A Fairness-Aware Congestion Control Scheme in Wireless Sensor Networks

Xiaoyan Yin, *Student Member, IEEE*, Xingshe Zhou, *Member, IEEE*, Rongsheng Huang, *Student Member, IEEE*, Yuguang Fang, *Fellow, IEEE*, and Shining Li

***Abstract*—The event-driven nature of wireless sensor networks (WSNs) leads to unpredictable network load. As a result, congestion may occur at sensors that receive more data than they can forward, which causes energy waste, throughput reduction, and packet loss. In this paper, we propose a rate-based fairness-aware congestion control (FACC) protocol, which controls congestion and achieves approximately fair bandwidth allocation for different flows. In FACC, we categorize intermediate relaying sensor nodes into near-source nodes and near-sink nodes. Near-source nodes maintain a per-flow state and allocate an approximately fair rate to each passing flow. On the other hand, near-sink nodes do not need to maintain a per-flow state and use a lightweight probabilistic dropping algorithm based on queue occupancy and hit frequency. Our simulation results and analysis show that FACC provides better performance than previous approaches in terms of throughput, packet loss, energy efficiency, and fairness.**

***Index Terms*—Congestion control, media access control (MAC), quality of service (QoS), wireless sensor networks (WSNs).**

I. Introduction

Wireless sensor networks (WSNs) have been widely applied to habitat monitoring [1], healthcare [2], object tracking [3], battlefield surveillance, etc. They are different from traditional wireless networks in several aspects [4]. 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 [1], [5], the media access control (MAC) layer [6]–[8], and the network layer [9]–[11].

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 bursty traffic that results from the detected events can easily cause congestion in the networks, particularly in high-rate applications. In WSNs, when data converge toward the sink, i.e., the base station, congestion is more likely to happen at sensors that receive more data than they can forward. Therefore, congestion control is a critical issue in WSNs.

In addition, to let the sink successfully receive the data from different sensors (i.e., acoustic, video, and vibration sensors), we need to consider the fairness issue among the source nodes. For example, in the battlefield surveillance application, each sensor continuously measures its vicinity at a rate of several hundred samples per second. When a significant event (a tank enters the monitored field) is detected by acoustic or pressure sensor nodes, every sensor transmits a time series of recorded samples to the base station. To acquire a multidimensional view of the battlefield, all source sensors should transmit data to the base station in a fair fashion. Energy efficiency is also a critical issue in WSNs because of the restricted power supply. Typically, sensor nodes are battery driven and, hence, have to operate on a limited energy budget. Furthermore, battery replacement is impossible in many sensor networks due to the inaccessible or hostile environments.

Congestion control in WSNs remains as a hot topic. Some papers [12]–[15] provide reliable end-to-end data delivery from every sensor to a sink. A few papers [16]–[20] discuss congestion control mechanisms. However, how to ensure fairness among sensors is not well addressed by previous research. In event-to-sink reliable transport (ESRT) [13], by monitoring the congestion-notification bit carried in the packet header, the base station decides a common rate for all sensors so that no packet will be dropped in the network. This approach achieves fairness but is too pessimistic because every sensor must adapt to the worst rate in the most congested area.

In this paper, we propose a new congestion-control scheme that achieves an approximately fair bandwidth allocation. Intuitively, we require that each flow receives a fair share of the available bandwidth according to its generating rate. However, in WSNs, both the available bandwidth and the number of active flows are time varying. Thus, it is very impractical to allocate a fixed rate to each flow. To achieve an approximately fair bandwidth share, we develop a novel mechanism, namely, fairness-aware congestion control (FACC). In FACC, to adjust the sending rate of each flow as early as possible and save the precious resource at the nodes close to the sink, we categorize all intermediate sensor nodes into near-source nodes and near-sink nodes. Near-source nodes maintain a per-flow state and allocate an approximately fair rate to each passing flow by comparing the incoming rate of each flow and the fair bandwidth share. On the other hand, near-sink nodes do not need

Manuscript received April 1, 2009; revised June 5, 2009. First published July 14, 2009; current version published November 11, 2009. The work of X. Yin, X. Zhou, and S. Li was supported in part by the National Key Technology R&D Program of China under Grant 007BAD79B02 and Grant 2007BAD79B03. The work of R. Huang and Y. Fang was supported in part by the U.S. National Science Foundation under Grant CNS-0721744. The review of this paper was coordinated by Dr. G. Cao.

X. Yin, X. Zhou, and S. Li are with the School of Computer Science, Northwestern Polytechnical University, Xi'an 710129, China (e-mail: SCxiaoyanyin@gmail.com; zhouxs@nwpu.edu.cn; dtlsn278@yahoo.com.cn).

R. Huang and Y. Fang are with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32601-6130 USA (e-mail: rshuang@ufl.edu; fang@ece.ufl.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2009.2027022

to maintain a per-flow state and use a lightweight probabilistic dropping algorithm based on queue occupancy and hit frequency. The reason for this classification is that we can delicately design a strategy to assign an appropriate rate to the near-source nodes and explore a simple strategy on the near-sink nodes to save energy and avoid congestion at the same time.

The rest of this paper is organized as follows. Section II surveys the related works. Section III discusses the network model and the target problem. Section IV introduces the proposed FACC scheme. A discussion is presented in Section V. The performance evaluation of FACC is carried out in Section VI. Finally, we conclude this paper in Section VII.

II. Related Works

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

The first group consists of transport protocols that provide end-to-end reliability without congestion control. Reliable multisegment transport (RMST) [12] is an example of these protocols. RMST is a hop-by-hop reliable transport protocol built 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 rate at which data are transmitted by a node must be manually set by a system administrator.

The second group consists of centralized congestion control schemes. ESRT in WSNs [13] allocates transmission rates to sensors such that an application-specific number of sensor readings are received at the base station, which prevent the network from 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. ESRT uses a sophisticated control law based on empirically derived regions of operation and does not attempt to find an efficient and optimal rate allocation. Unlike ESRT, the work of Kim *et al.* [14] uses a simple sink-initiated control protocol to coordinate transmissions with end-to-end selective negative acknowledgments and retransmissions to provide reliability. Paek and Govindan [15] place all the congestion-detection and rate-adaptation functionality in the sinks and use end-to-end explicit loss recovery to achieve reliable transport control.

The third group of protocols consists of distributed congestion control schemes. Fusion [16] uses the queue length to measure the level of congestion and integrates three techniques: hop-by-hop flow control, rate control, and prioritized MAC. With this combination, Fusion achieves higher goodput and better fairness with heavy loads than previous schemes. Congestion detection and avoidance (CODA) in sensor networks [17] is another congestion-mitigation strategy, which uses slightly different mechanisms from Fusion. It senses both channel and buffer occupancies to measure the congestion level. CODA considers two strategies: open-loop backpressure for transient congestion and end-to-end acknowledgment based approach for persistent congestion. Unlike Fusion, CODA does not explicitly focus on per-source fairness. Interference-aware fair rate control [21] is another distributed rate allocation scheme that employs schemes to achieve fair and efficient rate limiting. It uses a tree rooted at each sink to route all data. When congestion occurs, the rates of the flows on the interfering trees are throttled. However, these schemes do not differentiate between intermediate nodes. Furthermore, in a large network that is under congestion, our approach can allocate the exact bandwidth share to each passing flow and make the best of resource.

