Control of Wireless Networks with Flow Level Dynamics under Constant Time Scheduling

Long Le and Ravi R. Mazumdar
Department of Electrical and Computer Engineering
University of Waterloo,Waterloo, ON, Canada N2L 3G1
Email: longble@engmail.uwaterloo.ca, mazum@ece.uwaterloo.ca.

***Abstract*— We consider the network control problem for wireless networks with flow level dynamics under the general k-hop interference model. In particular, we investigate the control problem in low load and high load regimes. In the low load regime, we show that the network can be stabilized by a regulated maximal scheduling policy considering flow level dynamics if the offered load satisfies a constraining bound condition. Because maximal matching is a general scheduling rule whose implementation is not specified, we propose a constant-time and distributed scheduling algorithm for a general k-hop interference model which can approximate the maximal scheduling policy within an arbitrarily small error. Under the stability condition, we show how to calculate transmission rates for different user classes such that the long-term (time average) network utility is maximized. This long-term network utility would capture the real network performance due to the fact that under flow level dynamics, the number of users randomly changes so instantaneous network utility maximization does not result in useful network performance. Our results imply that congestion control is unnecessary when the offered load is low and optimal user rates can be determined to maximize users' long-term satisfaction. In the high load regime where the network can be unstable under the regulated maximal scheduling policy, we propose the cross-layer congestion control and scheduling algorithm which can stabilize the network under arbitrary network load. Through extensive numerical analysis for some typical networks, we show that the proposed scheduling algorithm has much lower overhead than other existing queue-length-based constant-time scheduling schemes in the literature, and it achieves performance much better than the guaranteed bound. In addition, using congestion control in the low load condition results in much lower average utility compared to that due to the optimal transmission rate derived in the paper.**

***Index Terms*— Flow level dynamics, capacity region, constant-time scheduling, network stability, maximal matching, k-hop interference model**

I. Introduction

Resource allocation in communication networks has been an active research topic for the last several years. While optimal rate control in wired networks can be achieved by a distributed algorithm [1]-[4], solving this problem in wireless networks is much more challenging. In fact, the bottleneck of the resource allocation problem in wireless networks lies in the scheduling sub-problem [5]-[6]. The difficulty of the scheduling sub-problem comes from the interference coupling of simultaneous transmissions from different wireless links in the network.

In general, interference coupling in wireless networks depends on the communication technologies employed at the physical layer. For example, the node exclusive interference model can be assumed for Bluetooth networks or FH-CDMA networks [7], [8]. This interference model is also referred to as one-hop interference model. Also, for the 802.11 WLAN with four-way handshake (i.e., with RTS/CTS), the two-hop interference model is implicitly assumed in the MAC protocol. Moreover, it has been shown in [9] that in certain network settings and QoS requirements, neither one-hop nor two-hop interference model is optimal to achieve the maximum number of simultaneous transmissions in the network.

In [9] the authors proposed a general interference model called k-hop interference model which is determined by a single parameter k. For this interference model, wireless links $k+1$ or more hops away from one another can be scheduled to transmit data at the same time. Developing a joint resource allocation and scheduling algorithm for this k-hop interference model is, therefore, much more desirable than working with a special case of this general interference model. This is indeed what we will pursue in this paper.

Regarding the scheduling problem in wireless networks, there are several optimal and suboptimal schemes proposed in the literature. In a seminal paper [10], Tassiulas and Ephremides proposed an optimal back-pressure policy which achieves the maximum network throughput. This scheduling policy is, however, centralized and computationally expensive. In [11], a randomized linear-complexity scheduling algorithm was proposed where a transmission schedule in time slot t was constructed by choosing the schedule with larger total weight between the schedule in time slot $t-1$ and a newly-generated one in time slot t. This idea was used to develop distributed throughput-optimal scheduling policies in [12]-[14] for one-hop and two-hop interference models. Note that these scheduling algorithms achieve full utilization of wireless networks with respect to what remains in the data transmission phase only. Specifically, a large amount of bandwidth has been wasted to exchange control information in the schedule construction phase which would otherwise be used for data transmission. In general, the amount of scheduling overhead grows with the network size for these throughput-optimal scheduling policies.

Due to implementation constraints, the time slot interval is usually limited to a few milliseconds as in most current wireless systems. Therefore, developing a scheduling algorithm with low and constant-time overhead would be very desirable. In fact, some queue-length-based constant-time scheduling algorithms were proposed for one and two hop interference models [15]-[17] recently in the literature. These scheduling algorithms only achieve a guaranteed faction of the capacity region but they have constant time overhead. For practical implementation, collecting queue length information may be difficult, and it will create further overhead. A more general maximal scheduling policy was considered in [19], [20] where several throughput performance bounds were investigated. However, implementation of this general scheduling policy and investigation of its actual performance in typical wireless networks were not conducted in these papers.

In practice, it is desired that each wireless node only communicates with its neighbors (e.g., those whose transmissions interfere with that of the underlying node) to construct a transmission schedule in each time slot. Also, scheduling algorithms should work for a general class of interference models (e.g., k-hop interference model [9]). Another aspect which was ignored by most existing works in the literature is that no conflict-free schedule is available to exchange control information at the beginning of each time slot. Therefore, control information can only be exchanged by using contention-based transmissions which renders information exchange more than one hop away a time-consuming operation. Also, it is important to quantify amount of time/overhead used to construct the schedule and to develop explicit procedure to exchange control information in each time slot.

In this paper, we show the performance guarantee of the *regulated maximal scheduling* policy in wireless networks considering flow level dynamics. In fact, *regulated maximal scheduling* is a combination of the maximal scheduling policy [19], [20] and a traffic regulator implemented at each wireless link. Because regulated maximal scheduling is a general rule, we propose a constant-time and distributed algorithm to implement it in each time slot. We show that the proposed scheduling algorithm can approximate the regulated maximal scheduling policy within an arbitrarily small error. The proposed scheduling algorithm works for a general k-hop interference model and does not require queue length information. Moreover, we explicitly describe how wireless links coordinate their contentions to construct the schedule in each time slot.

The afore-mentioned performance guarantee result implies that we do not need to perform congestion control even with flow level dynamics if the traffic load lies within a region which can be stabilized by the underlying scheduling algorithm. This is an interesting finding given the fact that there are existing works which employ congestion control algorithms to stabilize the network [5], [21] under low network load. Given the fact that the scheduling algorithm is a randomized one and the number of users dynamically changes, instantaneous network utility maximization would not result in good network performance. Because of this, we are interested in finding transmission rate for each user class which achieves maximum long-term (time average) utility. In fact, we show that there exists optimal transmission rates for all user classes to achieve such maximum long-term utility. When the network load is high, we propose a cross-layer congestion control algorithm which can stabilize the network for arbitrary network load.

The results presented in this paper have several important implications for system implementation. First, we do not need to perform congestion control in low network load even with flow level dynamics. Hence, communication overhead as well as implementation issues such as asynchronous [8] and noisy [24] feedback due to congestion control operations can be completely avoided. Second, the problem of network utility maximization can be decoupled from that of stabilizing the network. Specifically, the network can be stabilized by implementing traffic regulators at wireless links together with a suitable scheduling mechanism. In fact, we show via numerical examples that using congestion control algorithm to stabilize the network in low load actually degrades the long-term network utility considerably.

The remaining of this paper is organized as follows. We describe the system models and performance bound in section II. Performance guarantee of the regulated maximal scheduling policy is presented in section III. In section IV, we present the distributed scheduling algorithm to approximate the maximal scheduling policy. We derive the optimal transmission rates to achieve long-term utility maximization in section V. The cross-layer congestion control and scheduling algorithm in the high load regime is described in section VI. Some numerical results are presented in section VII and section VIII states the conclusion. For notational convenience, we will put elements of different measures into the corresponding vectors. For example, the vector of transmission rates will be denoted by $\vec{x}$ where x_s is its s-th element which is the transmission rate of class-s users.

II. System Models and Performance Bound

We model a wireless network as a directed graph $G = (V, E)$ where V is the set of wireless nodes and E is the set of wireless links. A wireless link from node i to node j exists if node j can correctly receive information transmitted by node i. In practice, existence of such a link depends on transmission power, path loss, fading, interference, desired bit error rate and other factors.

We assume that there are S classes of users each of which associates with a fixed routing path from a source node to a destination node. The user routes are stored in an incidence matrix $[H_s^k]$ where $H_s^k = 1$ if link k is on the route of class-s users and $H_s^k = 0$ otherwise. Users of class s arrive to the network with rate λ_s and each brings a file for transfer whose size is exponentially distributed with mean $1/\mu_s$. The offered load by class-s users is, therefore, $\rho_s = \lambda_s/\mu_s$. The vector of offered load will be denoted as $\vec{\rho} = [\rho_1, \rho_2, \cdots, \rho_S]$. We assume that users of each class transmit at the same rate.

Interference constraints are denoted by a contention matrix $[C_{ij}]_{i,j \in E}$. Specifically, link i is said to interfere with link j if $C_{ij} = 1$ and $C_{ij} = 0$ otherwise. This general notation of the interference relationship is used to describe the capacity region and to derive the performance guarantee of the regulated maximal matching policy in this section and section III only. The k-hop interference model is a special class representing this general interference relationship. The k-hop interference model is assumed in all remaining sections of this paper. Time is divided into slots of unit duration. Link l can transmit at rate R_l if its interfering links are not scheduled to transmit in a same time slot. In the following, we provide some important definitions which will be used in the sequel.

Definition I: Interference set I_l of link l is the set of links which interfere with link l, i.e.,

$$I_l = \{k \in E : C_{kl} = 1\}. \quad (1)$$

Definition II: Interference degree $d_I(l)$ of link l is the maximum number of links in its interference set which do not interfere with each other.

