased by adjusting their transmission rate. In Eq. (1), t_a^i and t_s^i at each node i are measured using EWMA (exponential weighted moving average) algorithm as follows.

In the process of determining the congestion degree, t_a^i is updated periodically whenever there are N_p (=50 in PCCP) new packets arriving as follows:

$$t_a^i = (1 - w_a) * t_a^i + w_a * T_{N_p}/N_p, \tag{2}$$

where $0 < w_a < 1$ is a constant (= 0.1 in PCCP examples to be discussed later), T_{N_p} is the time interval over which the measurements are performed, and within which the N_p new packets arrive.

Also, t_s^i is updated each time a packet is forwarded as follows:

$$t_s = (1 - w_s) * t_s + w_s * t_s^{'}, \tag{3}$$

where $0 < w_s < 1$ is a constant (again set to 0.1 in the future examples), $t_s^{'}$ is the service time of the packet just transmitted.

2) Implicit Congestion Notification (ICN): There are two approaches to propagate congestion information: Explicit Congestion Notification (ECN) and Implicit Congestion Notification (ICN). The explicit congestion notification uses special control messages and inevitably introduces additional overhead. In contrast, implicit congestion notification piggybacks congestion information in the header of data packets. Taking advantage of the broadcast nature of wireless channel, child nodes listen to their parent node to get congestion information. In the implicit congestion notification, transmission of an additional control message is avoided.

PCCP uses ICN at each sensor node i to piggyback congestion information in the header of data packets to be forwarded. Notification is triggered by either of the two events: (1) the number of forwarded packets by a node exceeds a threshold ($= O(i) * N_p$ in PCCP); (2) the node overhears a congestion notification from its parent node. The piggybacked information at a sensor node i includes mean packet service time $\left(t_a^i\right)$, mean packet inter-arrival time $\left(t_a^i\right)$, global priority $GP(i)$, and the number of offspring node $O(i)$. A node then computes its global priority index by summating its source traffic priority index and all the global priority index of its child nodes, which is piggybacked in the received data packets.

3) Priority-Based Rate Adjustment (PRA): As shown in Fig. 4, we introduce a scheduler with two sub-queues between the network layer and the MAC layer. If the scheduling rate r_{svc}^i is kept below the MAC forwarding rate r_f^i, the output rate will approximately equal the output rate r_{out}^i. Therefore, through adjusting the scheduling rate r_{svc}^i, congestion could be avoided or mitigated.

There are generally two ways to perform this task. The first is Additive Increase Multiplicative Decrease or AIMD such as commonly used in the traditional TCP protocol or its variants. At this time, congestion information indicates whether there is congestion or not which can be transferred using a binary congestion notification (CN) bit. Unfortunately, AIMD is unable to exactly adjust the transmission rate because CN bit provides limited information. However when nodes are specifically informed as to how much to increase or decrease their rates, exact rate-adjustment becomes possible.

Congestion degree $d(i)$ and priority indices ($TP(i)$ and $GP(i)$) introduced here provides more information than the CN bit and enables exact rate adjustment. PRA needs to adjust the scheduling rate r_{svc}^i and the source rate r_{svc}^i at each sensor node after overhearing congestion notification from its parent node, in order to control both link-level congestion and node-level congestion.

First we consider single-path routing, where each node (i) has only one parent node (let it be called $p_{i,0}$). Node

01 *Initialization*()
02 $d'(p_{i,0}) = 1, O'(p_{i,0}) = 0, r^i_{svc} = r_0$;
03 *GetSvcRate*($t_s^{p_{i,0}}$, $t_a^{p_{i,0}}$, $GP(p_{i,0}), O(p_{i,0}), r^i_{svc}, GP(i)$)
04 $d(p_{i,0}) = t_s^{p_{i,0}} / t_a^{p_{i,0}}$;
05 $total\_rate = 1/t_s^{p_{i,0}}$;
06 *If*($O(p_{i,0}) < O'(p_{i,0})$) $r^i_{svc} = r^i_{svc} / d(p_{i,0})$;
07 *If*($O(p_{i,0}) > O'(p_{i,0})$) $r^i_{svc} = total\_rate * \frac{GP(i)}{GP(p_{i,0})}$;
08 *If*($O(p_{i,0}) = O'(p_{i,0})$) {
09 *If*($d(p_{i,0}) < d'(p_{i,0})$) $r^i_{svc} = r^i_{svc} / d(p_{i,0})$;
10 *If*($d(p_{i,0}) > d'(p_{i,0})$) $r^i_{svc} = base\_rate * GP(i) / GP(p_{i,0})$;
11 }
12 $d'(p_{i,0}) = d(p_{i,0}), O'(p_{i,0}) = O(p_{i,0})$;
13 $r^i_{svc} = \min(r^i_{svc}, 1/t^i_s)$;
14 *return* $r^i_{svc} * \mathrm{h}$;
15 *GetSrcRate*(r^i_{svc})
16 $r^i_{src} = r^i_{svc} * SP(i) / GP(i)$;
17 *return* r^i_{src};