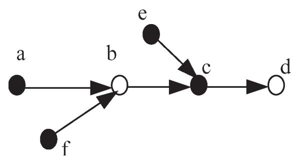


Fig. 1. Simple example of rate allocation.

III. Network Model and Problem Formulation

A. Network Model

A sensor network consists of a large number of sensors and a base station, i.e., a sink. The sink is connected via an external network to a data-collection center. We assume that the transceivers of sensors operate at adjustable transmission rates and that each source sensor generates data at the same original rate.

The sensors share the same wireless medium, and each packet is transmitted as a local broadcast in the neighborhood. Two sensors are neighbors if they are in the transmission range of each other and can directly communicate with each other. We assume a MAC protocol, i.e., IEEE 802.11, which ensures that, among the neighbors in the local broadcast range, only the intended receiver keeps the packet and other neighbors discard the packet. Data packets are sent from source nodes, which can detect the event and generate data packets to the base station. We assume that all data packets have the same size and that sensors are static after initial deployment.

Consider a network of N sensor nodes, where each node is uniquely identified by an integer in the range of [1, N]. Each sensing node always has traffic to send. The traffic originated by source node i is denoted by the ith flow, i.e., f_i. We seek to assign a fair and efficient rate r_i to f_i (or, equivalently, to node i). Specifically, r_i is the transmission rate of flow f_i and does not include the rate at which node i forwards traffic.

The key to congestion control is to make sure that the total rate at which every sensor node transmits data is equal to or less than its available bandwidth. As shown in Fig. 1, the sensor nodes with solid dots are source nodes, i.e., nodes a, c, e, and f, and the rates at which they generate data are r_a, r_c, r_e, and r_f, respectively. For sensor node c, we assume that its available bandwidth is B_c. Obviously, f_a, f_c, f_e, and f_f need to get through node c. The number of active flows, where node c is engaged, should be different from the number of flows going through it. For those flows either originated or terminated at a node, the node counts each as one flow, whereas for those flows only passing through the node, the node counts each as two flows, i.e., one in and one out. Therefore, $2r_a + r_c + 2r_e + 2r_f \leq B_c$. Similarly, $2r_a + 2r_f \leq B_b$. Consider the scenario that an object entering a field triggers a large number of sensors to track its movement. How fast should those sensors send data to the sink? If the generating rate is too low, the system may lose track of the object. If the generating rate is too high, it may cause congestion. Suppose that the sensors initially attempt to generate as much as they can. When congestion occurs at an intermediate sensor x, by our scheme, the generating rates of source nodes are forced to slow down, in accordance with x's available bandwidth. Eventually, the whole network adapts toward the maximum congestion-free throughput. Furthermore, the lower generating rates will alleviate wireless interference and contention.

B. *Target Problem*

The network load greatly depends on the total transmission rate of flows and has a significant impact on packet loss and energy efficiency. The transmission rate of a flow is determined by the channel capacity, the activity of the neighbor sensors and the quality-of-service (QoS) requirements (e.g., lifetime). Thus, both the sending rates and the network load are time varying and are hard to measure. Therefore, estimating the sending rate for each flow and the network load is our first concern.

When the network load exceeds the available bandwidth, congestion occurs. Congestion has dreadful consequences in terms of network utilization, energy efficiency, and packet loss in WSNs. When the offered load goes beyond the critical point of congestion, fewer bits can be sent with the same amount of energy, and the throughput is significantly degraded. Moreover, when a packet is dropped, the energy spent by upstream sensors on the packet is wasted. The farther the packet has traveled, the greater the waste is. We should avoid transmitting these packets bounded to be dropped. Therefore, we adjust the sending rate for each flow as early as possible to avoid congestion. How to effectively adjust the sending rate for each flow is our second concern.

According to the event-driven nature of WSNs, a large number of flows will be produced when events take place. To acquire a multidimensional view of the object region, we must guarantee that each flow transmits its data to the sink in a fair fashion. However, a collection of sensors generating high-rate data can easily overwhelm the network such that the network is unable to operate efficiently. Therefore, how to efficiently and fairly allocate the rate of each flow is our foremost concern.

IV. Fairness Aware Congestion Control Description

To avoid transmissions of unnecessary packets that will otherwise cause a waste of bandwidth and energy, the sending rate of each flow should be adjusted to an appropriate level as early as possible. Thus, it is desirable to adjust the sending rate of each flow at the nodes that are close to source nodes. On the other hand, in WSNs, the nodes that are close to the sink forward more traffic than other intermediate nodes. Thus, their resource and energy are more precious. To adjust the sending rate of each flow as early as possible and save the scarce resource at the nodes close to the sink at the same time, we categorize all intermediate sensor nodes into near-source nodes and near-sink nodes. Near-source nodes maintain a per-flow state and allocate an approximately fair rate to each passing flow by comparing the incoming rate of each flow and the fair bandwidth share. On the other hand, near-sink nodes do not need to maintain a per-flow state and use a lightweight probabilistic dropping algorithm based on queue occupancy and hit frequency.

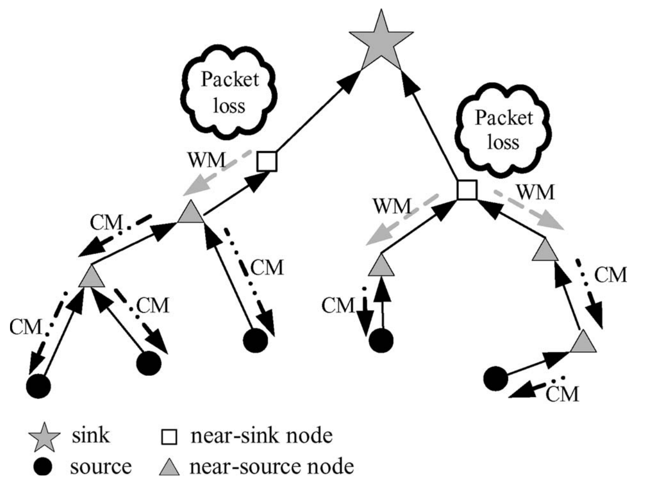


Fig. 2. Logic framework of our scheme.

Our scheme is shown in Fig. 2. First, the near-sink node sends a warning message (WM) back to the near-source nodes once a packet is dropped at this node. Second, the near-source nodes calculate and allocate the approximately fair rate share for each passing flow. Finally, the near-source node sends a control message (CM) to notify the designated source node of the updated sending rate.

