A Virtual-Queue based Backpressure Scheduling Algorithm for Heterogeneous Multi-Hop Wireless Networks

Zhenzhen Jiao[1,2], Baoxian Zhang[1] and Jun Zheng [3]
[1] Research Center of Ubiquitous Sensor Networks, University of Chinese Academy of Sciences
Beijing 100049, China
[e-mail: jiaozhenzhen11b@mails.ucas.ac.cn, bxzhang@ucas.ac.cn]
[2] China Academy of Information and Communications Technology, Beijing 100033, China
[3] National Mobile Communications Research Lab, Southeast University
Nanjing, Jiangsu 210096, China
[e-mail: junzheng@seu.edu.cn]
*Corresponding author: Jun Zheng

Received January 7, 2014; revised July 28, 2015; revised September 23, 2015; accepted October 7, 2015; published December 31, 2015

Abstract

Backpressure based scheduling has been considered as a promising technique for improving the throughput of a wide range of communication networks. However, this scheduling technique has not been well studied for heterogeneous wireless networks. In this paper, we propose a virtual-queue based backpressure scheduling (VQB) algorithm for heterogeneous multi-hop wireless networks. The VQB algorithm introduces a simple virtual queue for each flow at a node for backpressure scheduling, whose length depends on the cache size of the node. When calculating flow weights and making scheduling decisions, the length of a virtual queue is used instead of the length of a real queue. We theoretically prove that VQB is throughput-optimal. Simulation results show that the VQB algorithm significantly outperforms a classical backpressure scheduling algorithm in heterogeneous multi-hop wireless networks in terms of the packet delivery ratio, packet delivery time, and average sum of the queue lengths of all nodes per timeslot.

Keywords: heterogeneous network; backpressure scheduling; packet delivery ratio; virtual queue

This work was supported by National Natural Science Foundation of China under Grant Nos. 61501125, 61531006, and 61372105, and the Six Talent Peaks Project in Jiangsu Province under Grant No. 2013-DZXX-010.

http://dx.doi.org/10.3837/tiis.2015.12.007 **ISSN : 1976-7277**

1. Introduction

Backpressure based routing and scheduling has been considered as a promising technique for improving the throughput of a wide range of communication networks because of its throughput-optimality property. The first backpressure scheduling algorithm was proposed in [1]. It is a cross-layer queue-length based scheduling algorithm for packet transmission, which has been proven to be throughput optimal. Moreover, it can provide network-utility-optimal scheduling guarantee when combined with rate control (e.g., [2]-[4]) and high packet delivery performance when used for route selection (e.g., [5]-[8]). Although this algorithm has many advantages, e.g., throughput optimality, achievable adaptive resource allocation, support for stateless and flexible load-aware routing and scheduling, there are still some problems that prevent it from being widely deployed in the real world, including its centralized control mode, high computational complexity, and poor delay performance [9]. Recently, considerable research work has been conducted to address these problems and provide different solutions to efficient deployment of backpressure based scheduling [7]-[14]. However, the use of a backpressure based scheduling algorithm in heterogeneous multi-hop wireless network environments is still an issue not well addressed, which motivated us to conduct this work.

Heterogeneous multi-hop wireless networks have become increasingly common with the popularity of wireless devices with routing functions. As such heterogeneous devices usually have different capabilities in terms of cache, battery, and computation, many classical backpressure based algorithms designed for homogeneous networks cannot perform efficiently in such networks. For a classical backpressure algorithm, time is slotted and each node in the network maintains a queue structure for each flow passing through it. In each timeslot, a set of non-interference links whose overall link weight can contribute to a global maximum sum are activated to transmit packets. Here, the weight of a link is the largest flow weight on the link, where the weight of a flow, also called flow weight, is the differential of the flow's queue backlogs between the two endpoints of the link. A classical backpressure algorithm only concerns about the *absolute* (per-flow) queue backlog differential between neighbor nodes. However, when it is used in a heterogeneous network where different nodes have different cache sizes, it may cause unexpected problems. For example, a packet scheduled to be sent to a node may be directly dropped if the receiving node does not have enough buffer space, which would reduce the packet delivery ratio and thus the throughput of the network.

To address the above problem, we propose a new efficient backpressure scheduling algorithm for heterogeneous multi-hop wireless networks. Specifically, we introduce a simple virtual queue for each flow at a node for backpressure scheduling, whose size depends on the cache size of the node. To the best of our knowledge, this is the first work that considers the impact of the cache size of a heterogeneous node on backpressure scheduling. Simulation results show that the proposed backpressure scheduling algorithm outperforms a classical backpressure scheduling algorithm significantly in heterogeneous multi-hop wireless networks in terms of the packet delivery ratio, packet delivery time, and average sum of the queue lengths of all nodes per timeslot.

The rest of the paper is organized as follows. Section II reviews related work. Section III describes the system model under study. Section IV first gives a brief review to classical backpressure algorithms and then presents the proposed backpressure scheduling algorithm.

Section V shows simulation results to evaluate the performance of the proposed algorithm. Section VI concludes this paper.

2. Related Work

The backpressure algorithm in [1] was proposed by Tassiulas and Ephremides. Although this algorithm has many advantages, e.g., throughput optimality, achievable adaptive resource allocation, support for stateless and flexible load-aware routing and scheduling, there are still some problems that prevent it from being widely deployed in the real world, including its centralized control mode, high computational complexity, and poor delay performance. To address these issues, considerable research work has been conducted recently in this field.

In [12], Laufer et al. designed a practical cross-layer backpressure structure to reach the full capacity of a multi-hop wireless network. Under that structure, a mesh network is transformed to a wireless switch, where packet routing and scheduling decisions are made by a backpressure scheduler. Moreover, they also implemented and evaluated the performance of backpressure scheduling over a Time Division Multiple Access (TDMA) based Media Access Control (MAC) protocol in such networks. In [10, 18], backpressure scheduling was implemented on a contention-based MAC protocol by adjusting the contention behavior according to contending nodes' queue differentials. Moreover, backpressure-based rate control at the transport layer was also considered in [10]. Horizon in [11] is a backpressure-based system designed for enhancing Transmission Control Protocol (TCP)'s performance, which balances the traffic load over multiple disjoint paths by using the information on queue states. Backpressure Collection Protocol (BCP) in [7] is a backpressure based collection protocol for wireless sensor networks (WSNs). In BCP, backpressure-based routing is carried out by taking into account queue differentials and link rates. In [8], Diff-Max considers the issue of separating routing and scheduling in the context of backpressure in order to make practical implementation easier. In [8], the authors focused on how to implement backpressure based scheduling in CSMA based wireless networks in a distributed manner. In [14], Sridharan et al. conducted empirical study on the performance of backpressure scheduling in CSMA wireless networks.