Fig. 5. Rate-adjustment algorithm in PCCP used by node i to calculate its scheduling rate and source rate under single-path routing.

i obtains the congestion information (including $t_s^{p_{i,0}}$, $t_a^{p_{i,0}}$, $GP(p_{i,0})$, and $O(p_{i,0})$) about node $p_{i,0}$ through the packets forwarded by $p_{i,0}$. Node i then updates its local scheduling rate and its source rate, respectively, using the procedures of $GetSvcRate()$ and $GetSrcRate()$ as in Fig. 5. The initial scheduling rate is set at a small initial value r_0. We consider four cases in $GetSvcRate()$:

1) First, when some offspring node becomes idle, packet inter-arrival time $t_a^{p_{i,0}}$ at node $p_{i,0}$ will increase and the congestion degree $d(p_{i,0})$ at the node $p_{i,0}$ will decrease, in this case, node i could scale-up its scheduling rate according to $d(p_{i,0})$ in order to improve link utilization (Line 6 of Fig. 5).
2) When new nodes become active, packet inter-arrival times $t_a^{p_{i,0}}$ will decrease and therefore congestion degree $d(p_{i,0})$ will increase. Node i may need to reduce its scheduling rate in this case. Keeping in mind the fact that all the offspring nodes of the node i are subset of offspring nodes of the node $p_{i,0}$ because node i is the child node of the node $p_{i,0}$, node i sets its scheduling rate to the maximum allowable rate in order to guarantee fairness and high link utilization. The new scheduling rate is dependent on the global priority index at both node i and its parent node $p_{i,0}$ (Line 7 of Fig. 5).
3) When the number of offspring nodes $O(p_{i,0})$ remains constant but some nodes don't have enough traffic, congestion degree $d(p_{i,0})$ will become smaller, and therefore node $p_{i,0}$ can scale-up its scheduling rate according to $d(p_{i,0})$ in order to improve link utilization (Line 9 of Fig. 5).
4) In case Four, $O(p_{i,0})$ still remains the same but some nodes with small traffic, will produce more traffic. At this time, congestion degree $d(p_{i,0})$ will increase and possibly be larger than 1, and node i resets its scheduling rate back to the allowable rate as in case 2 so that to mitigate congestion while maintaining high link utilization (Line 10 of Fig. 5).

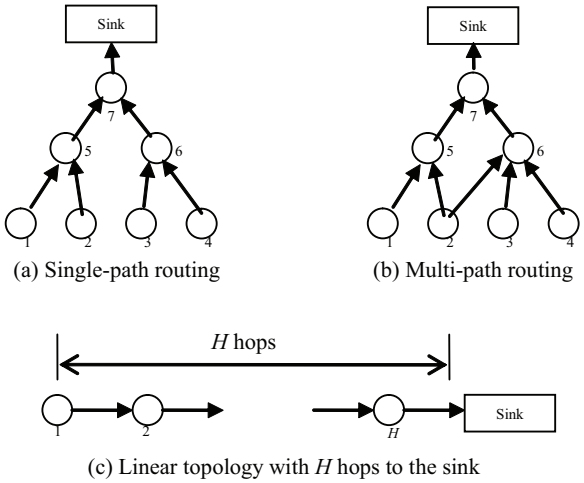


Fig. 6. Simulation topology.