A. *Differentiation Between Near-Source Nodes and Near-Sink Nodes*

In this paper, we introduce two concepts, i.e., near-source nodes and near-sink nodes. Just as their names imply, near-source nodes are those nodes close to source nodes, and near-sink nodes are those nodes close to the sink.

We use the optional field as our specific label field for the purpose of differentiation. Every source node sets its label field (e.g., label $= k$) for every packet. This label indicates how far away this packet is from the sensing field. Every forwarding node updates the label field by subtracting one (label $=$ label $-$ 1) when it receives a packet until the label field equals zero. During a fixed interval, every intermediate node calculates the ratio R_p as

$$R_p = \frac{\sharp \text{ of packets (label} > 0)}{\sharp \text{ of total passing packets}}. \tag{1}$$

Intuitively, the larger R_p is, the closer the node is to the source nodes. Therefore, the intermediate node is a near-source node if R_p is no less than a threshold T_p (e.g., 90%). Otherwise, the intermediate node is a near-sink node.

In WSNs, a flow usually traverses a few hops from its source to the sink. The number of hops can be determined by routing protocols and may be dynamic. The intermediate nodes in the path will cooperate with each other to transmit the packet to the sink. According to our scheme, these nodes take on different roles and implement different processes for different purposes. The differentiation between near-source nodes and near-sink nodes depends on applications and QoS requirements. For example, if the complexity on the near-source node and energy efficiency are concerned, a smaller k will be used to provide fewer near-source nodes and more near-sink nodes. On the other hand, if energy is not limited, we can set a larger k to control possible congestion.

B. Near-Source Node Process

In WSNs, both the available bandwidth and the traffic load are time varying. It is very complicated to implement fair resource allocation when considering medium contention and wireless interference. To allocate the available channel resource to each node and to each flow passing through that node, we adopt channel busyness ratio c_b [22] as a metric to characterize the network utilization and congestion status for the IEEE 802.11 MAC. We estimate the available bandwidth resource, the arrival rate of each flow, and the number of active flows for the particular node. As a result, we develop a fairness-aware transmission control mechanism based on the aforementioned metrics.

1) Estimation of the Available Bandwidth: Channel busyness ratio c_b, which is defined as the ratio of time intervals when the channel is busy due to a successful transmission or collision to the total time, provides a good early sign of network congestion. As shown in our previous work [22], the channel utilization for the optimal point is almost the same for different numbers of active nodes and packet sizes, i.e., 95% (with request to send/clear to send). We accordingly set a threshold, which is denoted by th_b, to 92% and leave 3% space for saturation. After choosing th_b, we can estimate the available bandwidth of each node, which is denoted by BW_a, as shown in [22], as follows:

$$BW_a = \begin{cases} 0, & c_b \geq th_b \\ BW(th_b - c_b)\overline{data}/T_s, & c_b < th_b \end{cases} \tag{2}$$

where BW is the transmission rate in bits per second for the DATA packet, $\overline{data}$ is the average payload size measured by the channel occupancy time, and T_s is the average time of a successful transmission at the MAC layer. Therefore, as long as the channel busyness ratio does not exceed the threshold, the node will not operate in the overload status, and the available bandwidth could be used to accommodate more traffic without causing severe MAC contention. Note that the available bandwidth can be shared by all the nodes in the neighborhood, including the observed node.

2) Computation of the Flow Arrival Rate: At each near-source sensor node, we use exponential averaging, as shown in (3), to estimate the rate of a flow. Let t_i^k be the arrival time of the kth packet of flow i and l be the packet length. The estimated rate of flow i, i.e., r_i, as shown in [23], is updated when the kth packet is received as

$$r_i^k = \left(1 - e^{-\frac{T_i^k}{K}}\right)\frac{l}{T_i^k} + e^{-\frac{T_i^k}{K}} r_i^{k-1} \tag{3}$$

where $T_i^k = t_i^k - t_i^{k-1}$ is the interpacket arrival time, and K is a constant. The choice of K is critical. First, a small K can make the system quickly adapt to rate fluctuations, and a large K filters the noise and avoids potential system instability. Second, K should be large enough such that the estimated rate remains reasonably accurate after a packet traverses multiple links. This is because the delay jitter changes the packet interarrival pattern, which may result in an increased difference between the estimated rate and the real rate. To counteract this effect, as a rule of thumb, K should be one order of magnitude larger than the delay jitter experienced by a flow over a time interval of the same size. Finally, K should be no larger than the average duration of a flow. In [23], it has been shown that, by using parameter $e^{-(T_i^k/K)}$, under a wide range of conditions, the estimated rate will asymptotically converge to the real rate.

3) Estimation of the Number of Active Flows: For WSNs, all sensors generate or relay packets. Flows terminate only at the sink. Since the channel is shared by both incoming and outgoing traffic, the number of flows J should be different from the real number of flows. This is because flows passing through the node occupy twice the channel resource compared with flows originating or terminating at the node.

We use one bit in the header of the packet as our special field. It is set only by the source nodes. When the number of remaining packets is larger than $\overline{T_c} \times r$, this bit, which determines whether there are remaining packets in the current flow to be transmitted in the next period, is set to 1; otherwise, it is set to 0, where $\overline{T_c}$ is the control interval, and r is the rate of the corresponding flow originated at the source node. During the fixed period $\overline{T_c}$, each intermediate node counts the number of flows N according to the source address and excludes the packets with special field 0. Thus, J can be estimated as

$$J = \begin{cases} 2N + 1, & \text{a flow is originated} \\ 2N, & \text{otherwise.} \end{cases} \tag{4}$$

4) Transmission Control on Near-Source Nodes:

1) *Inter-node resource allocation*: According to (2), each node could calculate the total available bandwidth for its neighborhood based on the measured channel busyness ratio in $\overline{T_c}$. To determine the available bandwidth of each node, we assign the channel resource ΔS for each node proportionally to its current traffic load S in $\overline{T_c}$. Noticing the linear relationship between BW_a and BW in (2), we have

$$\Delta S = \frac{th_b - c_b}{c_b} \times S. \tag{5}$$

Since both the incoming and outgoing traffic of each node consume the same shared channel resource, S should include the total traffic load (in bytes), i.e., the sum of the total incoming and outgoing traffic. In Fig. 1, for example, there are three flows at node c, and the total traffic for node c is $S = 2r_a + r_c + 2r_e + 2r_f$, where r_a, r_c, r_e, and r_f are the rates at which source nodes a, c, e, and f generate data, respectively. When $c_b < th_b$, ΔS is positive, which means that we can increase the traffic load. In this case, the collision probability is very small, and all the traffic gets through. Thus, the total throughput is approximately equal to the total traffic load. Since the available bandwidth is proportional to $th_b - c_b$, we may increase S by such an amount that, after the increase of ΔS, S is proportional to th_b, which is the optimal channel utilization.