To our knowledge, the only previous work that considered the issue of backpressure based scheduling in heterogeneous networks is [4]. In this work, Neely et al. considered the issue of optimal rate control for a network with both wireless and wired components and time varying channels. They developed a dynamic strategy to handle all possible traffic to make optimally fair decisions on which data to serve when input traffic exceeds the network capacity. However, they did not consider the impact of different characteristics of heterogeneous nodes (e.g., buffer size) on the performance of backpressure scheduling in a heterogeneous network.

In this paper, we consider the issue of performing backpressure scheduling in a heterogeneous network where different nodes have different cache sizes. We introduce a new metric taking into account nodes' cache sizes for improving the backpressure scheduling performance in the heterogeneous network. In this context, there has been some previous work reported in the literature [6][7][15][16]. In [6] and [7], backpressure routing protocols for Delay Tolerant Networks (DTNs) and WSNs consider not only queue differential but also other factors (e.g., routing loop punishment, packet reception ratio, node's social selfishness behavior, etc.) to make a proper routing decision. In [15], Ji et al. introduced a delay metric and accordingly extended classical backpressure to a delay-aware modified version. In [16], backpressure is combined with inter-flow network coding and the available coding

opportunity is a new concern introduced in making scheduling decisions to reduce the packet delivery latency.

3. System Model

We consider a multi-hop wireless network and model it as a directed graph $G = (V, E)$, where $V(G)$ and $E(G)$ represent the set of nodes and the set of links in the network, respectively. The weight associated with a link $(u, v) \in E(G)$ is not necessarily the same as its counterpart for link (v, u). Time is divided into timeslots, which are denoted by t. We use the "unidirectional equal power" interference model described in [17]. When a node is transmitting, all the links whose receivers are located within the communication range of the node will be interfered. A data packet can be transmitted on link (u, v) if the link does not interfere with the transmissions currently ongoing on other links. A non-interference link set in timeslot t is denoted by $I(t)$, which is a subset of $E(G)$ and contains the links with no interference to each other. The set of all possible non-interference link sets in timeslot t is denoted by $S(t)$. Thus, $I(t) \in S(t)$.

We use F to denote the set of flows in the network. The data packets of a flow arrive in and leave the network via the source node and destination node of the flow, respectively. The arriving rate of a flow $f \in F$ is denoted by λ_f and is measured in number of packets per timeslot. We only consider unidirectional flows in which the packets of a flow always travel from the source of the flow to its destination.

Next, we introduce the queue structure at a node in $V(G)$. Each node in $V(G)$ maintains a forwarding queue for each flow traversing it. The data packets received at a node are buffered in the forwarding queue of the node before the packets are forwarded. The forwarding queue backlog of flow f at node n in timeslot t is denoted by $Q_n^f(t)$. Assume that data packets enter the network via the source node of each flow in each timeslot. Thus, the dynamics of the queue of flow f at its source node can be described as

$$Q^f_{source(f)}(t+1) = Q^f_{source(f)}(t) + \lambda_f. \tag{1}$$

Further, the data packets of a flow leave the network after arriving at the flow's destination. As a result, the queue of a flow at the flow's destination node can always be set to being empty, i.e.,

$$Q^f_{dest(f)}(\infty) = 0. \tag{2}$$

To handle the diversity of nodes' different cache sizes in a heterogeneous network, we introduce a virtual queue structure into backpressure scheduling. That is, in addition to a real queue, each node maintains one virtual queue for each flow passing through it. We use $\widetilde{Q}_n^f(t)$ to denote the virtual queue backlog of flow f at node n in timeslot t. Furthermore, the dynamics of these virtual queues at the source and destination of the flow are the same as their counterparts for real queues[1], i.e.,

[1] This is made under an assumption that flow sources and destinations will not act as intermediate nodes for other flows, an assumption used in a lot of work in this area.

$$\tilde{Q}^{f}_{source(f)}(t+1) = \tilde{Q}^{f}_{source(f)}(t) + \lambda_f, \tag{3}$$

$$\tilde{Q}^{f}_{dest(f)}(\infty) = 0. \tag{4}$$

4. Virtual-Queue Based Backpressure Algorithm

In this section, we first describe the scheduling process of the classical backpressure algorithm (denoted as BP hereafter) proposed in [1], and then present the proposed virtual-queue based backpressure (VQB) algorithm.

4.1 Classical Backpressure Algorithm

In the BP algorithm [1], for each link $(m, n) \in E(G)$, its link weight in timeslot t is the maximum backlog differential of all flows currently passing through the link, where ties are broken arbitrarily, i.e.,

$$W_{mn}(t) = max_{f:(m,n)\in E(G)}[Q^{f}_{m}(t) - Q^{f}_{n}(t)]. \tag{5}$$

The objective of the classical backpressure algorithm is to determine a schedule $\pi(t)$ which solves the following optimization:

$$\begin{aligned} &\text{Maximize: } \textstyle\sum_{(m,n)} W_{mn}(t) R_{mn}(I(t)), \\ &\text{Subject to: } I(t) \in S(t), \end{aligned} \tag{6}$$

where $R_{mn}(I(t))$ is the link rate of $(m, n) \in I(t)$, which is measured in number of packets per timeslot. As a result, the schedule $\pi(t)$ indicates the number of packets needed to be transmitted on links belonging to $I(t)$, while the packets are from the flows whose differential backlogs achieve the maximum in (5).