After scheduling rate is obtained, source rate of the node i (r^i_{src}) is updated based on its source traffic priority index $SP(i)$ and global priority index $GP(i)$ as shown in Line 15 of Fig. 5. In Line 14 of Fig. 5, h is a parameter smaller than but close to 1, which is used to maintain a small queue length and high throughput. In the future examples, h equals 0.98.

In the above rate adjustment process, global priority index is used to update the scheduling rate while the source traffic priority index is used to calculate source rate. Therefore, the rate adjustment has the following properties: (1) nodes with the same source traffic priority index get the same source rate; (2) nodes with a larger source priority index receive higher source rate and higher bandwidth.

B. Multi-Path Routing

We have introduced a rate adjustment process for the scenario of single-path routing in the previous section. In case of multi-path routing [9], a child node could have multiple parents and packets will be forwarded from the child node to all parents according to certain policy including uniform forwarding or proportional forwarding, which depends on the specific multi-path routing protocol. Under this case, the transit traffic priority ($TP(i)$) will be impacted and should be calculated in a way different from the case of single-path routing. For example, in Fig. 6(b), node 2 has two paths connecting to node 5 and node 6, simultaneously and respectively. In this case, node 5 has two child nodes and node 6 has three. If we assume each node has the same source priority index, and let it be 1 without loss of generality. According to the rules in the previous section for single-path routing, the transit traffic priority of node 5 and node 6 might be 2 and 3 respectively. This way is, however, not wise, since both node 5 and node 6 gets only parts of traffic from node 2. If node 2 sends packets uniformly to both node 5 and node

6, each of them will get half. It looks like that node 5 has 1.5 child nodes and node 6 has 2.5 ones, and their transit traffic priority should be, respectively, 1.5 and 2.5. Therefore their global priority index will be, respectively, 2.5 and 3.5. As a result, the ratio of obtained packet forwarding rate of node 5 over node 6 will be 2.5 and 3.5. This way can guarantee proportional fairness for sensor nodes.

Assume each node i in a WSN has $M_i\,(\geq 1)$ parents resulted from multi-path routing protocol, and the node i will proportionally forward its packets to each of parents $p_{i,j}\,(1 \leq j \leq M_i)$ according to a weight $w_{i,j}$, where $w_{i,j}$ satisfies $0 \leq w_{i,j} \leq 1$ and $\sum_{j=1}^{M} w_{i,j} = 1$. When $M_i = 1$, $w_{i,1} = 1$. If $w_{i,j} = 0.2$, this would mean that 20% of node i traffic will be routed to parent $p_{i,j}$. Therefore we give out the calculation of transit traffic priority for multi-path routing as follows.

Definition 4: Transit Priority $TP\,(i)$ (In Case of Multi-Path Routing) - Suppose the node i has C_i child nodes. If $C_i = 0$, $TP\,(i) = 0$. When $C_i > 0$, let $id\,(c)$ denote the identity of the c-th child node ($1 \leq c \leq C_i$), and $w_{id(c),i}$ is the weight in child node c. $w_{id(c),i}$ represents the traffic percentage of child node c, which will be routed to the node i. Then, $TP\,(i)$ in multi-path routing is calculated as:

$$TP\,(i) = \sum_{c=1}^{C_i} \left[GP\,(id\,(c)) * w_{id(c),i}\right]. \tag{4}$$

In multi-path routing, global priority still equals the sum of source traffic priority and transit traffic priority. The source traffic priority keeps constant. Therefore with $TP\,(i)$ calculated according to Eq. (4), we can obtain $GP\,(i) = SP\,(i) + TP\,(i)$ in the same way as in the single-path routing. Then, rate adjustment algorithm in Fig. 5 can be employed independently to calculate the rate for each path. Then the scheduling rate r_{svc}^{i} could be determined as the sum of calculated rates on all paths.