When $c_b \geq th_b$, ΔS is negative, indicating that we should decrease the traffic load. In this case, the linear relationship between the available bandwidth and c_b no longer holds, and the collision probability dramatically increases as the total traffic load increases. In addition, when the node enters saturation, both the collision probability and c_b achieve the maximum and do not change as the traffic increases, although the total throughput decreases. It thus appears that, ideally, we need to aggressively decrease the total traffic load. However, it is difficult to derive a simple relationship between the traffic load and c_b when $c_b \geq th_b$. We hence use the same linear function as in the case of $c_b < th_b$.

2) *Intra-node fair-resource allocation*: After calculating ΔS, i.e., the change in the total traffic, we need to assign it to all flows passing through that node to achieve both efficiency and fairness. Obviously, the total available resource at the node is $\Delta S + S$, i.e., $(th_b - c_b/c_b) \times S + S = (th_b/c_b) \times S$. Thus, the fair rate share $F(t)$ can be computed as follows:

$$F(t) = \frac{th_b}{c_b} \times S/J. \tag{6}$$

3) *Transmission control on near-source nodes*: To avoid congestion, we must ensure that the total transmission traffic is no greater than the instantaneous channel capacity. Thus, the rate of flow i should be updated by $\min(r_i^k, F(t))$. When $r_i^k > F(t)$, the near-source node sends a CM to notify the corresponding source node of the updated rate. The CM contains the flow ID, the node ID, and $F(t)$. When the source node receives the CM, it immediately resets the sending rate.

We expect that no congestion exists after the source node updates its rate. On the other hand, to save energy for near-sink nodes, we take a simple process on near-sink nodes. Specifically, we should take the feedback from near-sink nodes into consideration. When a near-source node receives a WM message from a certain near-sink node, it implies that the rate of the flow is higher than the ideal case. Hence, we update the rate for the particular flow by $\alpha \times \min(r_i^k, F(t))$ (α is a system parameter, e.g., 0.9). When $r_i^k > F(t)$, the near-source node sends a CM to the corresponding source node to notify the updated rate as $\alpha \times F(t)$. When $r_i^k < F(t)$, the near-source node sends a CM to the corresponding source node to notify the updated rate as $\alpha \times r_i^k$.

5) Rate Update Strategy of Source Nodes: Every near-source node computes the fair bandwidth share for each passing flow. According to our scheme, the near-source node will send a CM to the corresponding source node when the rate of a particular flow exceeds the fair bandwidth share or the near-source node receives a WM from a particular near-sink node. Thus, every intermediate node and every source node will receive many CMs, which increases the overhead and may degrade the system performance.

We assume that there are m hops from the source node to the sink for flow f_i. The rate calculated by intermediate nodes for a certain flow is nonincreasing on the path toward the sink. We denote the rate at every intermediate node in terms of hop counts by r_{ik}, where k is the number of hops, and r_{ik} is the available bandwidth share calculated by the kth node along f_i's path. Obviously, we have $r_{i1} \geq r_{i2} \geq \cdots \geq r_{im}$. As long as the source node updates its sending rate as r_{im}, the network will be in good condition. Therefore, other CMs, except the one containing r_{im}, are meaningless. Each relaying node only relays the CM containing the smallest rate and discards the others.

C. Near-Sink-Node Process

1) Stateless Fair Queue Management Mechanism: Every near-sink sensor node is a hotspot with a high probability because of the nature of WSNs. Thus, the resource of near-sink nodes is more valuable. We explore a simple and efficient mechanism to implement transmission control for near-sink nodes.

Like random early detection [24], we preset two thresholds Q_l and Q_h for queue occupancy. When a new packet arrives, the near-sink node computes the hit frequency $h(t)$ by examining whether the packet is from the same flow as one of the M packets randomly selected from the buffer. The hit frequency $h(t)$ is increased by one if one of the packets and the newly arrived packet belong to the same flow. Intuitively, a higher hit frequency $h(t)$ implies that a larger number of packets exist in the buffer for a particular flow. To achieve fairness, we need to give more chances to those flows with lower occupancy. Therefore, The arriving packets that belong to higher occupancy flows have higher dropping probabilities. We calculate the dropping probability p_d of the arriving packet based on the hit frequency $h(t)$ as follows:

$$p_d = \begin{cases} 0, & Q(t) < Q_l \\ h(t)/M, & Q_l \leq Q(t) < Q_h \\ 1, & Q(t) \geq Q_h. \end{cases} \tag{7}$$

2) Hop-by-Hop Backpressure: When packets are dropped and the queue occupancy is between Q_l and Q_h, it indicates that the rate of a particular flow is still higher than that of others and needs to be decreased further. We can simply reduce the sending rate of the corresponding source node. If the queue occupancy exceeds Q_h, the arriving packets will be dropped, which indicates that the traffic is overwhelming, and we need to reduce the rate of all passing flows.

To feed the network status information back to the corresponding source node, the near-sink node will generate a WM containing a flow ID and a node ID, as long as packet loss occurs. The WM as a backpressure signal is eventually transmitted to a certain near-source node, as shown in Fig. 2. Finally, the near-source node will take corresponding aforementioned actions.

3) Fairness of the Stateless QUEUE-Management Mechanism: Consider a queue with N independent Poisson arrivals, each with rate λ_i and independent exponential service times. The queueing discipline is first-in-first-out (FIFO), and the mean service time of each packet is assumed to be $1/\mu$. To simplify the analysis, let us first consider only two arrival processes with arrival rates λ_1 and λ_2. We shall refer to the packets of these flows as type-1 and type-2 packets, respectively.

An arriving packet is either admitted to the queue or dropped, depending on the outcome of a certain comparison and buffer occupancy $Q(t)$, as explained next. When the buffer occupancy is smaller than Q_l, every arriving packet is admitted to the queue. When the buffer occupancy is larger than Q_h, each arriving packet is dropped. When the buffer occupancy is between Q_l and Q_h, each arriving packet is compared with M randomly selected packets from the buffer, where the dropping probability depends on the hit frequency $h(t)$.

Let us first suppose that $\lambda_1 + \lambda_2 < \mu$ so that the queue is stable, $M = 1$, and the selected packet is always the head of the queue. We will later see that, with the dropping strategy in place, the queue will be stable for all values of λ_1, λ_2, and μ. The assumption of stability guarantees that an equilibrium distribution exists for the queue-size process. We denote p_{1,A_1} (respectively, p_{2,A_1}) as the probability that the head of the queue is occupied by a type-1 (respectively, type-2) packet when a packet of type 1 arrives. The well-known PASTA property [25] asserts that $p_{i,A_1} = p_i$, for $i = 1$ and 2, where p_i are the corresponding occupancy probabilities for type i at arbitrary time instants. Since we have assumed that both the arrival processes are Poisson and independent, the same reasoning applies to the probability p_{i,A_2}, and the head of the queue is occupied by a type-i, $i = 1$ and 2, packet when a packet of type 2 arrives.