4.2 Motivation

In the BP algorithm, queue backlog differential is used as the only parameter for making scheduling decisions. However, this parameter may not be appropriate for a heterogeneous network where different nodes can have different capabilities. For example, let us consider the heterogeneous network shown in **Fig. 1**. There are two flows traversing the network from source S to destination D. Both node A and node B are intermediate nodes, and the buffer size of node B is much smaller than that of node A. Here, we assume that node B can buffer at most six packets. **Fig. 1** shows the queue backlog of each node in timeslot t. Accordingly, since there are six packets already stored in the buffer of node B in timeslot t, there is no more buffer space available in node B. In this case, if a new packet arrives at node B, it will be directly dropped. However, if a classical backpressure scheduling algorithm is performed, the backlog differential between the queue of flow f_2 at node S and that at node B, i.e., $Q_S^{f_2}(t) - Q_B^{f_2}(t)$, achieves the maximum in one-hop neighborhood. In this case, six packets will be scheduled to be transmitted from node S to node B consequently in timeslot t+1, which will be then dropped due to insufficient buffer space in node B.

The inadequacy of classical backpressure scheduling in a heterogeneous network motivated us to design a new backpressure scheduling algorithm, which can adapt to the heterogeneous network environment while still maintaining the advantages of the backpressure scheduling algorithm.

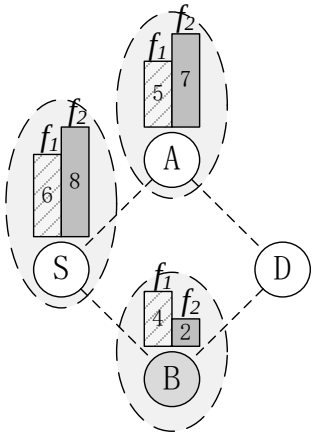


Fig. 1. An example illustrating the inefficiency of classical backpressure scheduling in heterogeneous networks. The digits in the slash-filled and gray filled columns in this figure represent the queue backlogs of flows f_1 and f_2, respectively.

4.3 VQB algorithm

1) Main idea

The main idea of the VQB algorithm is to allow each node to establish a virtual queue for each flow passing through the node, and use the backlog differential of the virtual queues instead of real queues to calculate flow weights. In the VQB algorithm, the varying rate of a virtual queue, also called virtual rate, plays a vital role. If the virtual rate equals to the rate of its corresponding real queue, the VQB algorithm becomes the BP algorithm. After assigning different virtual rates to different nodes, the VQB algorithm can then implement transmission scheduling using virtual queue length based backpressure, which treats heterogeneous nodes differently. To understand this, let us consider a flow-specific real queue of node n with VQB. When a packet of flow f arrives at node n in timeslot t+1, the dynamics of the real queue of the node can be described as

$$Q_n^f(t+1) = Q_n^f(t) + 1. \tag{7}$$

Meanwhile, the dynamics of the corresponding virtual queue is

$$\tilde{Q}_n^f(t+1) = \tilde{Q}_n^f(t) + R_{VQ}(n), \tag{8}$$

where $R_{VQ}(n)$ denotes the virtual rate of node n, which is measured in number of packets. According to Eq. (8), $R_{VQ}(\cdot)$ is obviously flow irrelevant. The virtual rates of different nodes in a heterogeneous network are usually different. For example, a node with a smaller cache could be assigned a larger virtual rate, which makes the virtual queues in this node grow more

rapidly than the nodes with a bigger cache and thus reduce the possibility for this node to be selected as a forwarder by virtual-queue based backpressure scheduling (recall that VQB calculates flow weights using backlog differentials of virtual queues).

In this context, there is some existing work using virtual queues reported in the literature [7, 9, 19-21]. For example, in [7], a virtual queue structure (called floating queue therein) is used for recording the number of packets which have been dropped and the dropped packets are still used in the calculation of queue differentials for packet scheduling. In [19], the purpose of using a virtual queue (called shadow queue) is to extend the network utility maximization framework to support multicast flows. Similar virtual queue structures are also used in [9, 20, 21]. Different from existing work, the purpose of introducing virtual queue and also virtual rate in this paper is to support efficient backpressure scheduling in a heterogeneous network while making it be aware of different nodes' cache status. Moreover, a key difference between this work and existing work is how queue dynamics is managed upon packet arrival and departure. In this work, the virtual queue size of a node is a function of the node's cache size. Instead, in existing work, queue dynamics is cache size irrelevant.

2) Major procedure

Firstly, each node maintains a virtual queue for each flow that passes through the node. A virtual queue does not need to store any real packet (or packet-related information). Thus, it can simply be implemented as a counter. When a packet of flow f arrives at (resp. leaves) node n in timeslot t, the corresponding virtual queue will change as shown in Eq. (9) (resp. Eq. (10)):

$$\tilde{Q}_n^f(t) = \tilde{Q}_n^f(t-1) + R_{VQ}(n), \tag{9}$$

$$\tilde{Q}_n^f(t) = \tilde{Q}_n^f(t-1) - R_{VQ}(n). \tag{10}$$

There are different ways to determine the values of $R_{VQ}(n)$, $n \in V(G)$. The way used in the VQB algorithm is a simple but effective method. Consider **Fig. 1** again and assume that nodes A and B play only as intermediate nodes between source S and destination D, and node A has a larger cache size than node B. In this case, node A is supposed to undertake more transmissions than node B as it has more available buffer space. For this purpose, $R_{VQ}(B)$ is supposed to be larger than $R_{VQ}(A)$ and it is intuitive that the larger the value of C_A/C_B is, the larger the value of $R_{VQ}(B)/R_{VQ}(A)$ should be, where C_A and C_B denote the cache size of node A and that of node B, respectively. Following this intuition, the virtual rate $R_{VQ}(n)$ for a node $n \in V(G)$ is defined as the ratio between C_{max} and C_n, i.e., $R_{VQ}(n) = C_{max}/C_n$, where C_{max} denotes the maximum cache size of an intermediate node in the network and C_n denotes the cache size of node n. The idea behind this virtual rate assigning method is that if an intermediate node has the maximal buffer space among all intermediate nodes, it will be assigned the lowest virtual rate, i.e., the same to that of its real queue, which may increase its possibility of being selected as the next hop when making forwarding decisions.

There is no modification to the management of real queues in the VQB algorithm. In addition to Eq. (10), the backlog of a virtual queue will immediately turn to zero when its corresponding real queue becomes empty.