Definition III: Interference degree $d_I(G)$ of graph G is the maximum interference degree of its links, i.e., $d_I(G) = \max_{l \in E} d_I(l)$.

Capacity region is defined to be the set of traffic load such that the network can be stabilized by some scheduling policy. In [10], capacity region for wireless networks is well characterized. In particular, capacity region is given by the set

$$\Omega = \left\{ \vec{\rho} : \left[\sum_{s=1}^{S} \frac{H_s^l \rho_s}{R_l} \right]_{l \in E} \in \mathrm{Co}(\mathcal{R}) \right\} \quad (2)$$

where $\mathrm{Co}(\mathcal{R})$ is the convex hull of all link schedules $\mathcal{R}$ that satisfy the constraints imposed by the underlying interference model. A scheduling policy is said to be throughput optimal if it stabilizes the network for all offered load within the capacity region Ω.

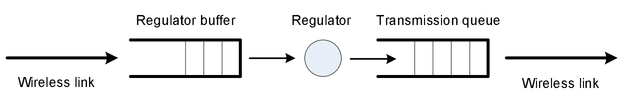

Fig. 1. Regulator implementation at each wireless link.

We assume that a new schedule is constructed in the first phase of each time slot which is used to transmit data in the second phase of each time slot. Traffic flows in the network may traverse one hop or multiple hops. For the multiple-hop case, we assume that traffic of each user class is regulated before entering a transmission queue for being transfered over each wireless link. The employment of regulators was previously proposed by Humes to stabilize manufacturing systems [22] which have been shown to be unstable in some cases due to cycles of material flow [23]. Regulators were recently used in wireless networks [8], [19]. A λ-regulator associated with link l generates packets to transmission queue of link l with a maximum rate of λ. A regulator can be implemented as follows. In each time slot, a λ-regulator associated with link l checks the corresponding regulator queue. If the queue length is greater than link capacity R_l then it transfers R_l units of data to the transmission queue with probability λ/R_l. Otherwise, it transfers nothing. The regulator implementation is illustrated in Fig. 1.

In this paper, we assume a maximal scheduling policy which was investigated in [19], [20]. The maximal scheduling rule can be described as follows. For any link $l \in E$ with transmission queue length larger than the link capacity in any time slot, it is required that at least one link in its interference set I_l be scheduled. Specifically, if $Q_l/R_l \geq 1$ (where Q_l is the queue length of transmission queue for link l), we require

$$\sum_{k \in I_l} \pi_k \geq 1 \quad (3)$$

where $\pi_k = 1$ if link k is scheduled and $\pi_k = 0$ otherwise. Due to the combination of maximal scheduling and regulator implementation, the resulting scheduling will be called *regulated maximal scheduling* in the sequel. Note that maximal scheduling is a general scheduling rule without specific implementation. We will present a constant time and distributed scheduling algorithm which approximates the maximal scheduling in section IV. The following performance bound of the maximal scheduling policy was proved in [20], [21], it is restated here for completeness.

Lemma 1: For all traffic load $\vec{\rho}$ within the capacity region defined in (2), we have

$$\sum_{k \in I_l} \sum_{s=1}^{S} \frac{H_s^k \rho_s}{R_k} \leq d_I(l), \quad \forall l \in E. \quad (4)$$

This upper bound will be used to quantify the throughput guarantee of the regulated maximal scheduling policy in the next section.

III. Performance of the Regulated Maximal Matching Scheme

In this section, we show that the network is stable under the regulated maximal scheduling when the offered load satisfies a specific condition. We assume that a $(\rho_s + k\epsilon)$-regulator is employed at k-th hop on the route of class-s users. It is worth to mention that the following stability result is similar in spirit to that in [19], although there is an important difference here. In fact, we capture flow dynamics in this paper while the authors in [19] only considered dynamics at the packet level. In [5] and [21], the authors captured flow dynamics but their stability results were achieved by a cross-layer congestion control algorithm. In this paper, network stability is achieved by a simple regulator implementation so communication overhead involved in the congestion control operation can be avoided. The stability result is stated in the following proposition.

Proposition 1: If the traffic load satisfies

$$\sum_{k \in I_l} \sum_{s=1}^{S} \frac{H_s^k \rho_s}{R_k} < 1, \quad \forall l \in E \quad (5)$$

then the network is stable under the regulated maximal scheduling policy. This condition will be called a constraining bound in the sequel.

Proof: The proof is in Appendix A.

This result was also stated in [21] where the network was stabilized by a cross-layer congestion control algorithm. Note that this constraining bound is tight in the sense that the network can become unstable with any arbitrarily small increase of the bound (i.e., the right hand side of (5) becomes

$1+\kappa$ for any $\kappa > 0$). This fact was proved in several papers (for example, see [19], [20]). In the following, we state the performance guarantee for the regulated maximal scheduling policy.

Lemma 2: The regulated maximal scheduling policy achieves $1/d_I(G)$ capacity region.

Proof: The proof follows directly by comparing the upper and constraining bounds on capacity region in (4) and (5), respectively, and using the definition of $d_I(G)$. □

IV. Distributed Scheduling Algorithm

As mentioned before, maximal scheduling is a general rule whose implementation is not specified. In this section, we present a distributed scheduling algorithm which approximates the maximal scheduling policy in each time slot within an arbitrarily small error. In fact, the proposed algorithm will include the BP-SIM scheduling algorithm [17] proposed for the node exclusive (i.e., one-hop) interference model as a special case. Our proposed algorithm works with the general k-hop interference model. Also, in contrast to the existing queue-length-based scheduling algorithms [16], [17], in our algorithm each node with incident backlogged links does not require queue length information of other links in its neighborhood to construct the transmission schedule. In addition, the proposed algorithm is fully distributed and it has constant time overhead which does not grow with the network size. Our proposed algorithm is, therefore, much more flexible and general than existing ones in the literature. For ease of reference, we will refer to our scheduling algorithm as random approximate maximal matching (RAMM) scheduling in the sequel.

A. Algorithm Description

The RAMM algorithm is run in the first phase of each time slot. Specifically, we divide each time slot into two phases: a scheduling phase and a data transmission phase. The transmission schedule is constructed in the scheduling phase, and it is used to transmit data in the data transmission phase. The scheduling phase is further divided into K rounds each of which contains B mini-slots. In each round, new links are added to the current transmission schedule. The transmission schedule obtained at the end of the K-th round will be used to transmit data in the data transmission phase. In addition, only wireless links whose queue lengths are larger than the link capacity are scheduled by the algorithm in each time slot. The time slot structure of the RAMM algorithm is illustrated in Fig. 2.

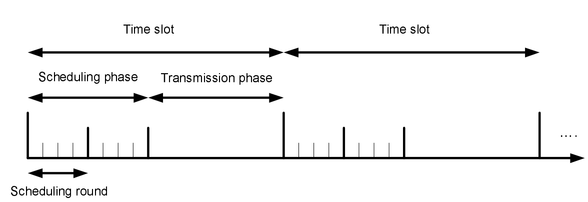

Fig. 2. Timing diagram of the RAMM scheduling algorithm.

Links are added to the schedule in each round through a matching request and matching acknowledgment message exchange as follows. At the beginning of each round, each *active* node (the notion of *active/inactive* nodes will be clarified shortly) decides to be *left* or *right* with probability 1/2. Nodes becoming right wait to receive matching requests from their neighboring nodes. Backlogged links are added to the schedule in each round as follows. Each left node with at least one backlogged outgoing link (i.e., a link from this node to one of its neighbors) will choose a random backoff in $[1, B]$. When the backoff expires, a left node will choose one of its backlogged neighbors randomly to send a matching request if it has not heard any matching requests transmitted by other nodes so far in the round. A right node which receives a matching request will reply with a matching acknowledgment message and the corresponding link is added to the schedule. We assume that if two or more matching requests are transmitted in one mini-slot, collision occurs and no matching acknowledgment message is transmitted.

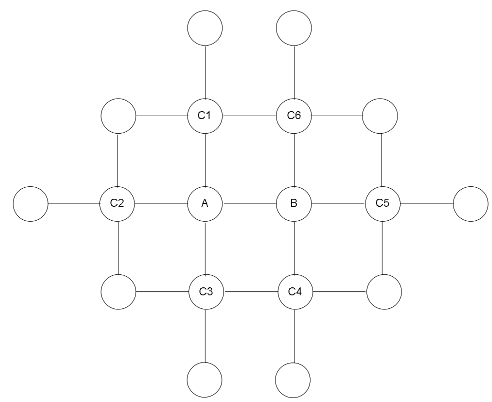

Fig. 3. Illustration of the contention resolution of RAMM scheduling under two-hop interference model.

In each round, we require that if a link is added to the transmission schedule, all wireless links in its interference set be not added to the transmission schedule in subsequent rounds. This requirement guarantees that we will obtain a conflict-free transmission schedule at the end of the scheduling phase. It is observed that this requirement can be easily achieved by one-hop and two-hop interference models. Specifically, for the one-hop interference model after a link is added to the schedule, both its transmitting and receiving nodes will not transmit and reply any matching requests. For the two-hop interference model after a link is added to the schedule, all their one-hop neighboring nodes (i.e., one hop away) of both transmitting and receiving nodes will be aware of this (through hearing the matching request or matching acknowledgment) and will not transmit or reply any matching requests until the end of the scheduling phase.