Given that the service time is independent identically distributed with exponential distribution and the service rate is μ, the head of the queue, whatever type it is, will be served every $1/\mu$ time unit on the average, as long as the queue is not empty. We denote $p_{i,s}$, $i = 1$ and 2, as the probability that the head of the queue is occupied by a type-i packet when the head of the queue is served. Applying the PASTA property again, we find that $p_{i,s}$ also equals p_i for $i = 1$ and 2.

We summarize these observations as follows: $p_{i,A_1} = p_{i,A_2} = p_{i,s} = p_i$, for $i = 1$ and 2, and $p_1 + p_2 = 1$. We now use a rate conservation argument to evaluate p_i. Consider type-1 packets with arrival rate λ_1. A portion (i.e., p_1) of these packets are dropped at arrival. Therefore, the departure rate of type-1 packets from the queue is $\lambda_1(1 - p_1)$. Since the service rate is μ and a portion (i.e., p_1) of the service rate is allocated for type-1 packets, the departure rate of type-1 packets is μp_1. The requisite rate conservation is therefore $\lambda_1(1 - p_1) = \mu p_1$. Solving for p_1 and p_2, we obtain that $p_1 = (\lambda_1/(\mu + \lambda_1))$ and $p_2 = (\lambda_2/(\mu + \lambda_2))$, respectively.

We can see that the occupancy probability for each flow is independent of the arrival rate of other incoming flows. Since μp_i is the departure rate of type-i packets, the goodput of each flow depends only on its own arrival rate and on the service rate μ. When the number of flows N is bigger than two, all the PASTA arguments will also go through. Since these arguments also hold for only one flow, we obtain $p_i = (\lambda_i/(\mu + \lambda_i))$, for any $i \in \mathbf{N}$.

TABLE I
SIMULATION PARAMETERS

Parameter	value
Number of sink nodes	1
Number of Sensor Nodes	50
Number of Flows	10
Transmission Range	250 M
Sensing Range	550 M
Channel Bandwidth	1 Mbps
MAC Protocol	802.11
Routing Protocol	AODV
Simulation Area	$1,000 * 1,000$
Payload Size of DATA Packet	1000 Bytes

An interesting point is that, when $\lambda_i \gg \mu$ and, thus, $p_i \simeq 1$, nearly all the packets of aggressive flows are dropped. At the other extreme when $\lambda_i \ll \mu$, we have $p_i \simeq \lambda_i/\mu$ and $p_i/p_j \simeq \lambda_i/\lambda_j$. We conclude that the ratio of the dropping probabilities is really an indicator for the fairness of the dropping strategy. That is, flows with higher arrival rates incur higher dropping probabilities.

V. DISCUSSION

So far, we have equally treated all source nodes. In reality, different sensors may be assembled with different onboard apparatuses, and their data may have different priorities. The weighted rate allocation captures such differences. Each source sensor x is assigned a weight w_x by operators. Intuitively, when two sensors compete for available bandwidth, $w_x r_x = w_y r_y$ is considered to be fair. Given a rate allocation $r_{\mathfrak{X}} = \{r_x | x \in \mathfrak{X}\}$, the corresponding weighted rate allocation unit is defined as $r_{\text{unit}} = \{(r_x/w_x) | x \in \mathfrak{X}\}$, where $\mathfrak{X}$ is the set of total source nodes. The following modifications should be made to our scheme with priority consideration.

The fair share rate $F(t)$ should be computed at the near-source nodes as follows:

$$F(t) = \frac{th_b/c_b \times S}{\sum w_x}. \tag{8}$$

The rate allocated to the specific source node should be calculated as follows:

$$F_{\text{new}}(t) = F(t) \times w_x. \tag{9}$$

Thus, the rate for flow i should be updated by $\min(r_i^k, F_{\text{new}}(t))$. When $r_i^k > F_{\text{new}}(t)$, the near-source node sends a CM to the corresponding source node to notify the updated rate.

The dropping probability p_d of the arriving packet based on the hit frequency $h(t)$ should be calculated at the near-sink nodes as follows:

$$p_d = \begin{cases} 0, & Q(t) < Q_l \\ h(t)w_x/M\sum w_x, & Q_l \le Q(t) < Q_h \\ 1, & Q(t) \ge Q_h. \end{cases} \tag{10}$$

VI. PERFORMANCE EVALUATION

We evaluate the proposed congestion-control scheme in this section. We use network simulator ns2 version 2.29 to conduct the simulations. The default simulation parameters are described in Table I.

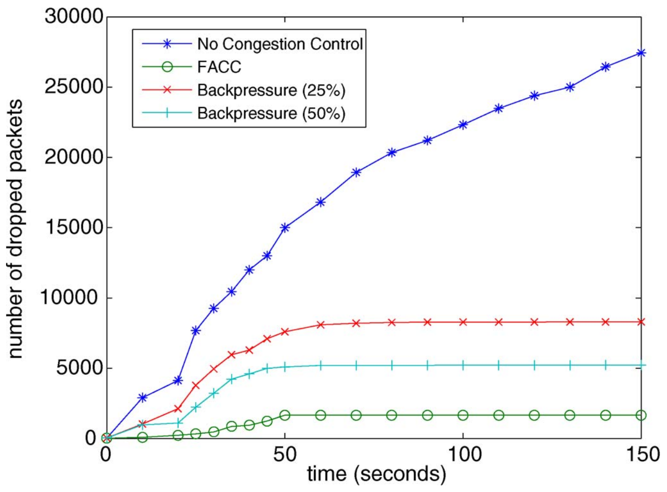


Fig. 3. Accumulated packet loss over time.

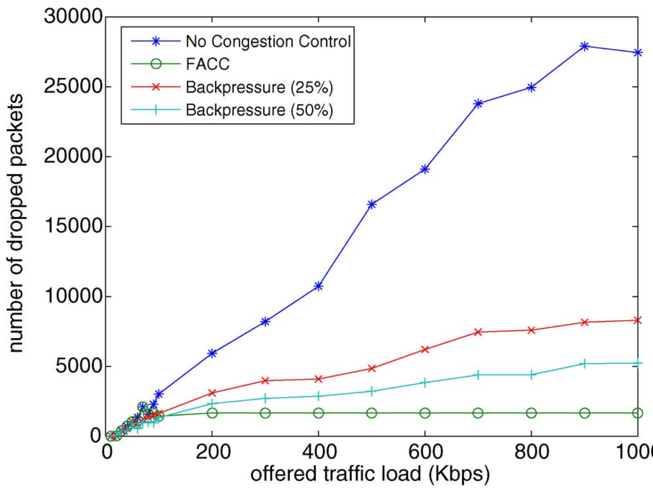


Fig. 4. Final packet loss with respect to offered traffic load.