In timeslot t, for each link $(m, n) \in E(G)$, its link weight equals to the maximum virtual-queue-backlog differential of all the flows passing through the link (ties broken arbitrarily):

$$\widetilde{W}_{mn}(t) = max_{f:(m,n)\in E(G)}[\tilde{Q}_m^f(t) - \tilde{Q}_n^f(t)]. \quad (11)$$

The VQB algorithm then selects a schedule $\pi(t)$ to solve the following optimization:

$$\begin{aligned} &\text{Maximize: } \textstyle\sum_{(m,n)} \widetilde{W}_{mn}(t) R_{mn}(I(t)), \\ &\text{Subject to: } I(t) \in S(t), \end{aligned} \quad (12)$$

where $R_{mn}(I(t))$ denotes the actual link rate of link (m, n) that belongs to $I(t)$. Thus, selecting such a schedule $\pi(t)$ will indicate the (potential) number of packets needed to be transmitted on the links in I(t), while the packets are from the flows whose virtual-queue-backlog differential achieves the maximum in Eq. (11). A real queue belonging to a flow which is allowed to be activated will send $\widetilde{W}_{mn}(t)$ packets or until it becomes empty in the case when the number of packets stored in a real queue is smaller than $\widetilde{W}_{mn}(t)$. When all the packets in a real queue have been sent out, the backlog of its corresponding virtual queue will also drop to zero.

Let us revisit the example shown in **Fig. 1** and see how the VQB algorithm works for this example network. As node *B*'s available buffer space is much smaller than that of node *A*, it is assigned a larger virtual rate $R_{VQ}(B) = C_A/C_B$ consequently based on the virtual rate assigning method described earlier. Meanwhile, we have $R_{VQ}(A) = C_A/C_A = 1$. Thus, node *B* may have a higher virtual queue backlog and smaller possibility of being selected as the next hop than node *A*.

It can be seen that the scheduling operations in VQB are mainly under the basic idea of backpressure, i.e., packets are always pushed from the nodes with a higher virtual queue backlog to the lower ones with a lower backlog. However, there may exist some cases wherein packets are scheduled to be transmitted from the nodes with a lower (real) queue backlog to the ones with a higher backlog. Let us again consider the example in **Fig. 1** but with some slight modifications. Assume node *D* is also an intermediate node on the way to a packet's destination and has a much larger cache space than node *B*. As a result, $R_{VQ}(B)$ will be larger than $R_{VQ}(D)$. In this case, if a larger $R_{VQ}(B)$ is assigned, the virtual queue length of node *B* may be higher than its counterpart of node *D* (i.e., $\tilde{Q}_B^f \gg \tilde{Q}_D^f$), which may cause a transmission from node *B* to node *D*. Meanwhile, it is still possible that the statuses of their real queues are the opposite to those of their virtual queues, i.e., the real queue backlogs of node *D* is higher than those of node *B* (i.e., $Q_D^f > Q_B^f$). As a result, the packets will have the probability to be transmitted from a node with a lower (real) queue backlog to one with a higher queue backlog.

Intuitively, a scheduling decision made by the VQB algorithm in such cases may bring a negative impact on the performance of backpressure scheduling. However, it should be noticed that although the real queue length of node *B* is smaller than that of node *D*, a larger virtual queue length of node *B* actually means that node *B* has a higher cache occupancy ratio than node *D*. In such a heterogeneous network, this is a hint that a quick relief to the *pressure* on the inadequacy of cache of node *B* is necessary. Thus, the scheduling decision made by the VQB algorithm is indeed consistent with the design goal to provide a cache-availability-aware backpressure algorithm for heterogeneous networks. Next, we will give a theoretical analysis to prove that the VQB algorithm can always stabilize a when the traffic arrival rate is within the capacity region of the network.

3) Performance of algorithm

First, we prove that the VQB algorithm is throughput-optimal, i.e., VQB can always stabilize a network when traffic arrival rate is within the capacity region of the network.

It is known that a network is strongly stable if for all $n \in V(G)$ and $f \in F$, the following

inequality holds [1]:

$$lim\ sup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}[Q_n^f(\tau)] < \infty. \tag{13}$$

Furthermore, in VQB, the length of a virtual queue is used for making scheduling decisions. Although a virtual queue is simply implemented by using a counter, its length still cannot increase without bounds because it would affect the corresponding scheduling decisions. Thus, to prove VQB's stability, the virtual queue at a network node $n \in V(G)$ also needs to satisfy the following inequality:

$$lim\ sup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}[\tilde{Q}_n^f(\tau)] < \infty. \tag{14}$$

For deducting our following analysis, we first have the following lemma.

Lemma 1 There always exists a finite factor θ_n (≥ 1) that can be used for expressing the relationship between the lengths of actual and virtual queues of node *n* as follows,

$$\tilde{Q}_n^f(t) \leq \theta_n Q_n^f(t). \tag{15}$$

Proof According to Eq. (7) and Eq. (8), it is known that once a packet arrives at an intermediate node *n*, its real queue will increase one, i.e., $Q_n^f(t+1) = Q_n^f(t) + 1$. Meanwhile, the dynamics of the corresponding virtual queue is $\tilde{Q}_n^f(t+1) = \tilde{Q}_n^f(t) + R_{VQ}(n)$. Here, recall $R_{VQ}(n)$ denotes the virtual rate of node *n*, which is defined as $R_{VQ}(n) = C_{max}/C_n$ in this paper, where C_{max} denotes the maximum cache size of an intermediate node in the network and C_n denotes the cache size of node *n*. Thus, $R_{VQ}(n) \geq 1$ must hold in this paper. As $R_{VQ}(n)$ is determined in system's initial phase, we have $Q_n^f(0) = \tilde{Q}_n^f(0) = 0$. During the time period t' (≥ 1), φ packets have arrived at Q_n^f, from which meanwhile ω packets are scheduled to be sent out according to Eq. (11). Then we have $Q_n^f(t') = Q_n^f(0) + \varphi - \omega$ and $\tilde{Q}_n^f(t') = \tilde{Q}_n^f(0) + (\varphi - \omega)R_{VQ}(n)$. Furthermore, each time Q_n^f drops to zero, the length of the corresponding virtual queue will also be set to zero at this time. Thus, we can always have $\tilde{Q}_n^f(t') = R_{VQ}(n)Q_n^f(t')$ when *n* is an intermediate node. Moreover, for source and destination nodes in network, $\tilde{Q}^f = Q^f$ always holds in our settings. As a result, we conclude that for each node *n* in network, there always exists a finite positive factor θ_n can make $\tilde{Q}_n^f(t) \leq \theta_n Q_n^f(t)$ hold. ■

Lemma 1 provides us an intuitive clue that once the real or virtual queues have been proved to be stable as shown in Eq. (13) and Eq. (14), the stability of VQB can be proved. Here, we still use the Lyapunov stability criterion for proving the stability of both the real and virtual queues, by using the Lyapunov function $L(Q) = \sum_n \sum_f \theta_n (Q_n^f)^2$, which is also used in [22].