For a general k-hop interference model with $k \geq 3$, we assume that a large enough power level is used to transmit matching request/acknowledgment messages in the scheduling phase so that if a link is added to the schedule in one round, all nodes within $k-1$ hops from both the transmitting and receiving nodes of the link are aware of this so they will not transmit or reply any matching requests in subsequent rounds. Nodes within $k-1$ hops from the transmitting and receiving nodes of any links in the schedule are called *inactive* nodes. All other nodes are *active* ones. Note that any *inactive* node will remain inactive until the end of the scheduling phase. In general, the number of nodes participating in the schedule construction process reduces rapidly over consecutive rounds. Because new links are kept added to the existing schedule in each round, the transmission schedule in the last round would approximate well the maximal schedule if B and K are large enough. We will show the performance guarantee of the proposed scheduling algorithm in the next subsection.

The contention resolution of the RAMM scheduling algorithm under two-hop interference model is illustrated in Fig. 3. In this figure, if link AB is added to the schedule, all nodes C_i ($i = 1, 2, \cdots, 6$) will be aware of this through hearing either the matching request or matching acknowledgment message from A or B. Hence, after link AB is added to the schedule, these nodes (C_i, $i = 1, 2, \cdots, 6$) will become inactive. Consequently, all the links in this figure except AB which are conflict with link AB will never be added to the schedule in the subsequent rounds.

B. Analysis

Now, let degree d_i of node i be the number of nodes having links directly connecting to node i (i.e., one-hop neighbors of node i). Let d^* be the maximum of d_i for all nodes in the network (i.e., $d^* = \max_{i \in V} d_i$). In addition, a matching request transmitted by one node may collide with those transmitted by other nodes. Let I_i be the number of nodes whose transmitted matching requests may collide with that of node i if node i and one or more of these nodes transmit simultaneously under the corresponding power level used in the scheduling phase. Let I be the maximum of I_i (i.e., $I = \max_{i \in V} I_i$). Also, let I_0^* be the maximum number of nodes which are at most $k-1$ hops away from either A or B including A and B for any link AB in the network. We have the following result.

Proposition 2: For any $\mu \in (0, 1)$, we can choose the number of scheduling rounds K which depends only on B, d^*, I, I_0^*, and μ but independent of network size such that for any backlogged link l, the probability that at least one backlogged link in its interference set I_l is scheduled after K rounds is larger than or equal to μ.

Proof: The proof is in Appendix B.

Using RAMM scheduling algorithm together with regulator implementation as described in section II, we have the following stability result.

Proposition 3: If the traffic load satisfies

$$\sum_{k \in I_l} \sum_{s=1}^{S} \frac{H_s^k \rho_s}{R_k} < \mu, \quad \forall l \in E \tag{6}$$

and under the condition stated in proposition 2, the network will be stable when RAMM algorithm is used together with the regulator implementation as described in section II.

Proof: The proof follows the same line with that of Proposition 1. However, the right hand side of the constraining bound becomes μ instead of one due to the performance bound achieved by RAMM scheduling scheme. □

The result in this proposition means that we can achieve the performance bound of the regulated maximal matching stated in proposition 1 within an arbitrarily small error by using the RAMM scheduling algorithm with constant-time overhead.

V. Long-Term Utility Maximization under Low Load Condition

Proposition 3 also means that when regulators and RAMM scheduling algorithm are implemented and traffic load satisfies the condition stated in (6), the network is stable as long as user rates are bounded away from zero. As a consequence of this result, it is clear that we do not need any congestion control algorithm as long as the traffic load in the network is low. Hence, communication overhead due to message exchange of the congestion control algorithm can be avoided if regulators are implemented in the network. In addition, the number of users for each class changes dynamically due to the flow level dynamics, so instantaneous network utility maximization may not lead to good network performance. Therefore, under this stability condition, it is natural to ask: how to choose user rates such that optimal long-term (time average) network utility can be achieved? Specifically, our objective is to maximize the long-term network utility which can be explicitly stated as

$$\max_{\vec{x}(t)} \lim_{\tau \to \infty} \frac{1}{\tau} \int_{t=0}^{\tau} \left[\sum_{s=1}^{S} n_s^t(t) U_s(x_s(t)) \right] dt \tag{7}$$

where $n_s^t(t)$ and $x_s(t)$ are the number of class-s users transmitting in time slot t and their transmission rate, respectively; $U_s(x_s)$ is the utility function, which can, for example, reflect the level of satisfaction for class-s users. We assume that users arriving during time slot t can only transmit from time slot $t+1$ onward. Suppose that the queueing process at each source node is ergodic (this fact was justified in [25]). Let $f(\vec{n}^t, \vec{x})$ denote the joint probability density function of $\vec{n}^t$ and $\vec{x}$ in equilibrium. Because elements of $\vec{n}^t$ are pairwise independent, we have $f(\vec{n}^t, \vec{x}) = \left[\prod_{s=1}^{S} f(n_s^t | \vec{x}) \right] f(\vec{x})$. Thus, we can rewrite (7) as

$$\max_{\vec{x}} \int_X \sum_{s=1}^{S} \left[\sum_{n_s^t = 0}^{\infty} n_s^t U_s(x_s) f(n_s^t | \vec{x}) \right] f(\vec{x}) d\vec{x}. \tag{8}$$

Let us define

$$\begin{aligned} g(\vec{x}) &= \sum_{s=1}^{S} \sum_{n_s^t = 0}^{\infty} n_s^t U_s(x_s) f(n_s^t | \vec{x}) \\ &= \sum_{s=1}^{S} U_s(x_s) \sum_{n_s^t = 0}^{\infty} n_s^t f(n_s^t | \vec{x}) \\ &= \sum_{s=1}^{S} U_s(x_s) E\left[N_s^t | \vec{x} \right] \\ &= \sum_{s=1}^{S} \frac{\rho_s}{x_s} U_s(x_s) \end{aligned}$$

where we have used Little's law in deriving $E\left[N_s^t|\vec{x}\right]$ in the above equation. Specifically, the expected waiting time for a class-s user is $1/(\mu_s x_s)$, using Little's law we have $E\left[N_s^t|\vec{x}\right] = \lambda_s/(\mu_s x_s) = \rho_s/x_s$. Thus, we can rewrite (8) as

$$\max_{\vec{x}} \int_X g(\vec{x}) f(\vec{x}) d\vec{x} \tag{9}$$

Now, suppose we wish to find optimal user rate $x_s \in [0, M_s]$. Let x_s^* be the maximum of $g_s(x_s) = \rho_s/x_s U_s(x_s)$ in $[0, M_s]$ and the corresponding optimum rate vector is $\vec{x}^*$. Then, it is easy to see that choosing $f(\vec{x}) = \delta(\vec{x} - \vec{x}^*)$ will maximize (9) where $\delta(.)$ is the delta function. Thus, the long-term utility maximization can be achieved by allowing users of class s to transmit at the optimal rate x_s^*. In summary, when the traffic load satisfies (6), the network is stable under the proposed scheduling policy and no congestion control is needed. In addition, we can decouple the long-term utility maximization from stability under this stability condition.

Example: When the utility function is $U_s(x_s) = \ln(x_s)$ which corresponds to proportional fair rate allocation among users, we have $g_s(x_s) = \rho_s/x_s \ln(x_s)$. The global maximum of $g_s(x_s)$ is $x_s^* = e$. Thus, if $M_s > e$, the optimal transmission rate to achieve maximum long-term utility is $x_s^* = e$. We will compare long-term utility under this solution and for the case where cross-layer congestion control algorithm is used [21].

VI. Congestion Control under Heavy Load

In this section, we consider the heavy load regime where the bound stated in (5) is violated. This may be the case when there are many long-lived flows in the network.

A. Cross-Layer Congestion Control Algorithm

In this subsection, we present a cross-layer congestion control algorithm which can stabilize the network under any offered load. We assume that the flow dynamics are slow compared to the time scale of a congestion control algorithm. Hence, we assume that there are fixed number of users of each class s in the network which will be denoted by N_s. Our proposed cross-layer congestion control algorithm works as follows.

- User rate is determined by

$$x_s(t) = \left[\min\left\{U_s^{'-1}\left(\sum_{l\in E} q_l(t) \sum_{k\in I_l : H_s^k = 1} \frac{1}{R_k}\right),\ M_s\right\}\right]^+$$

  where $U_s^{'-1}(.)$ is the inverse of derivative of utility function $U_s(.)$ and $[x]^+ = \max[x, 0]$.
- The implicit costs are updated by

$$q_l(t+1) = \left[q_l(t) + \alpha_l\left(\sum_{k\in I_l}\sum_{s=1}^{S} \frac{H_s^k N_s x_s}{R_k} - 1 + \kappa\right)\right]^+$$

  where $\kappa > 0$ is a small number and α_l is the step-size.
- Transmission scheduling: The regulated maximal scheduling policy is employed in each time slot.

We assume that the utility function $U_s(x_s)$ is increasing, strictly concave, twice differentiable. The proposed congestion control algorithm implicitly solves the following optimization problem.

$$\begin{array}{ll} \text{maximize} & \sum_{s=1}^{S} N_s U_s(x_s) \\ \text{subject to} & \sum_{k\in I_l} \sum_{s=1}^{S} \frac{H_s^k N_s x_s}{R_k} \le 1 - \kappa, \quad \forall l \in E \end{array} \tag{10}$$

Note that constraints of this optimization problem come from the constraining bound stated in (5). Here, the transmission rate of class-s users is used instead of its offered load. We introduce a small $\kappa > 0$ on the right hand side of these constraints to ensure that feasible solutions of this optimization problem strictly satisfy the constraining bound. By introducing Lagrange multipliers q_l for the constraints in (10), we have the following Lagrangian

$$L(\vec{x}, \vec{q}) = \sum_{s=1}^{S} N_s U_s(x_s) - \sum_{l\in E} q_l \left[\sum_{k\in I_l}\sum_{s=1}^{S} \frac{H_s^k N_s x_s}{R_k} - 1 + \kappa\right]$$

Using the standard dual decomposition technique, we can calculate the user rate from

$$x_s(t) = \underset{0\le x_s\le M_s}{\operatorname{argmax}} \left[N_s U_s(x_s) - \sum_{l\in E} q_l(t) \sum_{k\in I_l : H_s^k = 1} \frac{N_s x_s}{R_k}\right]$$

which gives the user rate update rule presented in the algorithm. Also, implicit cost update step of the proposed algorithm comes from the dual update of the underlying optimization problem. Now, let x_s^* be the optimal solution of the presented algorithm, we implement a $(x_s^* + k\epsilon)$-regulator in the k-th hop of class s users. We have the following results on the stability of the presented algorithm.