We implement the backpressure algorithm for comparison purposes in our simulations. As we all know, backpressure is CODA's hop-by-hop congestion-control mechanism [17]. If a sensor x is congested (based on channel utilization and buffer occupancy), it periodically sends backpressure messages to its neighbors, which will reduce their forwarding rates by a certain percentage (25% or 50% in the simulations). If an upstream node is a data source, it reduces the new data generating rate by the same percentage.

In what follows, we first compare no-congestion-control, FACC, and backpressure schemes in terms of packet loss, achievable source rate, and throughput. We then evaluate the property of energy expenditure. Finally, we study the impact of FACC on fairness. For each data point in the figures, we run the simulation on 30 randomly created networks and then take the average.

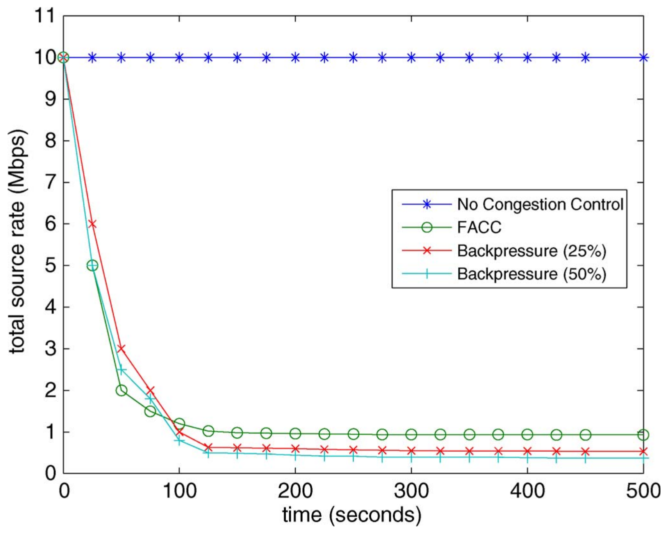


Fig. 5. Total source rate change over time.

A. *Packet-Loss Comparison*

The first set of simulations reveal that our congestion control scheme yields a lower packet dropping rate than other schemes. Fig. 3 shows the number of dropped packets in the networks with respect to time under the offered traffic load of 1000 kb/s for each flow. In Fig. 3, backpressure (50%) and backpressure (25%) refer to the backpressure algorithms with 50% and 25% reduction percentages, respectively, in a sensor's data rate in response to a backpressure message. Both backpressure (50%) and backpressure (25%) drop a significant number of packets during the process of congestion control, whereas fewer packet drops by our congestion control scheme are observed during simulations. Backpressure (50%) drops a smaller number of packets than backpressure (25%) because the former more aggressively reduces data rate and, thus, more quickly mitigates congestion. On the contrary, our congestion-control scheme tries to allocate exact bandwidth share to each flow and, therefore, remarkably lowers the packet drops. Fig. 4 compares the number of dropped packets with respect to the initial rate at which the source nodes generate new data. Intuitively, when the initial source rate is higher, it takes more reduction cycles (more time) to reduce the rate to an appropriate level, resulting in a larger number of packet drops. As we can see, our congestion-control scheme more precisely reacts to congestion and is less sensitive to the initial source rate.

B. *Source-Rate Comparison*

This simulation demonstrates that our congestion control scheme is capable of automatically adapting the sensor's data rate according to the network conditions and achieving better congestion-free rate than other schemes. The total source rate is defined as the total number of data packets generated by all data sources per second. Fig. 5 compares the total source rates of the schemes with respect to time under the offered traffic load of 1000 kb/s for each flow. During the course of congestion control, the total source rate is reduced. At the time instant of around 100 s, congestion control comes into play (except for no congestion control), and the total source rate becomes stable after a short time period. FACC achieves the largest total source rate due to its capability of allocating the exact bandwidth share to each passing flow. On the other hand, the total source rates of backpressure (50%) and backpressure (25%) are smaller than

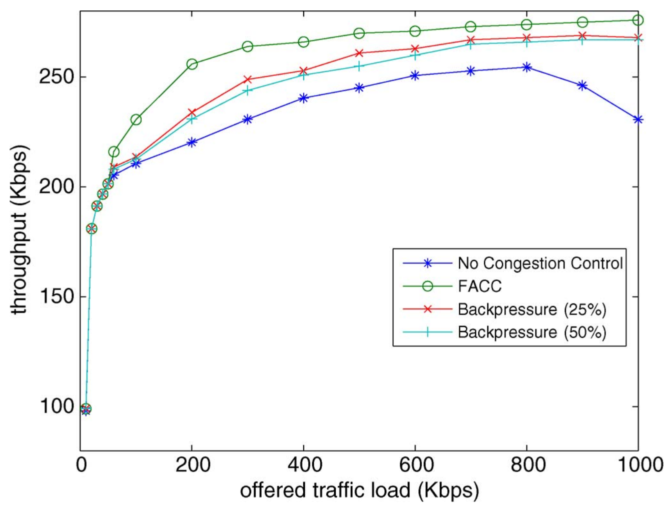


Fig. 6. Throughput with respect to offered traffic load.

our scheme, with the latter slightly better. Combining with the results in Fig. 3, we observe a tradeoff between the number of dropped packets and the total source rate. By more aggressively decreasing the rate in response to congestion, backpressure (50%) has a lower number of dropped packets but a smaller source rate than Backpressure(25%).

C. Throughput Comparison

The following simulation confirms that our congestion-control scheme achieves higher throughput than other schemes. Fig. 6 compares throughput with respect to the offered traffic load at which the sources generate new data. As aforementioned, when the offered traffic load is higher, the probability of congestion becomes higher, but the throughput will not proportionally increase. Fig. 6 shows that our congestion control scheme has higher throughput than the backpressure algorithms. This is because backpressure is hard to adapt to an appropriate level, while our scheme assigns the exact available bandwidth share to each flow and, thus, efficiently utilizes the available bandwidth.

D. Energy Expenditure

The average energy expenditure is defined as the total number of transmissions in the network divided by the number of packets successfully delivered to the sinks. One transmission moves a packet one hop closer to the sink. Fig. 7 illustrates how the average energy expenditure changes with respect to different initial source rates. Fig. 8 depicts how the average energy expenditure changes over time, with the initial rate of 1000 kb/s for each flow. We again find that our congestion-control scheme is more energy efficient than the backpressure scheme because of more efficient bandwidth utilization.

