A Generic Framework for Throughput-Optimal Control in MR-MC Wireless Networks

Hongkun Li, Yu Cheng
Department of Electrical and Computer Engineering
Illinois Institute of Technology
{hli55, cheng}@iit.edu

Xiaohua Tian and Xinbing Wang
Department of Electronic Engineering
Shanghai Jiao Tong University
{xtian, xwang8}@sjtu.edu.cn

***Abstract*—In this paper, we study the throughput-optimal control in the multi-radio multi-channel (MR-MC) wireless networks, which is particularly challenging due to the coupled link scheduling and channel/radio assignment. This paper has three-fold contributions: 1) We develop a new model by transforming a network node into multiple node-radio-channel (NRC) tuples. Such modeling facilitates the development of a *tuple-based back pressure algorithm*, the solution of which can jointly solve the link scheduling, routing and channel/radio assignment in the MR-MC network. 2) The tuple-based model enables the extensions of some well-known algorithms, e.g., greedy maximal scheduling and maximal scheduling, to MR-MC networks with guaranteed performance. We provide stability and capacity efficiency ratio analysis to the tuple-based scheduling algorithms. 3) The tuple-based framework facilitates a decomposable cross-layer formulation that enhances the delay performance of throughput-optimal control by integrating the link-layer scheduling with the network-layer path selection, where both hop-count and queuing delay are considered. Simulation results are presented to demonstrate the capacity region and delay performance of the proposed methodology, with comparison to the existing approach [3].**

I. Introduction

The multi-radio multi-channel (MR-MC) wireless networks have received substantial attention in the last few years [2], [5]. Such a networking model can significantly increase the network capacity by simultaneously exploiting multiple non-overlapping channels through different radio interfaces and mitigating interferences through proper channel assignment. However, multiple radios and channels lead to more complex network management such as the throughput-optimal control. The throughput-optimal schedule has been extensively studied [1], [7], [10] in the single-radio single-channel (SR-SC) context, which achieves the optimal capacity region while maintaining the network stable. The throughput-optimal scheduling relies on the *back-pressure* algorithm to select links for transmission [7], [12]–[14]. The MR-MC networks however induce significant challenges to the throughput-optimal network control, because the link scheduling is coupled with radio/channel assignment and the link capacity is normally channel dependent [3]. Currently, there are only a couple of heuristic studies on throughput-optimal control in the MR-MC context [3], [4].

This work was supported in part by NSF grants CNS-1053777 and CNS-0832093.

In this paper, we systematically study the throughput-optimal control in the MR-MC networks by developing a new modeling framework, which transforms a network node into multiple *node-radio-channel* (NRC) tuples, so that each communication link is mapped to multiple *tuple links* connecting the sending/receiving tuples. Consequently, an MR-MC network is transformed to an equivalent tuple-based network, where all the coupled resource allocation issues in throughput-optimal control can then be reduced to a tuple-based back-pressure algorithm. The tuple-based model has significant theoretical advantages compared to the existing link-based methods [3], [4]. It not only facilitates the formulation of throughput-optimal control in the MR-MC network, but also provides an opportunity to improve the delay performance of the back-pressure algorithm through a decomposable cross-layer structure. Such decomposable control is infeasible in the existing methods.

The back-pressure algorithm (essentially the maximum weighted independent set (MWIS) problem) is unfortunately NP-hard [1]. To reduce the high computational complexity, greedy algorithms have been extensively studied in recent years but mainly in the SR-SC context, referring to [3], [19] and the references therein, which normally have to sacrifice a fraction of the capacity region for the reduced complexity. It is however hard to directly extend the greedy algorithms developed in the SR-SC context to the MR-MC networks, again due to the coupled issues of link scheduling, channel and radio assignment [3], [4]. In this paper, we for the first time reveal the insight that *our tuple-based network model provides a virtual SR-SC environment, where those well-known polynomial approximate algorithms developed in the SR-SC context with guaranteed capacity region can now be applied over the tuple-based model*. We particularly discuss the extension of the greedy maximal scheduling [7], [16] and maximal scheduling [7], [13], [14] algorithms, and theoretically analyze the stability and capacity efficiency ratio of these two algorithms.