Proposition 4: If the stepsize is chosen to be small enough, the presented algorithm can stabilize the network under any offered load.

Proof: The proof is in Appendix C.

It is expected that a congestion control algorithm should stabilize the network under any offered load. In this respect, the proposed congestion control algorithm does a good job compared to those in [5], [21]. Note that in practice where the RAMM scheduling algorithm is implemented to approximate the maximal matching policy (RAMM instead of maximal matching is used in step 3 of this algorithm), we can modify the proposed congestion control algorithm by replacing 1 on the right hand side of the constraint in (10) by μ and modify the algorithm accordingly.

B. Some Implementation Issues

In practice, there is no simple way to know whether or not the constraining bound in (5) is satisfied. Therefore, it is unknown when the congestion control algorithm proposed in section VI.A should be activated. Moreover, it would be wise to avoid performing congestion control if it is unnecessary. Also, users in the network should transmit at their optimal rates as calculated in section V to maximize the long-term utility if the network is known to be stable by the regulated maximal scheduling policy.

To achieve such goals, users would attempt to transmit at their optimal rates assuming that the offered load is low and the network is stable. If this is not the case, the network should be equipped with an appropriate mechanism to detect and inform all users about the ongoing congestion and request them to activate the congestion control algorithm. Congestion can be detected at network nodes through observing frequent buffer overflows and/or large end-to-end delay. Upon detecting network congestion, network nodes should send appropriate feedback to inform all users in the network and request them to run the congestion control algorithm. These operations ensure that the best performance can be achieved and the most appropriate operations are performed at all times.

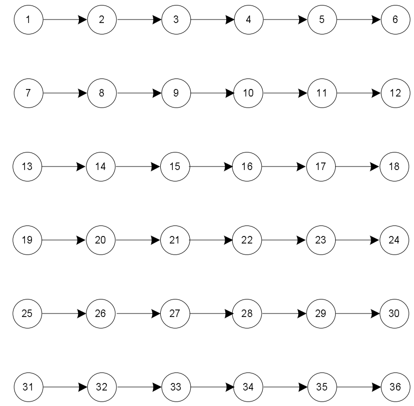

Fig. 5. Grid network of 36 nodes with 30 one-hop flows or 6 multihop flows.

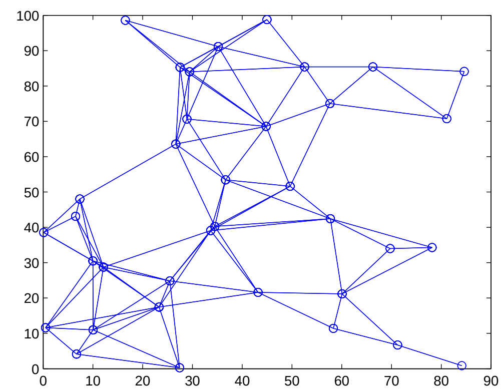

Fig. 6. Random network of 36 nodes with 198 bidirectional links (99 undirectional links), d^* = 9, I = 24.

VII. Numerical Results

In this section, we show some illustrative numerical results for the proposed scheduling algorithm and the long-term utility maximization. We consider grid networks with one-hop and multihop flows. We assume that transmission rate on each wireless link equals R_l = 10 units/time slot, average length of each file brought by any user class is $1/\mu_s$ = 10 units. Users of each class arrive according to Poisson process with arrival rate λ_s. We vary arrival rate to adjust the traffic load $\rho_s = \lambda_s/\mu_s$. Here, a unit of data is a block of information bits of suitable size. We assume all flows have the same load ρ in all the results.

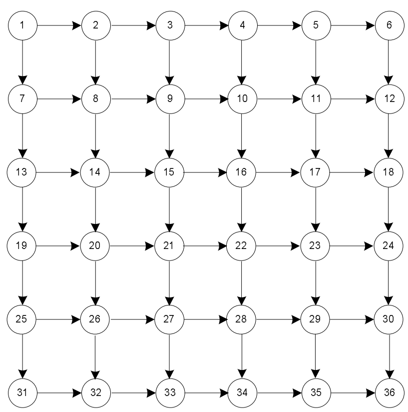

Fig. 4. Grid network of 36 nodes with 60 one-hop flows or 12 multihop flows.

A. Performance of Scheduling Algorithms

In Fig. 7, we show the minimum probability of achieving a maximal schedule due to the RAMM algorithm versus the number of rounds K under different maximum backoff values B for the grid and random topologies shown in Fig. 4, Fig. 6 under the one-hop interference model. For both topologies, we assume there are two flows in two different directions on each links (i.e., for any link AB, there are one flow from A to B and one flow from B to A). We assume that all the flows are always backlogged which is the worst case scenario. The probability of achieving a maximal schedule for a backlogged link is the probability that it is scheduled or at least one link in its interference set is scheduled. The minimum matching probability is the smallest probability for all links in the network. We have obtained this probability by averaging over 10^4 time slots. It is observed that with B = 3, to achieve a minimum matching probability of 0.9, we need K = 4 and 5 rounds for the grid and random topologies, respectively. Also, with B = 3 and K = 9, (i.e., the total number of mini-slots required is $M = B \times K$ = 27), the minimum matching probability is almost 1.

Similarly, we show the minimum probability of achieving a maximal schedule due to the RAMM algorithm versus the number of rounds K under different maximum backoff values B for the grid and random topologies under two-hop interference model in Fig. 8. This figure shows that with B = 4, we only need K = 11 for the grid topology and K = 13 for the random topology to achieve minimum matching probability very close to 1. It is also shown that even the size of the node interference set is very large (I = 22 for the grid topology and I = 24 for the grid topology), the improvement of minimum matching probability is very marginal when the maximum backoff B is larger than 8. In fact, it is more beneficial in terms of overhead (i.e., $M = B \times K$) to use small values of B (e.g., B = 4 or 5 is a good choice).

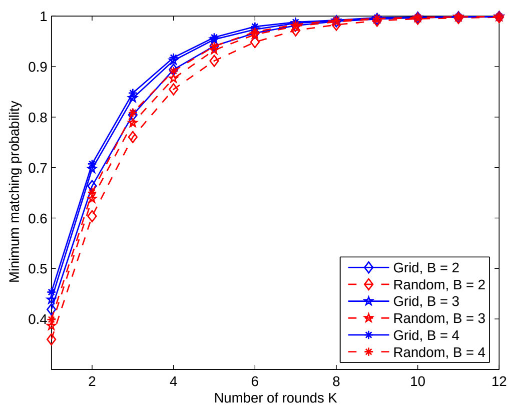

Fig. 7. Minimum probability of achieving the maximal scheduling versus number of rounds for a grid topology in Fig. 4 with 120 one-hop flows or random topology in Fig. 6 with 198 one-hop flows under one-hop interference model.

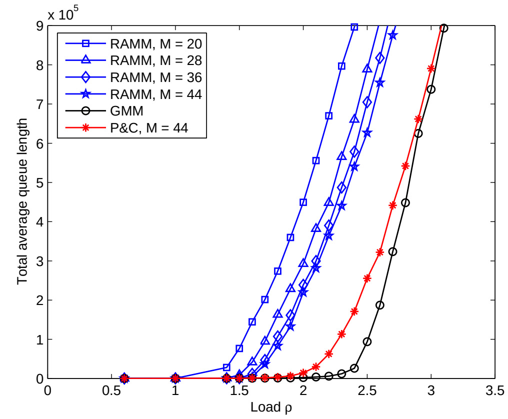

Fig. 9. Performance of RAMM scheduling algorithm under two-hop interference model (for maximum backoff value $B = 4$, grid network with 36 nodes and 30 one-hop flows in Fig. 5).

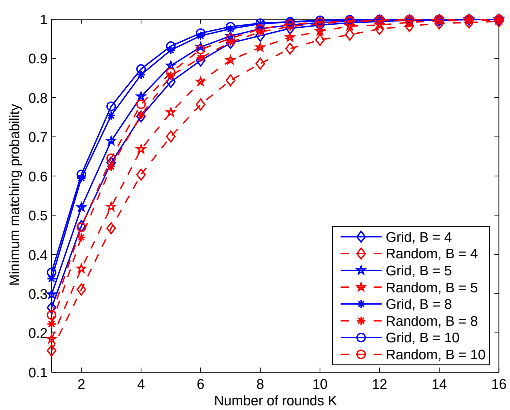

Fig. 8. Minimum probability of achieving the maximal scheduling versus number of rounds for a grid topology in Fig. 4 with 120 one-hop flows or random topology in Fig. 6 with 198 one-hop flows under two-hop interference model.