Lemma 2 Given finite constants $\mathbb{B}$ and ϵ, lengths of both real and virtual queues in network are always bounded as follows.

$$lim\ sup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\left[\sum_n \sum_f Q_n^f(t)\right] \leq lim\ sup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\left[\sum_n \sum_f \tilde{Q}_n^f(t)\right] < \frac{\mathbb{B}\theta_{max}}{\epsilon}.$$

Proof Consider that the actual queue dynamics at each timeslot satisfy the following inequality:

$$Q_n^f(t+1) \leq max[Q_n^f(t) - \sum_b \mu_{nb}^f(t), 0] + \sum_a \mu_{an}^f(t) + A_n^f(t), \tag{16}$$

In Eq. (16), A_n^f is an indicator for external traffic. When node n is the source of flow f, we have $A_n^f(t) = \lambda_f$; Otherwise, $A_n^f(t) = 0$. Further, $\mu_{ab}^f(t)$ represents the transmission rate of flow f on link (a, b) at timeslot t. Based on the fact that $(\max(U\text{-}b,0)+A)^2 \leq U^2+A^2+b^2+2U(A\text{-}b)$, we have:

$$\left[Q_n^f(t+1)\right]^2 \leq \left[Q_n^f(t)\right]^2 + \left[\sum_a \mu_{an}^f(t) + A_n^f(t)\right]^2 + \left[\sum_b \mu_{nb}^f(t)\right]^2 + 2Q_n^f(t)[\sum_a \mu_{an}^f(t) + A_n^f(t) - \sum_b \mu_{nb}^f(t)]. \quad (17)$$

Then we multiply both sides of (17) by θ_n and then sum all (n, f) pairs, i.e.,

$$\sum_n \sum_f \theta_n\left[Q_n^f(t+1)\right]^2 \leq \sum_n \sum_f \theta_n\left[Q_n^f(t)\right]^2 + \sum_n \sum_f \theta_n\left[\sum_a \mu_{an}^f(t) + A_n^f(t)\right]^2 + \sum_n \sum_f \theta_n\left[\sum_b \mu_{nb}^f(t)\right]^2 - 2\sum_n \sum_f \theta_n Q_n^f(t)[\sum_b \mu_{nb}^f(t) - \sum_a \mu_{an}^f(t) - A_n^f(t)]. \quad (18)$$

Thus we can rewrite the above inequality as follows.

$$\Delta L(Q) \leq \mathbb{B}\theta_{max} - 2\epsilon \sum_n \sum_f \theta_n Q_n^f(t), \quad (19)$$

where $\mathbb{B} \triangleq \sum_n[(\mu_{nb}^{max})^2 + (\mu_{an}^{max} + A_n^{max})^2]$, $\epsilon = \sum_b \mu_{nb}^f(t) - \sum_a \mu_{an}^f(t) - A_n^f(t)$, and θ_{max}, μ_{nb}^{max}, μ_{an}^{max}, and A_n^{max} represent the achievable maximum value of θ_n, $\sum_b \mu_{nb}^f(t)$, $\sum_a \mu_{an}^f(t)$, and A_n^f, respectively. The inequality in (19) exactly obeys the form provided in Lyapunov stability lemma in [21]. Thus we can have

$$lim\ sup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}[\sum_{nf} \theta_n Q_n^f(t)] < \frac{2\mathbb{B}\mathbb{N}\theta_{max}}{\epsilon}. \quad (20)$$

Based on Lemma 1, we can conclude the stability of both the real and virtual queues and thus the stability of VQB, i.e., its throughput optimality. ■

Next, we discuss the overhead of VQB. The only additional overhead introduced by VQB is that used for maintaining per-flow virtual queues at each node. A virtual queue is actually a counter for its corresponding flow-based real queue. Once the backlogs of real queues change, their corresponding virtual queues also need to be updated. The total number of such counters at a node is bounded by the number of active flows in the network, i.e., $O(|F|)$, where $|F|$ represents the number of flows in the network.

5. Performance Evaluation

In this section, we evaluate the performance of the VQB algorithm through simulation results. As discussed in the preceding section, for a node $n \in V(G)$, $R_{VQ}(n) = C_{max}/C_n$. In the simulations, we assume that $R_{VQ}(n) = \lfloor C_{max}/C_n \rfloor$, where $\lfloor x \rfloor$ returns the largest integer smaller than or equal to x. We consider such a setting because $R_{VQ}(n)$ is measured in number of packets, which is always an integer.

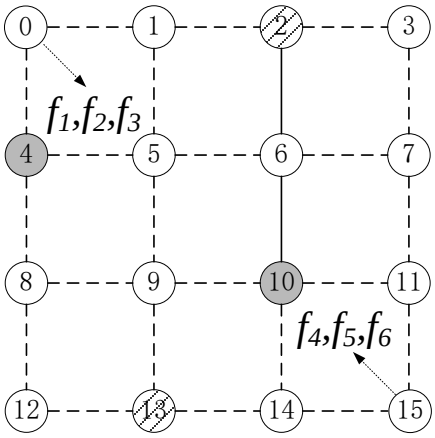


Fig. 2. A grid topology.

In the simulations, we compare the VQB algorithm with the BP algorithm. Both of the two algorithms were implemented using a commonly used practical method for schedulable link set generation, which is known as Greedy Maximal Scheduling (GMS) or Greedy Maximal Matching (GMM) [23]. Specifically, for each timeslot, when generating a non-interference schedule, a link (m, n) with the global maximum weight $W_{mn}(t)$ will be added to the schedule at the beginning. By removing all the links interfering with (m, n) and repeating such greedy selection until there is no link left, a schedulable link set for the timeslot is obtained. Such a GMS-based method usually generates suboptimal schedules but only has $O(|E|\log|E|)$ computational time, where $|E|$ is the number of links in the network.