E. Fairness Comparison

For a random topology in Fig. 9, which consists of 50 sensor nodes, one sink, and 10 flows (the solid circles are the source

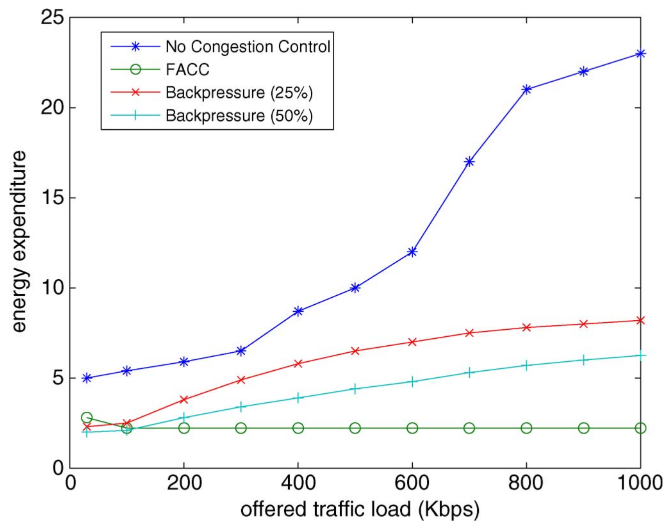


Fig. 7. Energy expenditure per packet with respect to offered traffic load.

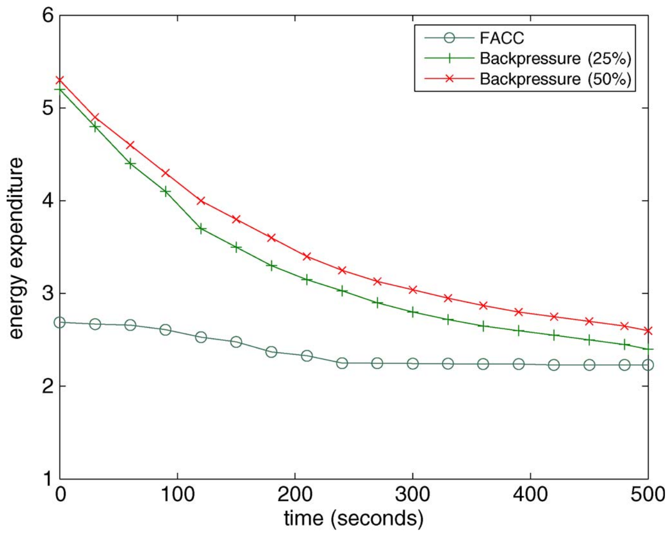


Fig. 8. Energy expenditure per packet over time.

nodes that generate data and form a flow), we explore the fairness property in term of per-flow throughput.

From Fig. 10, we observe that our scheme can acquire approximately fair bandwidth share, with each flow generating data at 1000 kb/s. By contrast, no congestion control completely fails to guarantee fairness for the flows. In particular, flow 2 takes the smallest share (zero), and flow 6 takes the largest share in terms of throughput for two reasons. The first is that different nodes suffer from different interference because of hidden terminals and exposed terminals. The second is that no congestion control favors shorter flows, particularly one- or two-hop flows, and penalizes longer ones. Flow 6 may be such a two-hop flow and achieves the maximum throughput as if there were no other flows in the neighborhood. As a victim, flow 2 encounters severe contention from surrounding nodes and gains no throughput at all. With our scheme, we observe that the starving problem for long flows is resolved. Our scheme controls exactly every flow's incoming traffic, hence reducing interference and improving the channel utilization. Moreover,

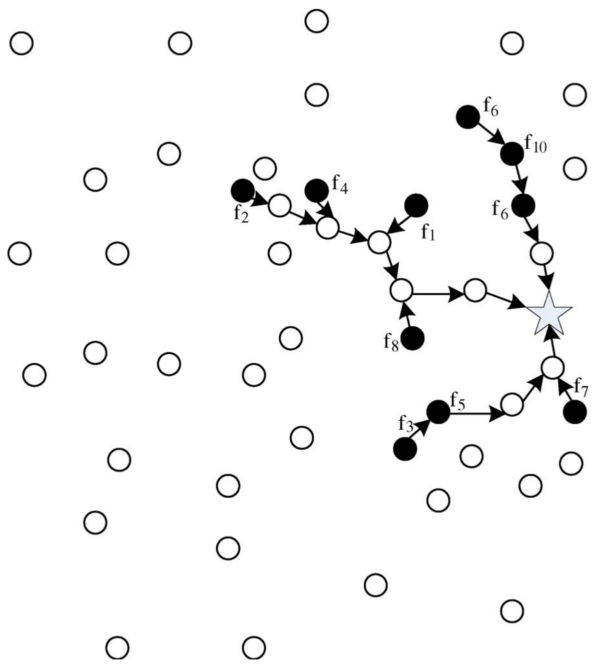


Fig. 9. Fifty-node random topology.

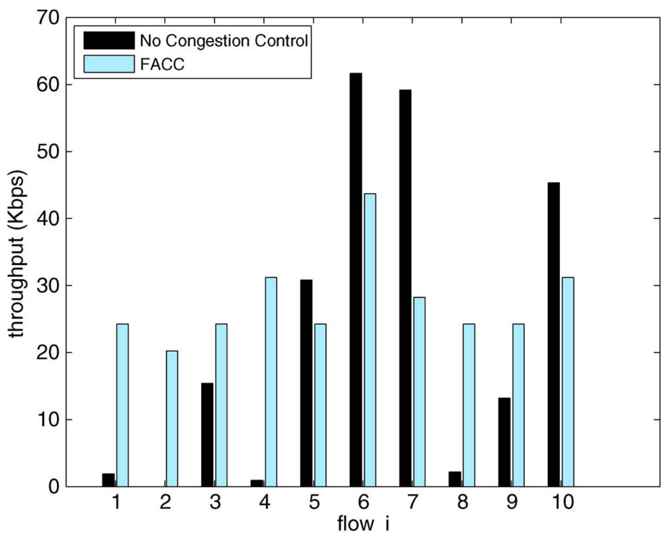


Fig. 10. Throughput with respect to different flows.

by monitoring the channel busyness ratio and each flow's traffic, our scheme can accurately calculate the available bandwidth of the channel and fairly assign it to each flow. Therefore, our scheme can achieve better fairness.

VII. Conclusion

In this paper, we have proposed a scheme for congestion control in WSNs. This paper has provided a new mechanism to control congestion and achieved reasonably fair bandwidth allocation in resource-constrained WSNs. We have shown by simulations that FACC has better performance than the backpressure schemes in terms of packet loss, energy efficiency, channel utilization, and fairness. Particularly for throughput, FACC can achieve up to 20% improvement compared with no congestion control. For packet loss, FACC becomes stable after a certain time in spite of the increase in offered traffic load, while no congestion control results in linearly increasing packet loss as the offered traffic load increases or as time elapses.

References

[1] 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.

[2] 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.

[3] B. Brooks, P. Ramanathan, and A. Sayeed, "Distributed target classification and tracking in sensor networks," *Proc. IEEE*, vol. 91, no. 8, pp. 1163–1171, Aug. 2003.

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

[5] R. Cramer, M. Win, and R. Scholtz, "Impulse radio multipath characteristic and diversity reception," in *Proc. IEEE ICC*, Atlanta, GA, Jun. 1998, pp. 1650–1654.