Let r_{svc}^{i} be the rate at the node i to send packets to parent $p_{i,j}$. Its initial value can be set to $r_0 * w_{i,j}$. $r_{svc}^{i,j}$ can be calculated whenever node i gets congestion information $\left(t_s^{p_{i,j}}, t_a^{p_{i,j}}, GP\,(p_{i,j}), O\,(p_{i,j})\right)$ from its parent $p_{i,j}$ using the same procedure $GetSvcRate()$ described in Fig. 5, but with different input parameters as shown in Eq. (5). Then the scheduling rate is:

$$r_{svc}^{i} = \sum_{j=1}^{M_i} r_{svc}^{i,j}. \tag{6}$$

The source rate r_{svc}^{i} is updated using the same method as the single-path routing (see Line 16 of Fig. 5). In addition, node i piggybacks $GP\,(i) * w_{i,j}$ instead of $GP\,(i)$ in each packet routed to its parent $p_{i,j}$ so that the parent $p_{i,j}$ can correctly deduce its transit traffic priority (according to Eq. (4)) and furthermore global priority index.

In this paper, we do not consider how to configure the weight $w_{i,j}$; however it could be dependent on routing protocols and/or load-balancing policies. Some possible selection of weight $w_{i,j}$, for example, could be as follows: 1) If we enforce even routing and uniform load-balancing for all paths, each path could be set with the same weight; 2) On the other hand, load-balancing could be proportional to the length of every path, the minimal residual energy of sensor nodes on every path, the buffer size of sensor nodes on every path, or the available bandwidth on every path. In this case, the weight $w_{i,j}$ will vary for every path. If a path has shorter length or the sensor nodes on this path have more residual energy, the weight assigned to this path can be bigger than others; 3) the weight $w_{i,j}$ could be dependent on link quality on every path. The path with smaller bit error rate (BER) can be assigned a larger $w_{i,j}$.

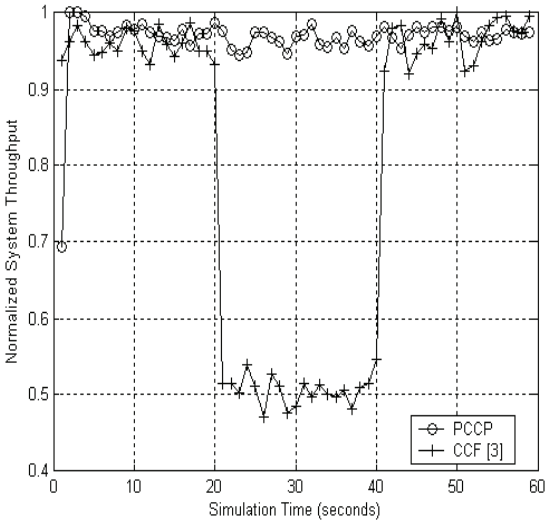
C. Discussion

Generally speaking, the performance of congestion control protocols mostly depends on whether congestion can be detected in time or even correctly predicted in advance, whether congestion degree can be accurately measured, whether the detected or predicted congestion can be notified quickly to the nodes generating heavy traffic, and whether these nodes can trigger correct rate adjustment. In PCCP, the term defined in Eq. (1) captures congestion degree at intermediate nodes much more precisely than queue-length-based congestion detection in existing work [4], [5], [7]. However the speed with which PCCP detects congestion is dependent on how quickly packet inter-arrival and service time can be correctly measured according to Eqs. (2) and (3). From Eqs. (2) and (3), we can see that w_a and w_s influence measurement speed and measurement accuracy. The larger value of them, the quickly to measure t_a^i and t_s^i but the less accuracy. There is a tradeoff to choose values for w_a and w_s. In this paper they are set to 0.1. PCCP uses hop-by-hop implicit congestion notification. In addition, the algorithm in Fig. 5 realizes hop-by-hop exact rate adjustment so as to guarantee high link utilization and avoid congestion. Although theoretical analysis is helpful to quantitatively discover the performance of PCCP, in this paper we first demonstrate through simulations in the next section that PCCP with these features and the default parameter values achieves fast and accurate congestion detection and efficient congestion control.