In the simulations, we consider two types of network topologies. The first topology is a diamond network shown in **Fig. 1**. Two flows traverse the network from left to right with arrival rates $\lambda_{f_1} = 3$ and $\lambda_{f_2} = 2$, respectively, while a third flow travels from right to left with $\lambda_{f_3} = 4$. All these rates are measured in number of packets per timeslot. The cache sizes of the only two intermediate nodes in the network are assigned as follows: $C_A = 150$, $C_B = 100$ in case 1, which means nodes A and B can store at most 150 and 100 packets, respectively; $C_A = 150$, $C_B = 70$ in case 2; $C_A = 150$, $C_B = 50$ in case 3. More details can be found in **Table 1**. The second topology is a 4×4 grid network shown in **Fig. 2**. In this network, node 0 and node 15 are the source and destination nodes, and all other nodes are intermediate nodes, which are further divided into three categories. The cache sizes of the slash-filled nodes (i.e., nodes 2 and 13) are assigned to be larger than the gray nodes (i.e., nodes 4 and 10) but smaller than those of others, where gray, slash-filled, and all remaining nodes' cache sizes are assigned 40, 80, 150, respectively, in case 1; 30, 60, 150 in case 2; and 20, 60, 150 in case 3. More details can be found in **Table 2**. There are six flows traveling through the network. Three of them travel from left top to right bottom with arrival rates $\lambda_{f_1} = 3$, $\lambda_{f_2} = 2$, and $\lambda_{f_3} = 4$, while the other three travel from right bottom to left top with $\lambda_{f_4} = 3$, $\lambda_{f_5} = 2$, and $\lambda_{f_6} = 5$.

Table 1. Cache sizes of nodes for different cases in topology in **Fig. 1.**

Cases / Node ID	Case 1	Case 2	Case 3
A	150	100	150
B	100	70	50

Table 2. Cache sizes of nodes for different cases in the topology in **Fig. 2**.

Cases / Node ID	Case 1	Case 2	Case 3
1	150	150	150
2	80	60	60
3	150	150	150
4	40	30	20
5	150	150	150
6	150	150	150
7	150	150	150
8	150	150	150
9	150	150	150
10	40	30	20
11	150	150	150
12	150	150	150
13	80	60	60
14	150	150	150

In the simulations, we assume that each flow has 200 packets in total, which are called original packets. Once an original packet is dropped by an intermediate node, it will be retransmitted by the source node until it is correctly received by the destination.

For performance evaluation, we use the following three metrics: packet delivery ratio, packet delivery time, and average sum of the queue lengths of all nodes per timeslot. The packet delivery ratio is defined as the ratio of the number of packets received by the destinations to the number of packets sent by the sources. The packet delivery time is defined as the number of timeslots needed for delivering all the original packets. The average sum of the queue lengths of all nodes per timeslot is the average value of the sum of all nodes' forwarding queue lengths during the simulation period.

Fig. 3 compares the BP algorithm with the VQB algorithm for the diamond network topology shown in **Fig. 1**. **Fig. 3(a)** shows the packet delivery ratio with the BP algorithm and the VQB algorithm, respectively. It is seen that the packet delivery ratio with VQB is much larger than that with BP except in case 1, which we will explain later. This is because VQB takes different node cache sizes into account and thus can reduce the number of dropped packets caused by cache shortage.

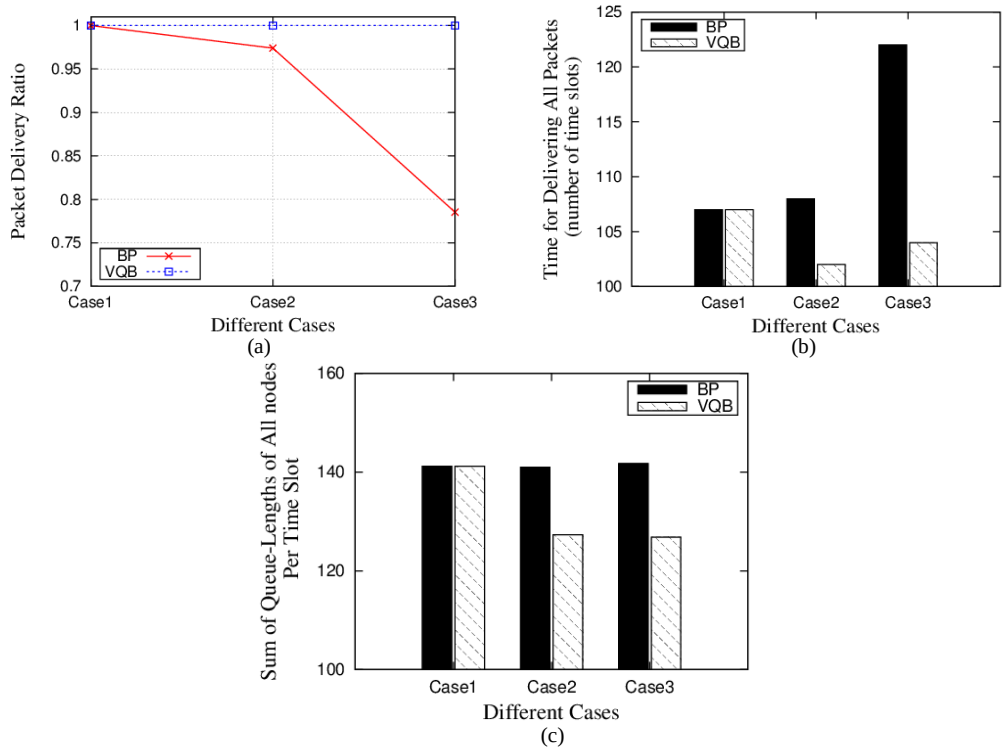


Fig. 3. Performance comparison for the diamond network in **Fig. 1**.