It is well-known that the classic back-pressure algorithm may lead to unnecessarily large delays [15], [19]. The tuple-based network model has the particular advantage in developing a cross-layer framework to enhance the delay performance of the back-pressure algorithm, which integrates the link-layer scheduling with the network-layer path selection taking both

the transmission delay and queuing delay into consideration. Furthermore, our cross-layer framework has a nice decomposition property for convenient implementation. Note that such a decomposable cross-layer framework is infeasible for the existing link-based scheduling formulations that interleave the link scheduling and channel/radio assignment together. The work in [3] did leverage the path selection to enhance the delay performance, but can not fully decouple link scheduling from channel/radio assignment and thus suffer from underexplored performance.

The reminder of this paper is organized as follows. Section II reviews more related works. Section III overviews the most popular link-based model in this area. Section IV presents the system model. Section V develops the tuple-based framework for throughput-optimal control. Section VI studies the tuple-based scheduling algorithms. A cross-layer framework is provided in Section VII to improve the delay performance. Section VIII shows some simulation results for performance evaluation. Section IX gives the concluding remarks.

II. Related Work

The throughput-optimal control for SR-SC wireless networks has been extensively studied. The classic back-pressure algorithm is first developed in [1] to achieve the optimal capacity region, which is NP-hard however. As a low-complexity approximation, the greedy algorithms have been comprehensively studied in terms of throughput [12]–[14] and delay [6], [15].

There is few study on the throughput-optimal control and practical scheduling design in the MR-MC networks. Lin and Rasool [3] propose a heuristic distributed scheduling algorithm to deal with the coupled resource allocation issues in MR-MC networks, which is the only significant reference known to us. The work in [3] however does not provide general guidance on how to design scheduling algorithms with provable performance in MR-MC networks. The study in [4] formulates a cross-layer optimization framework to jointly solve the resource allocation issues up to transport-layer congestion control, but it relies on the scheduling algorithm in [3] as lower-layer solutions. A polynomial-time approximation scheme (PTAS) framework of scheduling in MR-MC network is developed in [17] by dividing the whole area into several sub-areas to reduce the complexity; but no practical scheduling algorithm is presented either.

Optimizing a MR-MC networks by nature leads to a mixed integer programming (which is NP-hard), involving binary variables to describe channel assignment and radio interface allocation constraints [2], [5]. Our work [9] develops a novel tool of multi-dimensional conflict graph (MDCG), which enables a linear programming framework for optimizing MR-MC networks, based on the maximal independent set (MIS) based scheduling. The tuple-based transformation in this paper is inspired by the MDCG-based model, which facilitates the construction of a virtual SR-SC context for a systematic study of efficient throughput-optimal control algorithms.

It is known that the classic back-pressure algorithm could lead to unnecessarily large delays [15], [19]. Some recent studies pay attention to the delay performance of wireless scheduling policies [3], [6], [8], [15]. A particular interesting approach is to integrate the shortest path selection with the scheduling [15]. To facilitate the network protocol design, it is preferred to form a cross-layer control framework with a decomposable structure, which is impossible if the scheduling formulation interleaves all resource allocation issues [3], [4]. The proposed tuple-based network model allows a decomposable cross-layer framework to enhance the delay performance of the throughput-optimal scheduling in MR-MC networks.

III. Link-Based Method

In this section, we summarize the link-based model for developing the scheduling algorithm in the MR-MC networks [3], where the link scheduling and channel/radio assignment are interleaved. To handle the channel-diversity issue [3], each link l needs to maintain two-stage queues: q_l for the incoming packets and η_l^c for the packets assigned to channel c from q_l. Therefore, the packets arriving to link l are served in two steps. Given a set of alternate paths, an optimization problem is formulated for path selection (equation (14) in [3]):

$$\max_{\vec{P}_s}: \ -\frac{\beta_s}{2}\sum_{j=1}^{J(s)}(P_{sj})^2-\sum_{j=1}^{J(s)}P_{sj}\sum_{l=1}^{L}H_{sj}^{l}q_l(t)$$

where $J(s)$ is the number of paths for flow s, and P_{sj} denotes the fraction of input traffic injected into path j for flow s. H_{sj}^{l} is the routing matrix, and each element in the matrix indicates whether link l is on the path j of flow s. The routing decision is made based on the input queue q_l, but the schedule set is determined according to the channel queue η_l^c [3]. Specifically, the packets in queue q_l are assigned to an output queue η_l^c as (equation (2) in [3]):

$$x_l^c=r_l^c,\ \text{if}\ \frac{q_l}{\alpha_l}\geq\frac{1}{r_l^c}\Bigg\{\sum_{k\in I(l)}\frac{\eta_k^c}{r_k^c}+\frac{1}{M_{b(l)}}\sum_{k\in E(b(l))}\sum_{d=1}^{C}\frac{\eta_d^c}{r_d^c}+\frac{1}{M_{e(l)}}\sum_{k\in E(e(l))}\sum_{d=1}^{C}\frac{\eta_d^c}{r_d^c}\Bigg\}$$

where x_l^c is the number of packets that link l can assign to channel c at time slot t. The scheduling decision is then based on the queue length information η_l^c.