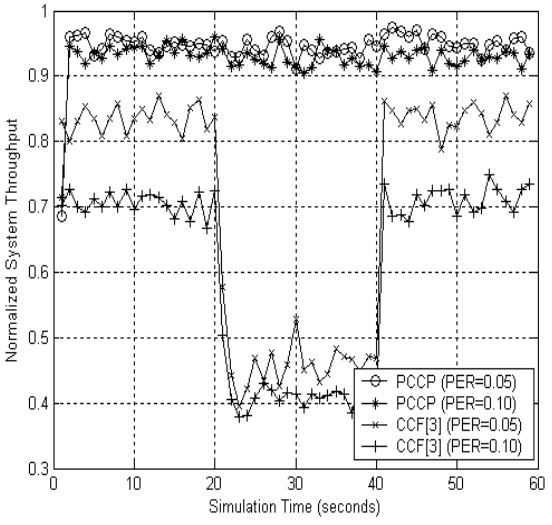
V. Simulation Results

Although PCCP relies on packet-based computation to measure packet service time and packet inter-arrival time and furthermore calculate congestion degree, it could be affordable in either time complexity or in consumed energy due to the two reasons aforementioned in Section I. First of all, the low link capacity enables the unit packet forwarding time long enough to finish such computation within it. In fact, PCCP needs only two multiplication operations and two add operations to measure packet service time according to Eq. (3). For measuring packet inter-arrival time, the time complexity is much lower since PCCP triggers computation according to Eq. (2) only if every $N_p = 50$ packets arrive. Second of all, computation-consumed energy is much smaller than communication-involved. Also as it will be shown in this section, PCCP maintains a short queue length and could avoid packet dropping due to buffer overflow and thereof save more energy furthermore.

$$r_{svc}^{i,j} = GetSvcRate(t_s^{p_{i,j}}, t_a^{p_{i,j}}, GP(p_{i,j}), O(p_{i,j}), r_{svc}^{i,j}, GP(i) * w_{i,j}). \quad (5)$$



(a) PER=0



(b) PER=0.05 and 0.10

Fig. 7. Single path routing-system throughput (PER: Packet Error Rate).

This section presents the simulation results for PCCP. Four scenarios are studied. In simulations, we neglect the details of MAC protocols, but assume they provide even access opportunities for each neighboring node.

A. Single Path Routing

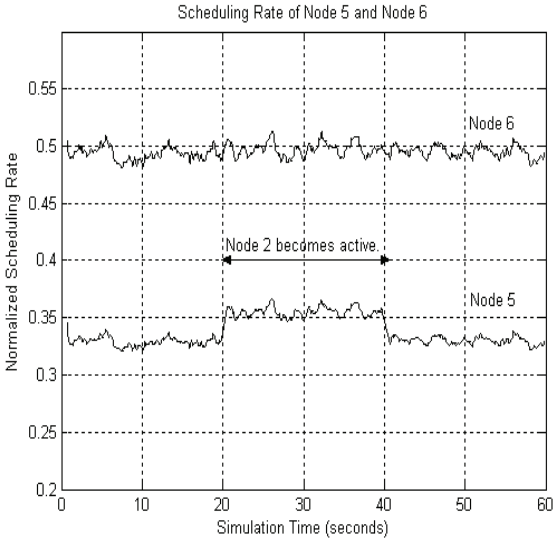
In this case, only single-path routing is assumed. We still assume that each sensor node has the same source traffic priority index and that the sink could obtain the number of packets from each sensor node. A simple tree topology as shown in Fig. 6(a) is assumed where there are 7 sensor nodes. Simulation time is 60 seconds. Sensor node 6 only remains active between [10 sec, 50sec] and generates a small traffic of 1/14 during [20sec, 40 sec], compared to the maximal normalized system throughput.. Other nodes are active throughout and have sufficient traffic throughout the simulation. The system bandwidth is normalized to 1, and therefore each node might receive a throughput of 1/7 in an ideal fair case. We assume that PER caused by bit error rate (BER) is zero.

We compare PCCP with CCF [3], since both share some similarities and CCF is one of the latest and typical congestion control protocol for WSNs. Fig. 7(a) shows the sum of normalized throughput, respectively, for CCF and PCCP. Because CCF cannot effectively allocate the remaining capacity and use work-conservation scheduling algorithm, it has a lower throughput in the interval [20 sec, 40 sec] when node 6 does not have sufficient source traffic. When node 6 generates small traffic during the interval [20 sec, 40 sec], PCCP determines that packet inter-arrival time at the node 6 has increased and congestion degree at the node 6 is smaller than 1. Therefore child nodes of the node 6 (i.e., nodes 3 and 4), increase their source rate. As a result, PCCP maintains high throughput during this time interval as shown in Fig. 7(a).