[6] 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.

[7] A. Woo and D. Culler, "A transmission control scheme for media access in sensor networks," in *Proc. ACM MobiCom*, Rome, Italy, Jul. 2001, pp. 221–235.

[8] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie, "Protocol for self-organization of a wireless sensor network," *IEEE Pers. Commun.*, vol. 7, no. 5, pp. 16–27, Oct. 2000.

[9] 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.

[10] 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.

[11] J. Kulik, W. Rabiner, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *Proc. ACM MobiCom*, Seattle, WA, Aug. 1999, pp. 174–185.

[12] 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.

[13] Y. Sankarasubramaniam, O. Akan, and I. F. Akyildiz, "ESRT: Event-to-sink reliable transport in wireless sensor networks," in *Proc. 4th ACM Int. Symp. Mobile ad hoc Netw. Comput. MobiHoc*, Annapolis, MD, Jun. 2003, pp. 177–188.

[14] S. Kim, R. Fonseca, P. Dutta, and A. Tavakoli, "Flush: A reliable bulk transport protocol for multihop wireless networks," in *Proc. 5th ACM Conf. Embedded Netw. Sensys*, Sydney, Australia, Nov. 2007, pp. 351–365.

[15] J. Paek and R. Govindan, "RCRT: Rate-controlled reliable transport for wireless sensor networks," in *Proc. 5th ACM Conf. Embedded Netw. Sensys*, Sydney, Australia, Nov. 2007, pp. 305–319.

[16] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating congestion in wireless sensor networks," in *Proc. 2nd ACM Conf. Embedded Netw. Sensys*, Baltimore, MD, Nov. 2004, pp. 134–147.

[17] C. Wan, S. Eisenman, and A. Campbell, "CODA: Congestion detection and avoidance in sensor networks," in *Proc. 1st ACM Conf. Embedded Netw. Sensys*, Los Angeles, CA, Nov. 2003, pp. 266–279.

[18] S. Chen, Y. Fang, and Y. Xia, "Lexicographic maxmin fairness for data collection in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 7, pp. 762–776, Jul. 2007.

[19] S. Chen and N. Yang, "Congestion avoidance based on lightweight buffer management in sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 9, pp. 934–946, Sep. 2006.

[20] R. Kumar, R. Crepaldi, H. Rowaihy, A. Harris, and G. Cao, "Mitigating performance degradation in congested sensor networks," *IEEE Trans. Mobile Comput.*, vol. 7, no. 6, pp. 682–697, Jun. 2008.

[21] S. Rangwala, R. Gummadi, R. Govindan, and K. Psounis, "Interference-aware fair rate control in wireless sensor networks," in *Proc. SIGCOMM*, Pisa, Italy, Sep. 2006, pp. 63–74.

[22] H. Zhai, X. Chen, and Y. Fang, "Improving transport layer performance in multihop ad hoc networks by exploiting MAC layer information," *IEEE Trans. Wireless Commun.*, vol. 6, no. 5, pp. 1692–1701, May 2007.

[23] I. Stoica, S. Shenker, and H. Zhang, "Core-stateless fair queueing: Achieving approximately fair bandwidth allocation in high speed networks," Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CS-98-136, Jun. 1998.

[24] S. Floyd and V. Jacobson, "Random early detection for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

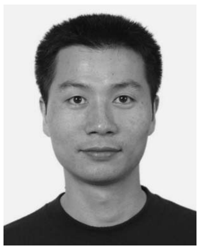
[25] G. Bolch, S. Greiner, H. Meer, and K. Trivedi, *Queueing Networks and Markov Chains*. Hoboken, NJ: Wiley, 2006.

Xiaoyan Yin (S'09) received the B.S. and M.S. degrees in computer science from Harbin University of Science and Technology, Harbin, China, in 2001 and 2004, respectively. She is currently working toward the Ph.D. degree with the School of Computer Science, Northwestern Polytechnical University, Xi'an, China.

She was an Assistant with the Department of Computer Science, Xi'an Technological University from April 2004 to September 2005. She was a Visiting Student with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, from December 2007 to May 2009. Her research interests are in the area of congestion control, optimization, and quality-of-service support for wireless sensor networks.

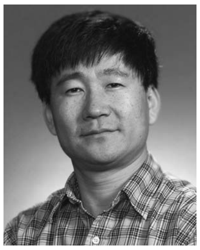
Xingshe Zhou (M'04) received the B.S. and M.S. degrees in computer science from Northwestern Polytechnical University, Xi'an, China.

He is currently a Professor with the School of Computer Science, Northwestern Polytechnical University. His research interests include distributed computing, embedded computing, and sensor networks.



Rongsheng Huang (S'07) received the B.S. and M.S. degrees in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 1996 and 1999, respectively. He is currently working toward the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Florida, Gainesville.

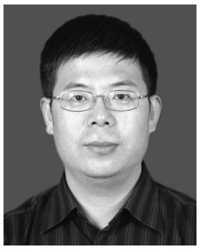
From 1999 to 2001, he was with Huawei Technologies Company Ltd., Shenzhen, China, as an R&D Engineer on General Packet Radio Service and third-generation (3G) projects. From 2002 to 2005, he was with UTStarcom Research Center, Shenzhen, as a Senior Engineer and the Team Leader of a 3G project. His research interests are in the area of media access control, protocol, and architecture for wireless networks.



Yuguang Fang (S'92–M'97–SM'99–F'08) received the Ph.D. degree in systems engineering from Case Western Reserve University, Cleveland, OH, in January 1994 and the Ph.D. degree in electrical engineering from Boston University, Boston, MA, in May 1997.

He was an Assistant Professor with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, from July 1998 to May 2000. In May 2000, he joined the Department of Electrical and Computer Engineering, University of Florida, Gainesville, as an Assistant Professor. He got an early promotion to Associate Professor with tenure in August 2003 and to Professor in August 2005. He will hold the University of Florida Research Foundation Professorship from 2006 to 2009. He is the author of more than 250 papers in refereed professional journals and conference proceedings. He has served on many editorial boards of technical journals, including *ACM Wireless Networks*.

Dr. Fang received the National Science Foundation Faculty Early Career Award in 2001 and the Office of Naval Research Young Investigator Award in 2002. He has served on the editorial boards of the IEEE TRANSACTIONS ON COMMUNICATIONS, the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and the IEEE TRANSACTIONS ON MOBILE COMPUTING. He is currently serving as the Editor-in-Chief for IEEE WIRELESS COMMUNICATIONS.



Shining Li received the B.S. and M.S. degrees in computer science from Northwestern Polytechnical University, Xi'an, China, in 1989 and 1992, respectively, and the Ph.D. degree in computer science from Xi'an Jiaotong University, Xi'an, in 2005.

He is currently a Professor of the School of Computer Science, Northwestern Polytechnical University. His research interests include mobile computing, wireless sensor networks, and embedded computing.