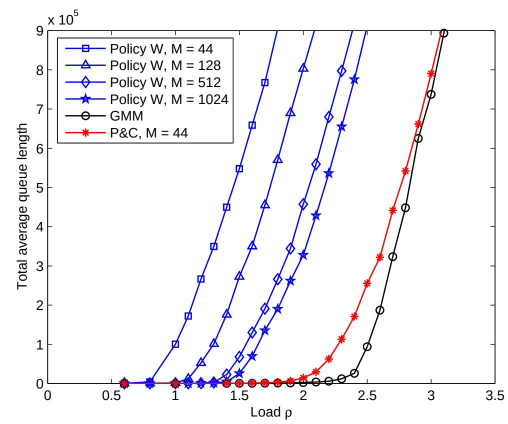

Fig. 10. Performance of policy W [16] under two-hop interference model (for M = 44, 128, 512, 1024, grid network with 36 nodes and 30 one-hop flows in Fig. 5).

B. Comparison of Different Scheduling Algorithms

In this sub-section, we compare performances of the proposed RAMM scheduling algorithm with the other two queue-length-based constant-time scheduling algorithms in [16], namely policy V and policy W for one-hop and two-hop interference models, respectively. For ease of reference, we briefly describe these two scheduling algorithms here. Before doing so, let us introduce some necessary notations. Let $s(l)$ and $r(l)$ denote the transmitting node and receiving node of link l, respectively. Also, let $E(i)$ denote the set of links connected to node i, and $N(l)$ denote the set of neighboring links sharing a common node with link l (i.e., $N(l) = E(s(l)) \cup E(r(l)) \setminus \{l\}$. In addition, let $N(l)^+$ denote the union of $N(l)$ and $\{l\}$, and n_l and n_l^+ denote the number of links in $N(l)$ and $N(l)^+$, respectively.

For the comparison purposes, we assume single-hop flows so data traffic will be directly buffered in the transmission queue for each link (i.e., no regulator implementation). In the scheduling policies V and W, each time slot also consists of scheduling phase and data transmission phase. There are M mini-slots in the scheduling phase. The two scheduling algorithms work as follows.

Policy V: In each mini-slot of the scheduling phase, each backlogged link contends with probability $\alpha \frac{x_l}{M}$ where $\alpha = (\sqrt{M} - 1)/2$ and

$$x_l = \frac{Q_l/R_l}{\max\left\{\sum_{k\in E(s(l))} Q_k/R_k, \sum_{k\in E(r(l))} Q_k/R_k\right\}}$$

where Q_l is the queue length and R_l is the transmission rate of link l, respectively.

Policy W: In each mini-slot of the scheduling phase, each backlogged link contends with probability $\beta \frac{y_l}{M}$ where

$$\beta = \frac{\sqrt{M} - 1}{n^*} \tag{11}$$

where $n^* = \max_{l\in E} n_l^+$ and

$$y_l = \frac{Q_l/R_l}{\max_{k\in N(l)^+} \sum_{h\in N(k)^+} Q_h/R_h}.$$

In Fig. 9, we show the performance of RAMM scheduling algorithm in a grid network with 36 nodes and 30 one-hop

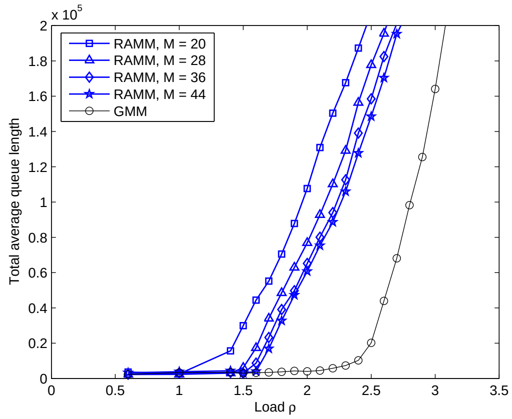

Fig. 11. Performance of RAMM scheduling algorithm under two-hop interference model (for B = 4, grid network with 36 nodes and 6 multihop flows in Fig. 5).

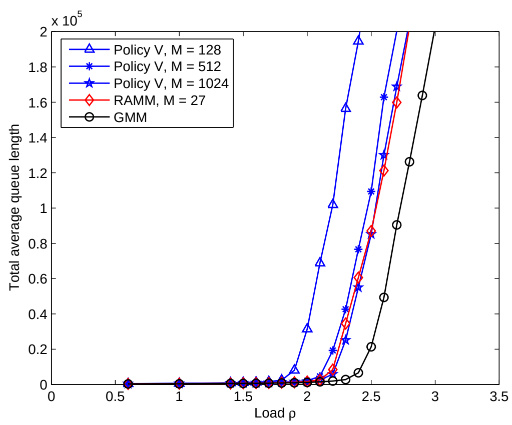

Fig. 12. Performance of RAMM (with B = 3, K = 9) and policy V [16] scheduling algorithms under one-hop interference model (for grid network with 36 nodes and 60 one-hop flows in Fig. 4).

flows as in Fig. 5. For the RAMM algorithm, we show the performance with different transmission rounds K while we fix the maximum backoff value of B = 4 (the number of mini-slots in the scheduling phase is $M = B \times K$). We also present performance of Pick-and-Compare (P&C) [11], [13] and greedy maximal matching (GMM) [5] scheduling schemes. For the P&C scheme, a new schedule is generated in each time slot by using RAMM algorithm. Then the total weights of the old schedule and the newly generated schedule are compared where the weight of a link is the length of its transmission queue, and the schedule with larger weight is chosen for transmission in the time slot. In fact, the huge complexity of the P&C scheme incurs in the "compare" step. For the GMM scheme, the schedule is constructed by adding one feasible link with the largest weight to the schedule and removing all conflicting links with the added link in each step. This procedure is repeated until no further link can be added to the schedule.

Note that both P&C and GMM scheduling schemes are either centralized or require huge overhead to implement in a distributed manner. Also, it is known that these two scheduling schemes achieve almost the capacity region. It is confirmed in this figure that P&C and GMM schemes achieve similar performance although GMM has a bit smaller total queue length for traffic load close to the boundary of the capacity region. It is evident that when we increase the number of rounds for the RAMM algorithm, we achieve better performance. Moreover, in the considered network $d_I(G)$ = 4, so the performance guarantee stated in proposition 1 is just 1/4 capacity region. However, the actual performance achieved by the proposed RAMM scheduling algorithm is much better than the performance bound as can be seen in Fig. 9.

In Fig. 10, we present the performance of the queue-length-based constant time scheduling policy W for the two hop interference model [16]. We show the performance of this scheduling scheme for different number of mini-slots used in the scheduling phase M (the first phase of this scheduling policy corresponds to the first phase of our RAMM scheduling algorithm). As is evident, with M = 44 mini-slots, policy W achieves much lower performance than the RAMM algorithm. Even with M = 1024 mini-slots, performance of policy W is still worse than that of RAMM algorithm with only M = 44 mini-slots. Note that due to practical implementation constraints, the time slot duration is usually limited to few milliseconds as in most current wireless systems. If the duration of a mini-slot is 20 μs as in the WLAN standard, M = 100 corresponds to 2 ms which is already quite large. Our scheduling algorithm, therefore, presents a significant advantage compared to policy W because we cannot make the time slot interval arbitrarily large in practice.

In Fig. 11, we show performance of the RAMM scheduling algorithm for the grid network with 6 multihop flows in Fig. 5. Similar performance to the single-hop case presented in Fig. 9 is observed for this setting. It is evident that although multihop flows contribute more traffic to the network because each flow traverses multiple links, scheduling algorithm is still the key to determine performance of the network.

In Fig. 12, we compare performance of RAMM and policy V [16] under one-hop interference model for the grid network with 60 one-hop flows in Fig. 4. It is evident that RAMM scheduling with only M = 27 mini-slots achieves similar performance with policy V for M = 1024 mini-slots. The RAMM scheduling algorithm, therefore, has significantly lower overhead compared to policy V. Note that the RAMM scheduling algorithm does not require queue length information, so it is much easier to implement compared to policy V and W. In addition, collecting queue length information will create further overhead which may also be very significant, especially for the two-hop interference model (because each node needs to forward queue length information of its incident links two hop away).

C. Long-term Utility

We compare the long-term (time average) utility under the optimal transmission rate derived in section V and under the case where a cross-layer congestion control algorithm is used. Specifically, we will consider the cross-layer congestion control algorithm proposed in [21] which is the extension of that for one-hop interference model in [5]. The cross-layer congestion control algorithm works as follows:

- Congestion price for each link l is updated as
$$q_l(t+1) = [q_l(t) + \alpha_l \Delta q_l(t)]^+ \quad (12)$$
where α_l is the step size and
$$\Delta q_l(t) = \sum_{k \in I(l)} \left[\sum_{s=1}^{S} H_s^k \int_t^{t+1} \frac{n_s(t) x_s(t)}{R_k} - 1_{k \in S(t)} \right]$$
and $S(t)$ denotes the set of links belonging to the schedule in time slot t, $1_{(.)}$ is the indicator function.
- Transmission rate of class-s user is updated as
$$x_s(t+1) = \min\left\{ \frac{1}{\sum_{l \in E} q_l(t+1) \sum_{k \in I(l)} \frac{H_s^k}{R_k}}, M_s \right\}. \quad (13)$$
- Transmission scheduling: The network is scheduled in each time slot by the corresponding scheduling algorithms (GMM or RAMM algorithm).