We can see that the two-stage design only partially presents the input backlog information to the network layer, but inhibits the output queues from being considered by the path selection. Thus the delay evaluation in the path selection algorithms in [3] is inaccurate. Moreover, the method in [3] provides only heuristic algorithm to compute the schedule set. In this paper, we develop a tuple-based model, which has significant advantages in three aspects. 1) The tuple-based model completely decouples the link scheduling and channel/radio assignment, therefore we can solve the joint resource allocation problem in the MR-MC context and achieve the throughput-optimal scheduling. 2) The tuple-based model

facilitates the direct extension of low-complexity scheduling in the MR-MC context, which is impossible under the link-based model due to the channel-dependent link capacity. 3) The nice decomposable property allows the delay enhancement of back pressure algorithm through a cross-layer formulation with more accurate delay evaluation.

IV. System Model

Consider the original network G with node set $\mathcal{N}$ and link set $\mathcal{L}$. Node n is equipped with M_n radio interfaces. There are C available channels.

A. Generating Tuple-Based Network

In the new model, node n is transformed to multiple *node-radio-channel (NRC) tuples*, denoted as (n, m_n, c), where $m_n = 1, \ldots, M_n\}$ and $\{c = 1, \ldots, C\}$. The tuple (n, m_n, c) means that the m_nth radio of node n is working on the channel c. Therefore, node n is mapped to $M_n \times C$ tuples. Let $n(p)$ denote the node which generates the tuple p. According to NRC-tuple transformation, the original link (i, j) is mapped to a couple of *tuple links* connecting two tuples (e.g., the tuple link $((i, m_i, c), (j, m_j, c))$). Two tuples form a tuple link if their associated nodes form an original link and they are working on the same channel. We refer the whole tuple set as $\mathcal{P}$, and the tuple link set $\mathcal{L}_\mathcal{P}$. For convenience of presentation, we use l_p or (p, p') to denote a tuple link.

The physical capacity of a tuple link is set the same as the bandwidth of the corresponding original link, since the physical link capacity is independent of the radio interface, we use $w_{l_p}^{c(l_p)}$ to indicate the capacity of tuple link l_p, and $c(l_p)$ is the channel assigned to l_p. In the tuple-based network, a packet may enter and leave a node through different radios or channels. Let $\mu_{p,p'}^f$ is the flow rate from tuple p to p' for flow f within a node. There is no bandwidth constraint on the variable $\mu_{p,p'}^f$. An example shown in Fig. 1 illustrates how to transform an original link to a set of tuple links. The dotted line denotes the flow transition within a node and solid line is the tuple link. We plot only a part of flow transition within a node for clear illustration.

Consider F commodity flows over the network with the input rate vector $\vec{\lambda} = (\lambda^1, \ldots, \lambda^f, \ldots \lambda^F)$. Let $r_{l_p}^f$ denote the traffic loaded to tuple link l_p for flow f. All such link flow loads constitute a link flow vector $\vec{r}$. The *capacity region* under a particular scheduling algorithm is the set of $\vec{r}$ such that the system remains stable. The *optimal capacity region* Ω is defined as the supremum of the capacity regions of all algorithms. Let $s(f)$ and $d(f)$ denote the source and destination nodes of flow f, respectively. Let p_s denote the tuples associated with a source node s, and $\lambda_{p_s}^f$ denote the input rate of flow f from a source tuple p_s. Thus we have $\lambda^f = \sum_{p_s : n(p_s) = s(f)} \lambda_{p_s}^f$. Table I summaries the main notations used in the paper.

B. Interference Model

The interference relation in this paper is determined according to the protocol model [11]. In the MR-MC network, both co-channel contention and radio interface competence would result in the interference. We define the following events to determine the interference relation between any two tuple links, denoted as (p_1, p_2) and (p_3, p_4):

- Radio interface contention: any two tuples in the pair (p_1, p_3), (p_1, p_4), (p_2, p_3) or (p_2, p_4) share a common radio interface.
- Co-channel transmissions within the interference range: the original links $l(p_1, p_2)$ and $l(p_3, p_4)$ interfere with each other, and the two tuple links work on the same channel.