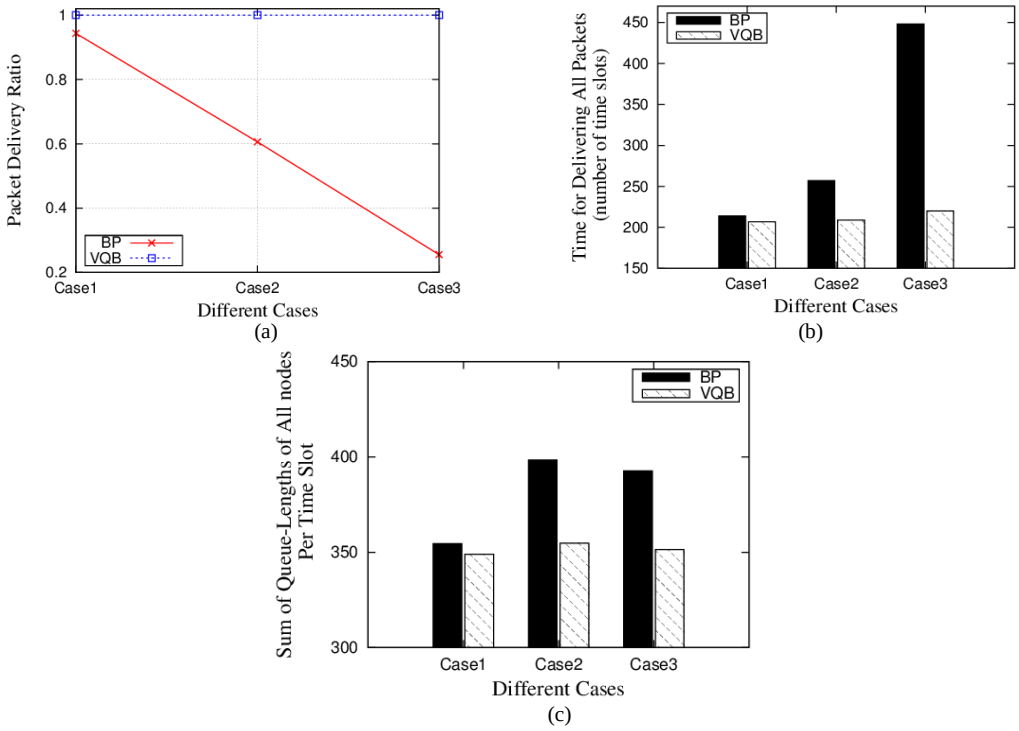


Fig. 4. Performance comparison for the grid network in **Fig. 2**.

Fig. 3(b) shows the overall packet delivery time with the BP algorithm and the VQB

algorithm, respectively. It is seen that VQB needs less time for delivering all packets than the BP algorithm, which means that VQB can achieve a higher throughput than BP. The reason is that the VQB algorithm can reduce the number of dropped packets and thus the number of retransmissions than the BP algorithm.

Fig. 3(c) compares the average sum of the queue lengths of all nodes per timeslot with the BP algorithm and the VQB algorithm, respectively. It is seen that VQB has a smaller average queue length than BP.

Finally, it is important to indicate that when all nodes in the diamond network have similar cache sizes (e.g., in case 1 where $R_{VQ}(B) = 1$), VQB performs similarly to BP, which can be seen in **Figs. 3(a), (b)**, and (c) (for case 1). This is reasonable because we should not violate the throughput-optimal scheduling by backpressure when it is suitable for a network wherein all nodes have sufficient buffer spaces.

Fig. 4 compares the BP algorithm and the VQB algorithm in terms of the packet delivery ratio, the overall packet delivery time, and average queue lengths for the grid network shown in **Fig. 2**. It is seen that VQB can achieve a much larger packet delivery ratio, less overall packet delivery time, and smaller queue lengths than BP in all cases, which justifies that VQB outperforms BP in a heterogeneous network.

5. Conclusion

In this paper, we have proposed a VQB algorithm for heterogeneous multi-hop wireless networks. By introducing a per-flow virtual queue structure, VQB can effectively treat heterogeneous nodes differently based on their cache sizes during backpressure scheduling for packet transmissions. The simulation results show that VQB significantly outperforms the classical BP algorithm in a heterogeneous network in terms of the packet delivery ratio, packet delivery time, and average sum of the queue lengths of all nodes per timeslot.

Our future work will concentrate on designing more efficient virtual-queue based backpressure algorithms and related issues. Meanwhile, we are also interested in the performance of a virtual-queue based backpressure algorithm in a practical environment and will continue our exploration along this direction.

References

[1] L. Tassiulas and A. Ephremides, “Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks,” *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, December, 1992. Article (CrossRef Link)

[2] X. Lin and N. Shroff, “Joint rate control and scheduling in multihop wireless networks,” in *Proc. of 43th IEEE Conf. on Decision and Control*, pp. 1484–1489, December 14-17, 2004. Article (CrossRef Link)

[3] Eryilmaz and R. Srikant, “Fair resource allocation in wireless networks using queue-length based scheduling and congestion control,” in *Proc. of 24th Annual Joint Conference of the IEEE Computer and Communications Societie*, pp. 1794–1803, March 13-17, 2005. Article (CrossRef Link)

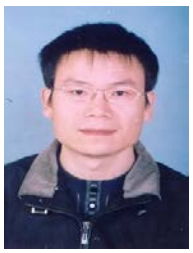
[4] M. Neely, E.Modiano, and C. Li, “Fairness and optimal stochastic control for heterogeneous networks,” in *Proc. of 24th Annual Joint Conference of the IEEE Computer and Communications Societie*, pp. 1723–1734, March 13-17, 2005. Article (CrossRef Link)

[5] M. J. Neely and R. Urgaonkar, “Optimal backpressure routing for wireless networks with multireceiver diversity,” *Ad Hoc Networks*, vol. 7, no. 5, pp. 862–881, July, 2009. Article (CrossRef Link)