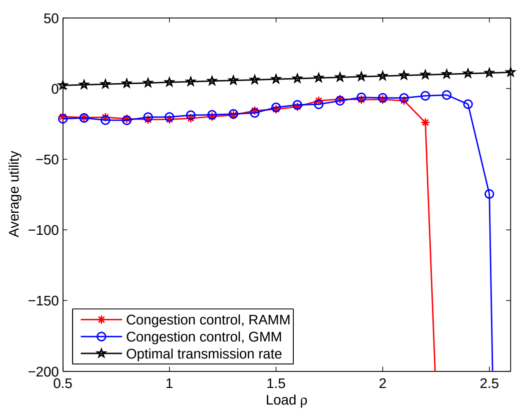

Fig. 13. Average utility of the proposed optimal transmission rate and with congestion control algorithm under one-hop interference model (for grid network with 36 nodes and 12 multihop flows, GMM and RAMM scheduling with B = 3, K = 9 in Fig. 4).

We consider the utility function $U_s(x_s) = \ln(x_s)$. Hence, for the proposed approach the optimal transmission rate for each user is $x_s^* = e$ (i.e., we assume that $M_s > e$). We will illustrate performance of the cross-layer congestion control algorithm under both GMM and RAMM scheduling algorithms. Long-term average utility is obtained by averaging the utility over 15.10^4 time slots. For the cross-layer congestion control algorithm, we fixed the transmission rate at $x_s^* = e$ for the first 10^3 time slots while still updating the congestion prices and generating a new schedule in each time slot. This initial period provides time for the congestion prices to converge to "better" values. The step size is initialized as $\alpha = 0.1$ and it is updated as $\alpha = \max\{\alpha/t, 10^{-3}\}$.

We show the average utility achieved by the optimal transmission rate and by the congestion control algorithm in Figs. 13, 14 for one-hop and two-hop interference models, respectively. It is evident that the proposed approach achieves higher average utility then those using congestion control algorithm for all traffic load under both interference models. In fact, the average utility with congestion control decreases significantly when the traffic load is close to the boundary of the region which can be stabilized by the underlying scheduling algorithm. This observation confirms the argument that performing congestion control is unnecessary if the network can be stabilized by the underlying regulated scheduling algorithms.

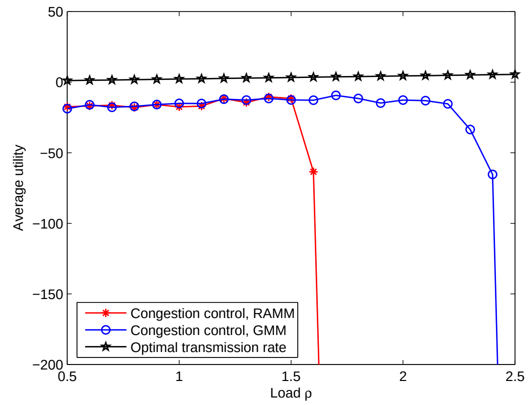

Fig. 14. Average utility of the proposed optimal transmission rate and with congestion control algorithm under two-hop interference model (for grid network with 36 nodes and 6 multihop flows, GMM and RAMM scheduling with B = 4, K = 11 in Fig. 5).

VIII. Conclusions

We investigated the network control problem using constant-time scheduling under the k-hop interference model. With flow dynamics consideration, we have shown that the network can be stabilized by using a regulated maximal scheduling policy if the offered load satisfies the constraining bound. We have presented a constant-time and distributed scheduling algorithm for a general k-hop interference model. The scheduling algorithm does not require queue length information and has overhead not growing with network size. Our proposed scheduling algorithm achieves performance arbitrarily close to that of the regulated maximal scheduling. Under the stability condition, we have derived optimal transmission rates which achieve the maximum long-term network utility. For the high load regime, we have proposed a cross-layer congestion control algorithm which can stabilize the network for any offered load. Numerical results have shown that the proposed scheduling algorithm achieves much better performance than the existing constant-time scheduling algorithms in the literature, and it has much better performance than its performance guarantee. Also, performing congestion control under low load condition actually degrades performance in terms of long-term utility significantly compared to the optimal transmission rate.

Appendix I
Proof of Proposition 1

Let $Q_l^s(t)$ and $Q_l(t)$ be transmission queue lengths for class s and for all user classes at link l in time slot t, respectively. Similarly, let $P_l^s(t)$ and $P_l(t)$ be regulator queue lengths for class s and for all user classes at link l in time slot t, respectively. Let us denote by b_l^s and a_l^s the previous link and next link of link l on the route of class-s users. Also, let $C_l^s(t)$ and $D_l^s(t)$ be the number of packets transmitted from regulator and transmission queues in time slot t, respectively. We have the following queue update equations

$$Q_l^s(t+1) = Q_l^s(t) - D_l^s(t) + C_l^s(t) \quad (14)$$

$$P_l^s(t+1) = P_l^s(t) - C_l^s(t) + D_{b_l^s}^s(t). \quad (15)$$

Thus, we have

$$Q_l^s(t+1)+P_{a_l^s}^s(t+1) = Q_l^s(t)+P_{a_l^s}^s(t)+C_l^s(t)-C_{a_l^s}^s(t). \quad (16)$$

We will use the following Lyapunov function for the system

$$V(\vec{P},\vec{Q}) = V_1(\vec{Q}) + \xi V_2(\vec{P},\vec{Q}) \quad (17)$$

where

$$V_1(\vec{Q}) = \sum_l \frac{Q_l(t)}{R_l} \sum_{k\in I_l} \frac{Q_k(t)}{R_k}$$

$$V_2(\vec{P},\vec{Q}) = \sum_l \sum_s \left(P_{a_l^s}^s + Q_l^s\right)^2.$$

In fact, this Lyapunov function was also used in [19]. Now, let us consider

$$\begin{aligned} & V_1(\vec{Q}(t+1)) - V_1(\vec{Q}(t)) \\ = \; & 2\sum_l \frac{Q_l(t)}{R_l} \left(\sum_{k\in I_l} \frac{(Q_k(t+1)-Q_k(t))}{R_k}\right) \\ + \; & \sum_l \left(\frac{(Q_l(t+1)-Q_l(t))}{R_l}\right) \\ & \times \left(\sum_{k\in I_l}\sum_s \frac{(Q_k(t+1)-Q_k(t))}{R_k}\right) \\ = \; & 2\sum_l \frac{Q_l(t)}{R_l}\left(\sum_{k\in I_l}\sum_s \frac{(C_k^s(t)-D_k^s(t))}{R_k}\right) \quad (18) \\ + \; & \sum_l \frac{(C_l^s(t)-D_l^s(t))}{R_l}\left(\sum_{k\in I_l}\sum_s \frac{(C_k^s(t)-D_k^s(t))}{R_k}\right) \quad (19) \end{aligned}$$

Since the number of packets transmitted from regulator and transmission queues in each time slot are bounded, the second term in (19) can be bounded by a constant B_1. Thus, we have

$$\begin{aligned} & E\left[V_1(\vec{Q}(t+1)) - V_1(\vec{Q}(t))|\vec{P}(t),\vec{Q}(t)\right] \\ & \leq 2E\left[\sum_l \frac{Q_l(t)}{R_l}\left(\sum_{k\in I_l}\sum_s \frac{(C_k^s(t)-D_k^s(t))}{R_k}\right) |\vec{P}(t),\vec{Q}(t)\right] + B_1 \\ & \leq 2E\left[\sum_{l:Q_l(t)\geq R_l} \frac{Q_l(t)}{R_l}\left(\sum_{k\in I_l}\sum_s \frac{C_k^s(t)}{R_k} - \sum_{k\in I_l}\frac{D_k(t)}{R_k}\right) |\vec{P}(t),\vec{Q}(t)\right] + B_2. \end{aligned}$$

Let L be the largest number of hops traversed by any user class, we have

$$\begin{aligned} & E\left[\sum_{l:Q_l(t)\geq R_l} \frac{Q_l(t)}{R_l}\sum_{k\in I_l}\sum_s \frac{C_k^s(t)}{R_k}|\vec{P}(t),\vec{Q}(t)\right] \\ & \leq \sum_{l:Q_l(t)\geq R_l}\frac{Q_l(t)}{R_l}\sum_{k\in I_l}\sum_s \frac{H_s^k\rho_s + L\epsilon}{R_k}. \quad (20) \end{aligned}$$

Also, due to the definition of maximal scheduling policy, we have

$$\sum_{l:Q_l(t)\geq R_l}\frac{Q_l(t)}{R_l}\sum_{k\in I_l}\frac{D_k(t)}{R_k} \geq \sum_{l:Q_l(t)\geq R_l}\frac{Q_l(t)}{R_l}. \quad (21)$$

For traffic load satisfying (5), we can find ϵ and δ small enough such that

$$\sum_{k\in I_l}\sum_{s=1}^{S}\frac{H_s^k\rho_s + L\epsilon + \delta}{R_k} \leq 1, \quad \forall l \in E. \quad (22)$$

From (20), (21), (22), we have

$$\begin{aligned} & E\left[V_1(\vec{Q}(t+1)) - V_1(\vec{Q}(t))|\vec{P}(t),\vec{Q}(t)\right] \\ & \leq -\delta\sum_{l:Q_l(t)\geq R_l}\frac{Q_l(t)}{R_l} + B_2 \leq -\delta\sum_l \frac{Q_l(t)}{R_l} + B_3. \quad (23) \end{aligned}$$

Now, using the procedure as in [19], we can obtain

$$\begin{aligned} & E\left[V_2(\vec{P}(t+1),\vec{Q}(t+1)) - V_2(\vec{P}(t),\vec{Q}(t))|\vec{P}(t),\vec{Q}(t)\right] \\ & \leq \sum_l\sum_s R_l Q_l(t) - \eta\sum_l\sum_s P_{a_l^s}^s(t) + B_4. \quad (24) \end{aligned}$$

Combining the results in (23), (24), we have

$$\begin{aligned} & E\left[V(\vec{P}(t+1),\vec{Q}(t+1)) - V(\vec{P}(t),\vec{Q}(t))|\vec{P}(t),\vec{Q}(t)\right] \\ & \leq -\sum_l\sum_s\left[\frac{\delta}{R_l} - \xi R_l\right]Q_l^s(t) - \xi\eta\sum_l\sum_s P_{a_l^s}^s(t) + B_5. \quad (25) \end{aligned}$$

We can choose ξ small enough such that $\frac{\delta}{R_l} - \xi R_l > 0$. Thus, the drift will be negative if the regulator and/or transmission queues become large enough. Therefore, the stability result follows by using theorem 2 of [26]. Note that the chosen Lyapunov function does not take regulator queue on the first hop of each user class into account. These regulator queues are, however, stable because their output rate is $\rho_s + \epsilon$ which is larger than the average input load (i.e., ρ_s).