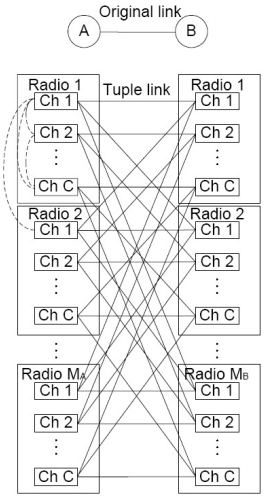


Fig. 1. Generating the tuple links.

If either of the above statements is true, two tuple links are interfering with each other. The first condition presents the radio interface contention, which is a special case in MR-MC networks. The second one implies the co-channel interference.

We further define the conflict area $I(l_p)$ of tuple link l_p, which contains all tuple links that interfere with l_p. For convenience, we assume tuple link l_p itself contained in $I(l_p)$ too. A non-interfering subset of $I(l_p)$ includes some tuple links that do not interfere with each other. Let ξ_{l_p} denote the maximum cardinality of any non-interfering subset of $I(l_p)$. The *conflict degree* ξ is the maximum ξ_{l_p} over all tuple links in $\mathcal{L}_\mathcal{P}$. We will reveal the relation between the conflict degree ξ in the MR-MC context and the interference degree $\mathcal{K}$ in the SR-SC context [13] in Section VI.

V. Tuple-Based Throughput-Optimal Control

A. Problem Formulation

The utility function of each commodity is $U(\lambda^f)$. Our goal is to maximize the total utility over all flows. For convenient use of convex optimization, the utility function is assumed to be strictly concave. The resource allocation problem contains a set of sub-problems:

- congestion control and traffic distribution $\vec{\lambda}$, $\vec{\lambda}_p$
- routing and scheduling $r_{l_p}^f$
- channel/raido assignment $r_{l_p}^f$, $\mu_{p,p'}^f$

TABLE I
MAIN NOTATIONS.

$\mathcal{P}$	The tuple set
$\mathcal{L}_{\mathcal{P}}$	The tuple link set
C	The number of available channels
$\vec{\lambda}$	The arrival rate vector at all the source nodes
$\vec{\lambda}_p$	The arrival rate vector at all tuples associated with the source nodes
F	The set of flow commodities
Q_p^f	The queue length of tuple p for flow f
$r_{l_p}^f$	The rate of flow f over tuple link l_p
$\mu_{p,p'}^f$	The transition rate of flow f from tuple p to p' within a node
$q_{l_p}^f$	The queue length of tuple link l_p for flow f
$n(p)$	The node which generates tuple p
$c(l_p)$	The channel assigned to tuple link l_p
$w_{l_p}^{c(l_p)}$	Physical capacity of tuple link l_p over channel $c(l_p)$
$I(l_p)$	The set of tuple links interfering with l_p
ξ	The conflict degree defined over an MDCG

We formulate the following optimization problem (**P1**) to solve the joint resource allocation:

$$\max_{\vec{r},\vec{\lambda},\vec{\lambda}_p,\vec{\mu}} \quad \sum_f U(\lambda^f) \tag{1}$$

Subject to:

$$\sum_{p':(p',p)\in\mathcal{L}_{\mathcal{P}}} r_{(p',p)}^f + \sum_{p'} \mu_{p',p}^f \le \sum_{p'':(p,p'')\in\mathcal{L}_{\mathcal{P}}} r_{(p,p'')}^f + \sum_{p''} \mu_{p,p''}^f \quad \forall f \in F,\ p \in \mathcal{P}(p \neq p_s) \tag{2}$$

$$\sum_{p':(p',p_s)\in\mathcal{L}_{\mathcal{P}}} r_{(p',p_s)}^f + \sum_{p'} \mu_{p',p_s}^f + \lambda_{p_s}^f \le \sum_{p'':(p_s,p'')\in\mathcal{L}_{\mathcal{P}}} r_{(p_s,p'')}^f + \sum_{p''} \mu_{p_s,p''}^f \quad \forall f \in F,\ \ p_s \in \mathcal{P} \tag{3}$$

$$\sum_{p':n(p')=n(p)} \sum_f \mu_{p',p}^f \le \max_{p'':(p,p'')\in\mathcal{L}_{\mathcal{P}}} w_{(p,p'')}^{c(p,p'')},\ \forall p \in \mathcal{P} \tag{4}$$

$$r_{l_p}^f \ge 0,\ \forall f \in F,\ l_p \in \mathcal{L}_{\mathcal{P}} \tag{5}$$