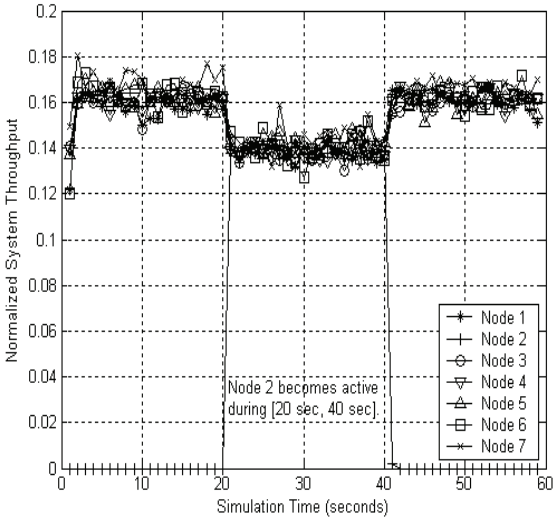
We next study the impact of packet error rate (PER) on throughput of CCF and PCCP. The results are presented in Fig. 7 (b) where the PER are set at 0.05 and 0.1. When PER increases, packet inter-arrival time at each node will increase as well and the link is therefore underutilized. However, CCF only uses packet service time to detect congestion and therefore it cannot detect either underutilized links or nodes. As a result, its throughput is lower with an increase in PER. In PCCP, since it also relies on packet inter-arrival time to detect congestion, it is able to detect underutilized links and nodes resulting from PER. Therefore PCCP results in much higher throughput than CCF as shown in Fig. 7(b).

B. Multi-Path Routing

We use the topology in Fig. 6(b) to study the performance of multi-path routing. In Fig. 6(b), node 2 has two paths to the sink; through node 5 and node 6. It is assumed that node 2 becomes active during the interval [20 sec, 40 sec] and schedules its packets on these paths equally. When node 2 remains inactive (during the interval [0 sec, 20 sec] and [40 sec, 60 sec]) , there are only 6 active nodes, and three of them (nodes 3, 4, and 6) transmit packets through node 6 and the other two transmit packets through node 5; therefore, the normalized scheduling rate of node 6 and node 5 should be around 3/6 and 2/6, respectively, as shown in Fig. 8(a). On the other hand, when node 2 becomes active during the interval [20 sec, 40 sec], half of its packet will be forwarded to node 5 and the other half to node 6. At this time it seems that node 5 has 1.5 child nodes while node 6 has 2.5 child nodes and the total number of active nodes is 7. Therefore, the normalized scheduling rate should be around 2.5/7 for node 5 and 3.5/7 for node 6. Fig. 8(b) shows the normalized throughput of each



(a) Normalized scheduling rate of nodes 5 and 6.



(b) Normalized node throughput.

Fig. 8. Multi-path routing.

node, where each node receives nearly the same throughput. Therefore PCCP supports multi-path routing.

C. Flexible Weighted Fairness

In this case, we use the same topology as in Fig. 6(a), but the nodes will be configured with different source traffic priority index ($SP(i)$) as follows: $SP(6) = 3$, $SP(5) = 2$, and all other nodes with $SP(i) = 1$. According to PCCP, node 5 will receive double the throughput and node 6 will obtain three times the throughput. Nodes 5 and 6 are set to become active during the interval [20 sec, 40 sec]. The result of normalized node throughput is shown in Fig. 9. During the interval [0 sec, 20 sec] or [40 sec, 60 sec], there are only 5 active nodes and PCCP allocated throughput of 0.2 to each active node. When nodes 5 and 6 become active in [20 sec, 40 sec], throughput of node 6 is around 0.3 while node 5 obtains a throughput of 0.2. Other nodes receive a

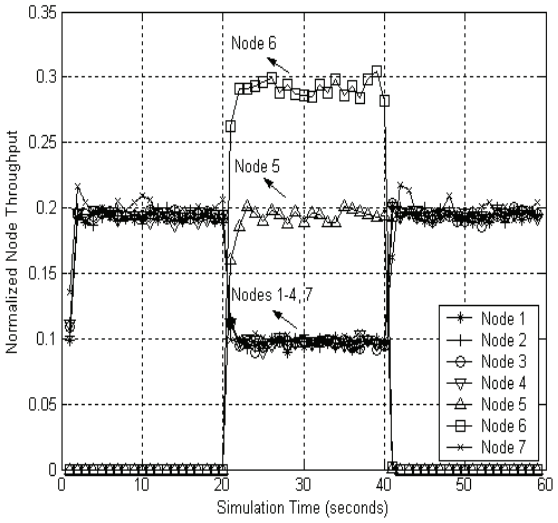


Fig. 9. Flexible weighted fairness.