Appendix II
Proof of Proposition 2

Consider any link AB between node A and B. We will find the probability that at least one link in the interference set I_{AB} is scheduled. This event will be denoted as M_{AB} in the sequel. As mentioned before, RAMM scheduling algorithm includes BP-SIM scheduling algorithm for the one-hop (node exclusive) interference model proposed in [17] as a special case. In the following we prove proposition 2 for $k \geq 2$. For the special case of $k = 1$, we refer the readers to [17] for the proof and the corresponding analysis. Note that the proof for the case $k \geq 2$ is very challenging and completely different from that for the case $k = 1$ in [17] due to the more complicated interference relationship.

We will illustrate some important definitions used in the proof in Fig. 15. Let I_0 be the set of nodes which is at most $k-1$ hops away from either A and B including A, B. For the grid network and link AB shown in Fig. 15 under the two-hop interference model, $I_0 = \{A, B, C_1, \cdots, C_6\}$. Also, the interference set for node C_6 (i.e., I_{C_6}) consists of all nodes in I_0 and all "blank" nodes. Note that the notion of node interference set is different from that of link interference set provided in definition I. Here, all nodes in I_{C_6} are at most 3 hops away from C_6. We observe that all links incident to any nodes in I_0 will belong to I_{AB} because they are within k hops from link AB. In the following, we will find the lower bound for the probability of M_{AB} by considering sub-cases in which there are i left nodes in the set I_0. For convenience, we will use I_0 to denote both the set itself and the corresponding number of nodes in I_0. Let L_i be the event that there are i left nodes in the set I_0. The probability that at least one link in the interference set of link AB is scheduled can be lower bounded as

$$P_m \geq \sum_{i=1}^{I_0} \Pr(M_{AB}|L_i)\Pr(L_i). \tag{26}$$

Recall that for any particular node A, there are at most I nodes whose matching request can collide with that transmitted from node A with the power used in the scheduling phase. Note that these I nodes will be at most $k+1$ hops away from mode A for the interference models with $k \leq 2$ where the same transmission power level is used in both scheduling and transmission phases of each time slot. To find the lower bound for the probability of M_{AB}, we will assume the worst-case scenario where each node has I interfering nodes. For convenience, we also use I_A to denote the set of these interfering nodes whose transmissions can collide with that from node A. We now consider the following cases.

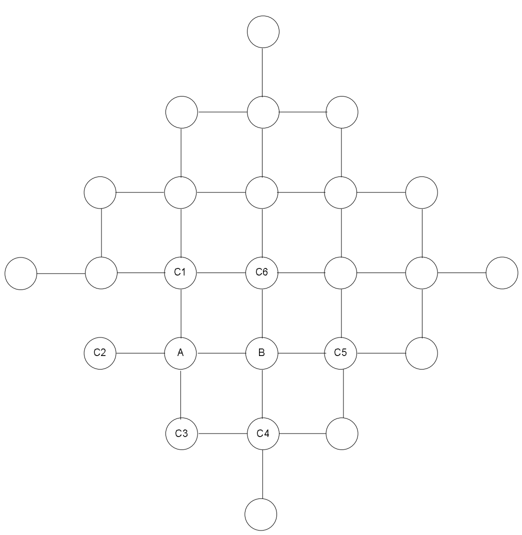

Fig. 15. Interference set I_{C_6} under two-hop interference model.

A. There is only one left node in I_0

We consider the following two sub-cases.

- If this left node is either A or B, then the left node will have at least one neighbor which is a right node. This case occurs with probability $2(1/2)^{I_0}$. For ease of reference, we will refer to this left node as node C (i.e., C is either A or B). Also, there are at most $I - I_0$ nodes whose matching requests can collide with that from node C. To find the lower bound for $\Pr(M_{AB})$, we assume that there are $I - I_0$ such interfering nodes.
  Now, suppose there are i left nodes among these $I - I_0$ interfering nodes. The matching request transmitted by node C will be successfully received if the backoff values of these i left nodes are larger than that of node C. Specifically, the matching request from node C will be successfully received and the corresponding link will be scheduled with a probability which is lower bounded by
  $$F_1 = \sum_{i=0}^{I-I_0} \begin{pmatrix} I-I_0 \\ i \end{pmatrix} \left(\frac{1}{2}\right)^{I-I_0} \frac{1}{B} \sum_{m=1}^{B} \left(1-\frac{m}{B}\right)^i$$
  where we have broken the event into sub-cases where there are i left nodes among $I - I_0$ interfering nodes and these i left nodes have backoff values larger than that of node C.
- If this left node is any node other than A and B then it can be any node among $(I_0 - 2)$ nodes. Again, we refer to this left node as node C. Note that all nodes which are within one hop from A or B including A and B are at most $k+1$ hops away from C so they all belong to the interfering set I_C. Let x be the total number of nodes within one hop from A and B including A and B, then there are at most $I - x$ nodes whose matching requests can collide with that from node C. This is because these x nodes belong to I_0 and they are all right nodes except C if C is in I_0. We will assume this worst-case scenario to calculate the lower bound of the matching probability in the following. Note that node C will have at least one neighbor which is a right node because it is the only left node in I_0.
  Again, suppose there are i left nodes among potential interfering nodes in I_C. The matching request transmitted by node C will be successfully received if the backoff values of these i left nodes are larger than that from node C and the matching request from node C is transmitted to a right node. Now, we consider the following two sub-cases. For the first case, if C is one of x nodes (i.e., x one-hop neighbors of A or B) but not A and B. Then, this case occurs with probability $(x-2)(1/2)^{I_0}$. In this case, the matching request from node C will be successfully received and the corresponding link will be scheduled with a probability which is lower bounded by
  $$F_2 = \sum_{i=0}^{I-x-1} \begin{pmatrix} I-x-1 \\ i \end{pmatrix} \left(\frac{1}{2}\right)^{I-x} \frac{1}{B} \sum_{m=1}^{B} \left(1-\frac{m}{B}\right)^i$$
  where i is the number of left node. Also, in calculating the lower bound for P_m we assume that C transmits its

matching request to a right node which is not an one-hop neighbor of A or B. Hence, there are at most $I-x-1$ left nodes which can collide with the matching request from C. For the second case, if C is not one of x nodes (i.e., x one-hop neighbors of A or B). Then, this case occurs with probability $(I_0-x)(1/2)^{I_0}$. In this case, the matching request from node C will be successfully received and the corresponding link will be scheduled with a probability which is lower bounded by

$$F_3 = \sum_{i=0}^{I-x-2} \left(\begin{array}{c} I-x-2 \\ i \end{array} \right) \left(\frac{1}{2}\right)^{I-x} \frac{1}{B} \sum_{m=1}^{B} \left(1-\frac{m}{B}\right)^{i}$$

where in calculating the lower bound for P_m we assume that C transmits its matching request to a right node which is not an one-hop neighbor of A or B. Hence, there are at most $I-x-2$ left nodes which can collide with the matching request from C.

B. There are two or more left nodes in I_0

Suppose node C in the set I_0 becomes left and wins the contention. Then node C should have the smallest backoff value among all the nodes whose matching requests can collide with the matching request from C. Also, node C should send the matching request to a node which is a right one. For ease of reference, we will refer to this right node as node D in the sequel. In general, D can belong to set I_0 or not. However, to find the lower bound of P_m, we assume that D belongs to I_0; therefore, there are at most I_0-2 other left nodes besides C and D in I_0.

As before, we assume the worst-case scenario where there are I nodes whose transmissions can collide with that of node C. Recall that all x nodes which are one-hop neighbors of A or B belong to the set I. Similar to the previous case, we consider the following two sub-cases. For the first case, C is one of x nodes (i.e., x one-hop neighbors of A or B). In this case the matching probability can be lower bounded as

$$F_4 = x \sum_{i=1}^{I_0-2} \sum_{j=0}^{I-x-1} \left(\begin{array}{c} I_0-2 \\ i \end{array} \right) \left(\frac{1}{2}\right)^{I_0} \left(\begin{array}{c} I-x-1 \\ j \end{array} \right) \times \left(\frac{1}{2}\right)^{I-x} \frac{1}{B} \sum_{m=1}^{B} \left(1-\frac{m}{B}\right)^{i+j}$$

where i is the number of left nodes besides C and D in the set I_0. And j is the number of left nodes which belong to I but are not one-hop neighbors of A or B (i.e., there is no link between these nodes and A or B). We will denote this set as $I \setminus x$ in the sequel. In general, D can belong to $I \setminus x$ or not; however, to find the lower bound for P_m, we only allow j takes values from 0 to $I-x-1$. In addition, C can be any node among x nodes so we have a factor of x before the sum. Also, we require that all left nodes (i left nodes belonging I_0 and j left nodes belonging to $I \setminus x$) achieve larger backoff values than that of node C.