$$\mu_{p,p'}^f \ge 0,\ \forall f \in F,\ n(p) = n(p') \tag{6}$$

$$\lambda^f = \sum_{p_s:n(p_s)=s(f)} \lambda_{p_s}^f,\ \forall f \in F \tag{7}$$

$$\vec{r} \in Co(\Gamma) \tag{8}$$

where $\vec{\lambda}$, $\vec{\lambda_p}$, $\vec{\mu}$ and $\vec{r}$ are the vector forms of optimization variables. The constraints (2) and (3) state the flow conservation at intermediate tuples and source tuples. The link capacity constraints are presented by (4) to (6). Specifically, condition (4) imposes a constraint on the in-node transition rates. The in-node transition rates should be large enough not to negatively impact the capacity region of the original network. We thus set an upper bound with the understanding that the total in-node transition rate to a tuple should not exceed the maximum possible output capacity for that tuple. Constraint (7) describes the arrival rate relation between the source node and the associated tuples. Constraint (8) indicates the feasible capacity region for the link flow vector $\vec{r}$, where Γ is the set of all possible link schedules and $Co(\Gamma)$ is the convex hull of the link schedules. The constraints are all in linear form, and the utility function is assumed strictly concave, therefore **P1** is a convex optimization problem. In the following, we apply the dual decomposition technique to solve **P1**.

B. Dual Decomposition

We introduce Q_p^f and $Q_{p_s}^f$ as Lagrange multipliers associated with constraints (2) and (3) as well as q^f associated with constraint (7). Then, we can derive the partial Lagrange dual function as follows:

$$\begin{aligned} L(Q,q) = \max_{\vec{\lambda},\vec{r},\vec{\lambda}_p,\vec{\mu}} \Bigg\{ & \sum_f U(\lambda^f) + \sum_f q^f \Bigg(\sum_{p_s} \lambda_{p_s}^f - \lambda^f \Bigg) \\ & + \sum_{p,f} Q_p^f \Bigg(\sum_{p':(p,p')\in\mathcal{L}_{\mathcal{P}}} r_{(p,p')}^f + \sum_{p'} \mu_{p,p'}^f \\ & - \sum_{p'':(p'',p)\in\mathcal{L}_{\mathcal{P}}} r_{(p'',p)}^f - \sum_{p''} \mu_{p'',p}^f \Bigg) \\ & + \sum_{p_s,f} Q_{p_s}^f \Bigg(\sum_{p':(p_s,p')\in\mathcal{L}_{\mathcal{P}}} r_{(p_s,p')}^f + \sum_{p'} \mu_{p_s,p'}^f \\ & - \sum_{p'':(p'',p_s)\in\mathcal{L}_{\mathcal{P}}} r_{(p'',p_s)}^f - \sum_{p''} \mu_{p'',p_s}^f - \lambda_{p_s}^f \Bigg) \Bigg\} \end{aligned}$$

The Lagrange function can be rewritten:

$$L(Q,q) = \max_{\vec{\lambda}} \Bigg\{ \sum_f U(\lambda^f) - q^f \cdot \lambda^f \Bigg\} \tag{9}$$

$$+ \max_{\vec{\lambda}_p} \Bigg\{ \sum_{p_s,f} (q^f - Q_{p_s}^f) \cdot \lambda_{p_s}^f \Bigg\} \tag{10}$$

$$+ \max_{\vec{\mu}} \Bigg\{ \sum_{p,p',f} (Q_p^f - Q_{p'}^f) \cdot \mu_{p,p'}^f \Bigg\} \tag{11}$$

$$+ \max_{\vec{r}} \Bigg\{ \sum_{(p,p'),f} (Q_p^f - Q_{p'}^f) \cdot r_{(p,p')}^f \Bigg\} \tag{12}$$

To obtain (12), we applied the important property [19]: $\sum_{p,f} Q_p^f \Big(\sum_{p':(p,p')\in\mathcal{L}_{\mathcal{P}}} r_{(p,p')}^f - \sum_{p'':(p'',p)\in\mathcal{L}_{\mathcal{P}}} r_{(p'',p)}^f \Big) = \sum_{(p,p'),f} r_{(p,p')}^f [Q_p^f - Q_{p'}^f]$. It can be seen that the resource allocation problem is decomposed to a couple of sub-problems. 1) The maximization (9) implies the flow control with utility maximization, which determines the input rate at each source node based on the local backlog information; 2) The expression (10) solves the traffic distribution issue. Lagrange multiplier q^f can be interpreted as the queue length at the source node for flow f. The flow control is completed in two steps: the source node of flow f first independently computes the optimal arrival rate to maximize (9); then it picks the tuple with the largest value of $q^f - Q_{p_s}^f$, and set $\lambda_{p_s}^f = \lambda^f$. Both steps can be implemented in a distributed manner. 3) The expression (11) determines the flow transition among the tuples within a node, which implies that a packet may enter and leave a node through different radios and channels. All the above three problems can be computed locally. 4) The formulation (12)