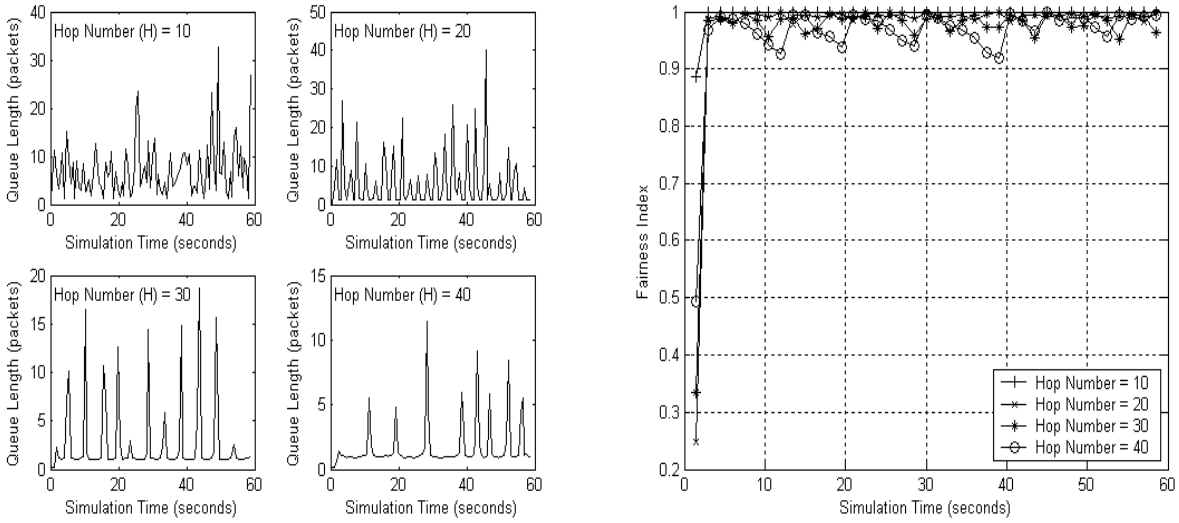
throughput of 0.1. Total throughput is close to 1.0. Therefore PCCP effectively supports flexible weighted fairness through configuration of source traffic priority index of each sensor node. If an applications desires to receive detailed data from sensor nodes in a special area, those nodes are given a higher priority index in order to let them receive higher bandwidth and report more frequently.

D. Impact of Hop Number

We also study the impact number of hops between sensor nodes and the sink on the performance of PCCP. The linear topology in Fig. 6(c) is used. Hop number (H) is varied between 10, 20, 30, and 40. We collect queue length at the node which is closest to the sink since this node has maximal input traffic and maximal queue length as found in simulations. We also calculate fairness index of each node's throughput according to the fairness definition given in [10]. As shown in Fig. 10(a), PCCP results in a short queue length even if $H = 40$ and it can avoid/reduce packet loss due to buffer overflow. In this case as shown in Fig. 10(a), if buffer size of each sensor node is more than 50 packets, there is no buffer overflow. The reduction in packet loss indirectly implies improved energy efficiency since lost packets might have been transmitted within several hops and consumed certain energy. We note that queue length decreases when H increases due to the role of parameter h in the priority-based rate adjustment (see Line 14 in Fig. 5). The fairness index presented in Fig. 10 (b) shows that PCCP provides good fairness (close to 1), although the fairness index declines when H increases.

VI. Conclusion

In this paper, we propose a hop-by-hop upstream congestion control protocol for WSNs, called PCCP. PCCP detects congestion jointly using packet inter-arrival and service times; it introduces node priority index and realizes weighted fairness; it works for both single-path and multi-path routing. It has been demonstrated through simulation that PCCP achieves



(a) Queue length at the node closest to the sink.