[6] Dvir and A. V. Vasilakos, "Backpressure-based routing protocol for DTNs," in *Proc. of the annual conference of the ACM Special Interest Group on Data Communication*, pp. 405–406, August 30-September 3, 2010. Article (CrossRef Link)
[7] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali, "Routing without routes: The backpressure collection protocol," in *Proc. of the 9th IEEE/ACM International Conference on Information Processing in Sensor Networks*, pp. 279–290, April 12-16, 2010. Article (CrossRef Link)
[8] H. Seferoglu and E. Modiano, "Diff-Max: Separation of routing and scheduling in backpressure-based wireless networks," in *Proc. of the 32nd IEEE International Conference on Computer Communications*, pp. 1555-1563, April 14-19, 2013. Article (CrossRef Link)
[9] L. Bui, R. Srikant, and A. L. Stolyar, "A novel architecture for reduction of delay and queueing structure complexity in the backpressure algorithm," *IEEE/ACM Transactions on Networking*, vol. 19, no. 6, pp. 1597–1609, December, 2011. Article (CrossRef Link)
[10] Warrier, S. Janakiraman, S. Ha, and I. Rhee, "DiffQ: Practical differential backlog congestion control for wireless networks," in *Proc. of the 28th Conference on Computer Communications*, pp. 262–270, April 19-25, 2009. Article (CrossRef Link)
[11] Radunovic, C. Gkantsidis, D. Gunawardena, and P. Key, "Horizon: Balancing TCP over multiple paths in wireless mesh network," in *Proc. of the 14th ACM Annual International Conference on Mobile Computing and Networking*, pp. 247–258, September 14-19, 2008. Article (CrossRef Link)
[12] R. Laufer, T. Salonidis, H. Lundgren, and P. L. Guyadec, "XPRESS: A cross-layer backpressure architecture for wireless multi-hop networks," in *Proc. of the 17th ACM Annual International Conference on Mobile Computing and Networking*, pp. 49–60, September 19-23, 2011. Article (CrossRef Link)
[13] M. Alresaini, M. Sathiamoorthy, B. Krishnamachari, and M. J. Neely, "Bacpressure with adaptive redundancy (BWAR)," in *Proc. of the 31st Annual IEEE International Conference on Computer Communications*, pp. 2300–2308, March 25-30, 2012. Article (CrossRef Link)
[14] Sridharan, S. Moeller, B. Krishnamachari, and M. Hsieh, "Implementing backpressure-based rate control in wireless networks," in *Proc. of Information Theory and Applications Workshop*, pp. 341–345, Feburary 8-13, 2009. Article (CrossRef Link)
[15] Ji, C. Joo, and N. B. Shroff, "Delay-Based Back-Pressure Scheduling in Multi-Hop Wireless Networks," in *Proc. of the 30th IEEE International Conference on Computer Communications*, pp. 2579–2587, April 10-15, 2011. Article (CrossRef Link)
[16] Z. Jiao, Z. Yao, B. Zhang, and C. Li, "NBP: An efficient network-coding based backpressure algorithm," in *Proc. of IEEE International Conference on Communications*, pp. 1625–1629, June 9-13, 2013. Article (CrossRef Link)
[17] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar, "Throughput guarantees in maximal scheduling in wireless networks," *IEEE Transactions on Information theory*, vol. 54, no. 2, pp. 572–594, Feburary, 2008. Article (CrossRef Link)
[18] U. Akyol, M. Andrews, P. Gupta, J. Hobby, I. Saniee, and A. Stolyar, "Joint scheduling and congestion control in mobile ad-hoc networks," in *Proc. of the 27th IEEE International Conference on Computer Communications*, pp. 619–627, April 13-18, 2008. Article (CrossRef Link)
[19] L. Bui, R. Srikant, and A. Stolyar, "Optimal resource allocation for multicast sessions in multihop wireless networks," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1872, pp. 2059–2074, June, 2008. Article (CrossRef Link)
[20] Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length based scheduling and congestion control," in *Proc. of 24th Annual Joint Conference of the IEEE Computer and Communications Societie*, pp. 1794–1803, March, 13-17, 2005. Article (CrossRef Link)
[21] R. J. Gibbens and F. P. Kelly, "Resource pricing and the evolution of congestion control," *Automatica*, vol. 35, no. 12, pp. 1969–1985, December, 1999. Article (CrossRef Link)
[22] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Now Publishers Inc*, 2006. Article (CrossRef Link)

[23] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 302–315, April, 2006. Article (CrossRef Link)

Zhenzhen Jiao received his PhD degree in Research Center of Ubiquitous Sensor Networks, University of Chinese Academy of Sciences, Beijing, China, in 2015. He is currently a research engineer in China Academy of Information and Communications Technology, Beijing, China. He has served on technical program committees for many international conferences and symposia and also as reviewers for many international journals. His research interests include wireless multihop network protocol and algorithm design and Internet of Things.



Baoxian Zhang received his PhD degree in Electrical Engineering from Northern Jiaotong University, China, in 2000. He is currently a Full Professor with the Research Center of Ubiquitous Sensor Networks at the University of Chinese Academy of Sciences (UCAS), Beijing, China. Prior joining UCAS, he was a Research Scientist with School of Information Technology and Engineering, University of Ottawa, Canada from 2002 to 2005. From 2001 to 2002, he was a Postdoctoral Fellow with Department of Electrical and Computer Engineering, Queen's University, Kingston, Canada. He is currently an Associate Editor of *IEEE Systems Journal* and has served as Guest Editors of several special issues including *IEEE Journal on Selected Areas in Communications* and *Elsevier Ad Hoc Networks Journal***.** He has served on technical program committees for many international conferences and symposia. He has published over 150 refereed technical papers in archival journals and conference proceedings. His research interests cover network protocol and algorithm design, wireless ad hoc and sensor networks, Internet of Things, IP networks. Dr. Zhang is a senior member of the IEEE (2012).

Jun Zheng is a Full Professor with the School of Information Science and Engineering at Southeast University (SEU), China. He received his Ph. D. degree in electrical and electronic engineering from The University of Hong Kong in 2000. Before joining SEU in 2009, he was with the School of Information Technology and Engineering, University of Ottawa, Canada. He has co-authored (first author) the book *Wireless Sensor Networks: A Networking Perspective*, published by Wiley-IEEE Press, and has co-authored over 100 technical papers in refereed journals and peer-reviewed conference proceedings. He is the recipient of an ICC 2014 Best Paper Award. His current research interests include vehicular ad hoc networks, mobile communication networks, wireless sensor networks, focused on network architectures and protocols. He serves as a Technical Editor of IEEE Communications Magazine and is an editorial board member of several other refereed journals, including Elsevier Ad Hoc Networks Journal and Springer Wireless Networks. He has co-edited twelve special issues for different refereed journals and magazines, including IEEE Communications Magazine, IEEE Network, and IEEE Journal on Selected Areas in Communications, all as Lead Guest Editor. He has served as the founding General Chair of AdHocNets'09, General Chair of AccessNets'07, and TPC or Symposium Co-Chair for many international conferences and symposia, including IEEE ICC and GLOBECOM. He has also served as a technical program committee member for a number of international conferences and symposiums. He is a senior member of the IEEE, IEEE Communications Society, and IEEE Vehicular Technology Society.