solves the scheduling and routing problems, which requires the global information and centralized computation. To optimize (12), for the tuple link (p, p'), we define the flow f^* such that $f^* = \text{argmax}_f\{Q_p^f - Q_{p'}^f\}$, and set the $r_{(p,p')}^{f^*} = w_{(p,p')}^{c(p,p')}$ and $r_{(p,p')}^f = 0$ if $f \neq f^*$. Then the maximization

$$\max_{\vec{r}} \left\{ \sum_{(p,p') \in \mathcal{L}_\mathcal{P}} [Q_p^{f^*} - Q_{p'}^{f^*}]^+ \cdot r_{(p,p')}^{f^*} \right\} \tag{13}$$

is essentially the tuple-based back pressure algorithm. We can apply the classic sub-gradient method to compute the multipliers since the Lagrange function is convex. With the flow allocation vector $\vec{r}$ obtained, at the $(t+1)$th iteration, the Lagrange multipliers are updated by:

$$Q_p^f(t+1) = [Q_p^f(t) - \sum_{(p,p') \in \mathcal{L}_\mathcal{P}} r_{(p,p')}^f + \sum_{(p'',p) \in \mathcal{L}_\mathcal{P}} r_{(p'',p)}^f - \sum_{p':n(p)=n(p')} \mu_{p,p'}^f + \sum_{p'':n(p'')=n(p)} \mu_{p'',p}^f + \lambda_{p_s}^f \mathbf{1}_{\{p \in p_s\}}]^+ \tag{14}$$

$$q^f(t+1) = [q^f(t) - \sum_{p_s} \lambda_{p_s}^f + \lambda^f]^+ \tag{15}$$

where $\mathbf{1}_A$ is 1 if event A is true and 0 otherwise. The convex optimization theory tells that the iterative computations of (13) and (14) converge to the optimal flow allocation for maximum capacity. The MWIS based scheduling of (13) represents a back-pressure based dynamic control over tuple links [1], [19]. The throughput-optimality of such a control algorithm could be readily obtained in the virtual SR-SC context provided by the tuple-based model; the stability analysis can follow the standard procedures given in [19], defining the Lyapunov function $V(t) = \sum_f \left(q^f(t)\right)^2 + \sum_{f,p \in \mathcal{P}} \left(Q_p^f(t)\right)^2$. Due to the space limit, the proof is ignored.

Remark 1: The tuple-based formulation (13) has significant theoretical advantage compared to the link-based model [3], [4]: the scheduling algorithm in [3] requires pre-computed flow paths, and does not address how to jointly solve the scheduling and routing. Nevertheless, the tuple-based formulation could achieve a joint design by letting the scheduling determine active tuples over the whole tuple set $\mathcal{P}$. Moreover, the back pressure algorithm in [4] is not capable of decoupling the channel/radio assignment, thus it involves the additional virtual link capacity into the formulation.

Complexity discussion: In the tuple-based framework, each source node $s(f)$ maintains an input queue q^f for flow f. Each tuple p creates a unique queue Q_p^f for commodity f. There are $\sum_{n=1}^{|\mathcal{N}|} M_n C$ tuples, and the number of queues is $|F| + \sum_{n=1}^{|\mathcal{N}|} M_n C F$. Though the number of tuples and queues may be large, they are still in the linear relation with the number of nodes and commodities.

VI. Low-Complexity Scheduling Algorithms

Though the tuple-based back pressure (13) leads to the throughput-optimal scheduling scheme, it is an NP-hard problem [1]. Our goal is to develop low-complexity algorithms with the guaranteed performance. In this section, we first implement the tuple-based greedy maximal scheduling[1] in the MR-MC network, and then extend the distributed maximal scheduling to MR-MC networks. Unlike the direct generalization of maximal scheduling, which may lead to very bad performance [3], the tuple-based maximal scheduling ia shown to achieve a constant portion of the optimal capacity region.