(b) Fairness.

Fig. 10. PCCP performance under multi-hop linear topology.

high link utilization and flexible fairness. PCCP leads to small buffer size; therefore it can avoid/reduce packet loss and in turn improves energy-efficiency, and provide lower delay.

We are planning to extensively investigate PCCP performance including energy efficiency through theoretical analysis

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks J*., vol. 38, pp. 393-422, 2002.

[2] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer Networks J*., to appear.

[3] C.-T. Ee and R. Bajcsy, "Congestion control and fairness for many-to-one routing in sensor networks," in *Proc. ACM Sensys*, Nov. 2004.

[4] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating congestion in wireless sensor networks," in *Proc. ACM Sensys*, Nov. 2004.

[5] C.-Y. Wan, S. B. Eisenman, A. T. Campbell, and J. Crowcroft, "Siphon: Overload traffic management using multi-radio virtual sinks in sensor networks," in *Proc. ACM SenSys*, Nov. 2005.

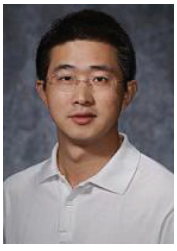
[6] C. Wang, B. Li, K. Sohraby, M. Daneshmand, and Y. Hu, "A Survey of transport protocols for wireless sensor networks," *IEEE Network*, vol. 20, no. 3, pp. 34-40, May/June 2006.

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

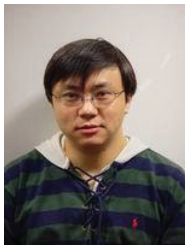
[8] A. Woo and D. C. Culler, "A transmission control scheme for media access in sensor networks," in *Proc. ACM Mobicom*, July 2004.

[9] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," *ACM Mobile Computing Commun. Review*, vol. 1, no. 2, pp. 10-24, 2002.

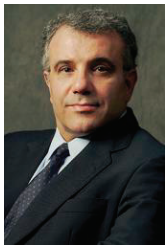
[10] R. Jain, *The Art of Computer Systems Performance Analysis*. New York: John Wiley and Sons, 1991.



Chonggang Wang is a post-doctoral research fellow at the University of Arkansas, Fayetteville. He received his PhD in communication and information system from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China. His current research interests include wireless sensor networks, wireless/mobile networks, and Internet protocols.



Bo Li is a faculty member in the Department of Computer Science, Hong Kong University of Science and Technology. He received his PhD in Electrical and Computer Engineering from the University of Massachusetts at Amherst. His current research interests are multi-hop wireless networks and live media streaming. He is an editor for several IEEE Transactions and ACM journals, and was Co-TPC Chair for IEEE Infocom 2004. He is currently a distinguished lecturer for the IEEE Communications Society.



Kazem Sohraby is a Professor of Electrical Engineering and served as head of the Department of Computer Science and Computer Engineering at the University of Arkansas, Fayetteville. He served as the director of Interdisciplinary Telecommunications Management at Stevens Institute of Technology. He is the founder of the Center for Advanced Computing and Communications Research, Networking Research Laboratory, and a Principal Consultant with industry. Before joining academia, he was with Bell Labs at Lucent and AT&T. He has over 20 pending and granted patents and over 60 publications, books, and book chapters. He currently serves as an IEEE Communications Society Director and served as its president's representative on the Committee on Communications and Information Policy (CCIP). He received his PhD, MS, and BS all in electrical engineering, and his MBA from the Wharton School at the University of Pennsylvania.

Mahmoud Daneshmand is a SR. Technical Consultant and Director of University Collaborations at AT&T Labs Research, and Affiliate Professor of the School of Technology Management and Computer Science at the Stevens Institute of Technology. He has more than 25 years of teaching, research, and management experience in academia and industry including Bell Laboratories, AT&T Labs, University of California at Berkeley, University of Texas at Austin, Tehran University, and New York University. He has a PhD and MA in Statistics from the University of California, Berkeley, and a MS and BS in Mathematics from the University of Tehran.

Yueming Hu is a professor for the College of Information at the South China Agricultural University, Guangzhou, China.