For the second case, C belong to the set $I \setminus x$. In this case, the matching probability can be lower bounded as

$$F_5 = (I_0-x) \sum_{i=1}^{I_0-2} \sum_{j=0}^{I-x-2} \left(\begin{array}{c} I_0-2 \\ i \end{array} \right) \left(\frac{1}{2}\right)^{I_0} \left(\begin{array}{c} I-x-2 \\ j \end{array} \right) \times \left(\frac{1}{2}\right)^{I-x} \frac{1}{B} \sum_{m=1}^{B} \left(1-\frac{m}{B}\right)^{i+j}$$

where C can be one of I_0-x nodes so we have the factor (I_0-x) before the sum. Also, to calculate the lower bound for the matching probability, we assume D always belong to $I \setminus x$, so j can be at most $I-x-2$.

Substitute results of all considered cases into (26), the matching probability is lower bounded by

$$P_m \geq P_m^0 = 2(1/2)^{I_0} F_1 + (x-2)(1/2)^{I_0} F_2 + (I_0-x)(1/2)^{I_0} F_3 + F_4 + F5 \quad (27)$$

where F_1, F_2, F_3, F_4, and F_5 are defined above.

From the lower bound of the matching probability P_m^0 derived above, we can calculate the lower bound of P_m for any link AB as

$$p^* = \min_{x,\, I_0} P_m^0$$

where we find the minimum of P_m^0 over all possible x and I_0. Note that possible values of x and I_0 will be in the range of $[3, 2d^*]$ and $[3, I_0^*]$, respectively. Here, x and I_0 are at least three for the network to be connected (i.e., A and B should have at least one one-hop neighbor). It is observed that p^* is independent of the network size and depends only on B, d^*, I, I_0^*. Now, we can choose K such that at least one link in I_l of a backlogged link l is scheduled with probability greater than μ after K rounds as follows:

$$K = \min_{k \geq 1} \left\{ k : (1-p^*)^k \leq 1-\mu \right\}. \quad (28)$$

Thus, we can choose K which is independent of network size and only depends on B, d^*, I, I_0^*, μ such that performance guarantee arbitrarily close to the constraining bound can be achieved.

Example: For the grid network and two-hop interference model, we have I = 22, $I_0 = x$ can take values of 5, 6, 8. With maximum backoff value B = 10, by using the analysis presented above, the minimum number of scheduling rounds to achieve μ = 0.9 is K = 60. In fact, this calculation is quite conservative because it considers the worst case scenario. In practice, the size of interference sets decreases quickly over scheduling rounds, so the required value of K is much smaller.

Appendix III
Proof of Proposition 4

The proof is similar to that of Proposition 1 (i.e., we use the same Lyapunov function and proof procedure). In particular, we have a similar bound as in (20) as follows:

$$E\left[\sum_{l:Q_l(t)\geq R_l}\frac{Q_l(t)}{R_l}\sum_{k\in I_l}\sum_{s}\frac{C_k^s(t)}{R_k}|\vec{P}(t),\vec{Q}(t)\right] \leq \sum_{l:Q_l(t)\geq R_l}\frac{Q_l(t)}{R_l}\sum_{k\in I_l}\sum_{s}\frac{H_s^k x_s^* + L\epsilon}{R_k}. \quad (29)$$

Due to the constraints of the optimization problem in (10), we can find ϵ and δ small enough such that

$$\sum_{k\in I_l}\sum_{s=1}^{S}\frac{H_s^k x_s^* + L\epsilon + \delta}{R_k} \leq 1, \quad \forall l \in E. \quad (30)$$

From (29) and (30), we have

$$E\left[V_1(\vec{Q}(t+1)) - V_1(\vec{Q}(t))|\vec{P}(t),\vec{Q}(t)\right] \leq -\delta\sum_{l:Q_l(t)\geq R_l}\frac{Q_l(t)}{R_l} + B_6 \leq -\delta\sum_{l}\frac{Q_l(t)}{R_l} + B_7. \quad (31)$$

The remaining steps to obtain negative drift when regulator and/or transmission queues become large enough are the same as in the proof of Proposition 1.

References

[1] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Networking*, vol. 8, no. 5, pp. 556-567, Oct. 2000.

[2] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *Journal of the Operation Research Society,* vol. 49, pp. 237-252, 1998.

[3] S. H. Low and D. E. Lapsley, "Optimal flow control, I: Basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, pp. 861-875, Dec. 1999.

[4] H. Yaiche, R. Mazumdar, and C. Rosenberg, "A game theoretic framework for bandwidth allocation and pricing in broadband networks," *IEEE/ACM Transactions on Networking*, vol. 8, no.5, pp. 667-678, 2000.

[5] 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.

[6] M. Neely, E. Modiano, and C. Li, "Fairness and optimal stochastic control for heterogeneous networks," *IEEE INFOCOM 2005*.

[7] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *IEEE Transactions on Information Theory*, vol. 34, pp. 910-917, 1988.

[8] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint asynchronous congestion control and distributed scheduling for multi-hop wireless networks," *IEEE INFOCOM 2006*.

[9] G. Sharma, R. Mazumdar, and N. Shroff, "On the complexity of scheduling in multihop wireless systems," *IEEE MOBICOM 2006*.

[10] 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, Dec. 1992.

[11] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches, " *IEEE INFOCOM 1998*.

[12] E. Modiano, D. Shah, and G. Zussman, "Maximizing throughput in wireless networks via gossiping, *ACM SIGMETRICS 2006*.

[13] A. Eryilmaz, A. Ozdaglar, and E. Modiano, "Polynomial complexity algorithms for full utilization of multi-hop wireless networks," *IEEE INFOCOM 2007*.

[14] S. Sanghavi, L. Bui and R. Srikant, "Distributed link scheduling with constant overhead," *ACM SIGMETRICS 2007*.

[15] X. Lin and S. Rasool, "Constant-time distributed scheduling policies for ad hoc wireless networks," *IEEE Conference on Decision and Control,* Dec. 2006.

[16] C. Joo and N. B. Shroff, "Performance of random access scheduling schemes in multi-hop wireless networks," *IEEE INFOCOM 2007*.

[17] A. Gupta, X. Lin and R. Srikant, "Low-complexity distributed scheduling algorithms for wireless networks," *IEEE INFOCOM 2007*.

[18] X. Wu and R. Srikant, "Bounds on the capacity region of multi-hop wireless networks under distributed greedy scheduling," *IEEE INFOCOM 2006*.

[19] X. Wu, R. Srikant, and J. R. Perkins, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 595-605, June 2007.

[20] P. Chaporkar, K. Kar, S. Sarkar, "Throughput guarantees through maximal scheduling in wireless networks," *Allerton Conference on Communication, Control and Computing,* Sept. 2005.

[21] G. Sharma, R. Mazumdar, and N. Shroff, "Joint congestion control and distributed scheduling for throughput guarantees in wireless networks," *IEEE INFOCOM 2007*.

[22] C. Humes, Jr., "A regulator stabilization technique: Kumar-Seidman revisited," *IEEE Transactions on Automatic Control*, vol. 39, no. 1, pp. 191-196, Jan. 1994.

[23] P. R. Kumar and T. I. Seidman, "Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems," *IEEE Transactions on Automatic Control*, vol. 35, no. 3, pp. 289-298, Mar. 1990.

[24] J. Zhang, D. Zheng, and M. Chiang, "The impact of stochastic noisy feedback on distributed network utility maximization," *IEEE INFOCOM 2007*.

[25] K. Ma, R. R. Mazumdar, and J. Luo, "On the performance of primal/dual schemes for congestion control in networks with dynamic flows," *IEEE INFOCOM 2008*.

[26] M. Neely, E. Modiano, and C. Rohrs, "Power allocation and routing in multibeam satellites with time-varying channels," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 138-152, Feb. 2003.

Long Le received the B.Eng. degree with highest distinction from Ho Chi Minh City University of Technology in 1999, the M.Eng. degree from Asian Institute of Technology (AIT) in 2002, and the Ph.D. degree from University of Manitoba in 2007. He is currently a Postdoctoral Fellow in the Department of Electrical and Computer Engineering at University of Waterloo. His current research interests include cognitive radio, cooperative diversity and relay networks, and cross-layer design for communication networks. He is a Member of the IEEE.

PLACE PHOTO HERE

Ravi Mazumdar was born in Bangalore, India. He obtained the B.Tech. in Electrical Engineering from the Indian Institute of Technology, Bombay, India in 1977, the M.Sc. DIC in Control Systems from Imperial College, London, U.K. in 1978 and the Ph.D. in Systems Science from the University of California, Los Angeles, USA in 1983. He is currently a University Research Chair Professor of Electrical and Computer Engineering at the University of Waterloo, Waterloo, Canada and an Adjunct Professor of ECE at Purdue University. He has served on the faculties of Columbia University (NY, USA), INRS-Telecommunications (Montreal, Canada) , University of Essex (Colchester, UK), and most recently at Purdue University (West Lafayette, USA). He has held visiting positions and sabbatical leaves at UCLA, the University of Twente (Netherlands), the Indian Institute of Science (Bangalore); and the Ecole Nationale Superieure desTelecommunications (Paris). He is a Fellow of the IEEE and the Royal Statistical Society. He is a member of the working groups WG6.3 and 7.1 of the IFIP and a member of SIAM and the IMS. He shared the Best Paper Award with G. Sharma and N. Shroff at the INFOCOM 2006 and was co-author with N. Likhanov for the Best Paper runner up at INFOCOM 1998. He is an Editor of the IEEE Trans. on Networking and has served as a guest editor for a number of journals. His research interests are in wireless and wireline networks; applications of game theory to networking; applied probability, queueing theory, and stochastic analysis with applications to traffic engineering, stochastic filtering theory, and mathematical finance.