Let $S(t)$ be a schedule set, which contains a set of tuple links to be active at time slot t. A schedule set $S(t)$ is maximal if $S(t)$ is an interference-free set, and any additional tuple link added to $S(t)$ will introduce conflict to $S(t)$. Each tuple maintains a unique queue for a commodity in the tuple-based model. However, scheduling algorithms are to pick a set of tuple links to transmit at each time slot, thus it is more convenient to focus on the queue length of the tuple links. The tuple link l_p maintains a unique queue $q_{l_p}^f$ for flow f, defined as:

$$q_{l_p}^f = [Q_{b(l_p)}^f - Q_{e(l_p)}^f]^+ \tag{16}$$

where $b(l_p)$ is the transmitting tuple of tuple link l_p, and $e(l_p)$ is the receiving tuple.

A. Greedy Maximal Scheduling

The tuple-based model maps the MR-MC network to a virtual SR-SC network, therefore the greedy maximal scheduling developed in the SR-SC context could be directly applied in the MR-MC network to approximate the throughput-optimal scheduling (13). The basic idea is to compute the maximal schedule set by starting from the tuple link l_p, which has the largest queue weighted rate $q_{l_p}^f \cdot w_{l_p}^{c(l_p)}$. The greedy maximal scheduling proceeds as follows:

- *Step 1:* Pick the tuple link with largest product $q_{l_p}^f(t) \cdot w_{l_p}^{c(l_p)}$, and add it into the schedule set $S(t)$.
- *Step 2:* Remove all the tuple links interfering with the selected tuple link l_p from the tuple link set $\mathcal{L}_\mathcal{P}$.
- *Step 3:* Repeat step 1 and 2 until the set $\mathcal{L}_\mathcal{P}$ is empty.

Regarding the performance of the tuple-based greedy maximal scheduling, we have the following theorem:

Theorem 1: The tuple-based greedy maximal scheduling could achieve an efficiency ratio of $\frac{1}{\xi}$, i.e. $\frac{1}{\mathcal{K}+2}$.

According to the definition of the interference degree $\mathcal{K}$, turning off the tuple link l_p allows at most $\mathcal{K}_l$ tuple links to be active on the channel $c(l_p)$, where l is the original link of tuple link l_p. In addition, two more radios are released, so that at most another two tuple links can be activated on two different channels other than $c(l_p)$. Thus, by turning off l_p, there are at most $\mathcal{K}+2$ tuple links that can be active simultaneously without interference. The conflict degree ξ of the tuple-based network equals to $\mathcal{K}+2$. Due to the limited space, we skip the proof details of theorem 1, which is similar to the argument in [3].

[1]We use greedy maximal scheduling and maximal scheduling to refer the centralized and distributed maximal scheduling respectively.

B. Distributed Maximal Scheduling

Greedy maximal scheduling has been shown to be an efficient algorithm [16], however it requires centralized computation and global information. We are particularly interested in the algorithm that can be implemented in distributed manner. The link-based model [3] interleaves the channel/radio assignment and link scheduling issues together, therefore direct extension of maximal scheduling to MR-MC networks may lead to very bad performance. The tuple-based model is capable of decoupling these issues, and allows us to directly extend the maximal scheduling to the MR-MC network with the tuple-based model. A tuple link l_p is backlogged if $\frac{\max_f q_{l_p}^f}{\gamma} > w_{l_p}^{c(l_p)}$, where γ is a positive constant. A large γ allows a smaller number of tuple links to be backlogged, while those heavy-backlogged tuple links are more likely to be selected. Furthermore, the control parameter γ can be used to deal with the channel-diversity issue [3], where a large γ can be applied to a tuple-link associated with a low-capacity channel to reduce its chance to be blocked, thus not being scheduled. For any tuple link l_p that is backlogged at time slot t:, either of the following is true

- The tuple link l_p is activated on channel $c(l_p)$.
- Or another tuple link l'_p, which is backlogged and conflicts with l_p, is scheduled on channel $c(l'_p)$.

The selected tuple link l_p will carry flow f with the rate $w_{l_p}^{c(p)}$ if flow f has the longest queue length compared to the other flows at l_p. It is obvious that the tuple-based maximal scheduling needs to compute only one maximal schedule set instead of one maximal schedule set per-channel [3].

We next establish the network stability of the tuple-based maximal scheduling. Since all commodities are multi-hop flows, we assume that each flow has one fixed path through the network. Let $[M_{l_p}^f]$ denote the routing matrix. $M_{l_p}^f = 1$ if the tuple link l_p is used by the flow f, and $M_{l_p}^f = 0$ otherwise. The evolution of each tuple link queue for flow f is given by:

$$q_{l_p}^f(t+1) = q_{l_p}^f(t) + M_{l_p}^f \lambda^f - w_{l_p}^{c(l_p)} \cdot \mathbf{1}_A \tag{17}$$

Note that the arrival rate is given. Event A denotes the fact that $l_p \in S(t)$ and $f = \text{argmax}_f q_{l_p}^f$, implying that the backlog $q_{l_p}^f$ will be reduced only if the tuple link l_p is scheduled and $q_{l_p}^f$ is longer than other flow queues. Equation (17) applies the simplified assumption that the new arrival packets of flow f are injected to all tuple links along the path simultaneously.

Theorem 2: The tuple-based maximal scheduling can stabilize the network with the multi-hop flows if the input rate within any interference area satisfies:

$$\sum_{l'_p \in I(l_p)} \frac{M_{l'_p}^f \lambda^f}{w_{l'_p}^{c(l'_p)}} < 1 \tag{18}$$

Proof: We define the following Lyapunov function to establish the stability.

$$V(t) = \sum_{l_p,f} \frac{q_{l_p}^f(t)}{w_{l_p}^{c(l_p)}} \Big(\sum_{l'_p \in I(l_p)} \frac{q_{l'_p}^f(t)}{w_{l'_p}^{c(l'_p)}} \Big) \tag{19}$$

Then we have:

$$V(t+1) - V(t) = \sum_{l_p,f} \frac{q_{l_p}^f(t+1) - q_{l_p}^f(t)}{w_{l_p}^{c(l_p)}} \Big(\sum_{l'_p \in I(l_p)} \frac{q_{l'_p}^f(t)}{w_{l'_p}^{c(l'_p)}} \Big)$$
$$+ \sum_{l_p,f} \frac{q_{l_p}^f(t)}{w_{l_p}^{c(l_p)}} \Big(\sum_{l'_p \in I(l_p)} \frac{q_{l'_p}^f(t+1) - q_{l'_p}^f(t)}{w_{l'_p}^{c(l'_p)}} \Big)$$
$$+ \sum_{l_p,f} \frac{q_{l_p}^f(t+1) - q_{l_p}^f(t)}{w_{l_p}^{c(l_p)}} \Big(\sum_{l'_p \in I(l_p)} \frac{q_{l'_p}^f(t+1) - q_{l'_p}^f(t)}{w_{l'_p}^{c(l'_p)}} \Big)$$

In the above derivation, the first two terms are equal based on the following simple property:

$$\sum_{l_p,f} \frac{q_{l_p}^f(t+1)}{w_{l_p}^{c(l_p)}} \Big(\sum_{l'_p \in I(l_p)} \frac{q_{l'_p}^f(t)}{w_{l'_p}^{c(l'_p)}} \Big)$$
$$= \sum_{l_p,f} \frac{q_{l_p}^f(t)}{w_{l_p}^{c(l_p)}} \Big(\sum_{l'_p \in I(l_p)} \frac{q_{l'_p}^f(t+1)}{w_{l'_p}^{c(l'_p)}} \Big)$$

Since the cardinality of any interference area $I(l_p)$ is bounded, and consider the queue update (17), we bound the drift:

$$\begin{aligned} & E[V(t+1) - V(t) | q_{l_p}^f(t)] \\ & \le 2 \sum_{l_p,f} \frac{q_{l_p}^f(t)}{w_{l_p}^{c(l_p)}} \Big(\sum_{l'_p \in I(l_p)} \frac{M_{l'_p}^f \lambda^f - w_{l'_p}^{c(l'_p)} \cdot 1_A}{w_{l'_p}^{c(l'_p)}} \Big) + B \\ & \le 2 \sum_{l_p,f} \frac{q_{l_p}^f(t)}{w_{l_p}^{c(l_p)}} \Big(\sum_{l'_p \in I(l_p)} \frac{M_{l'_p}^f \lambda^f}{w_{l'_p}^{c(l'_p)}} - 1 \Big) + B \\ & \le -2\epsilon \sum_{l_p,f} \frac{q_{l_p}^f(t)}{w_{l_p}^{c(l_p)}} + B \end{aligned} \tag{20}$$

where B is a constant, and $\epsilon = 1 - \max_{l_p} \sum_{l'_p \in I(l_p)} \frac{M_{l'_p}^f \lambda^f}{w_{l'_p}^{c(l'_p)}} > 0$ according to (18). The second inequality relies on the fact that at least one tuple link will be scheduled within any interference area at each time slot, i.e. $\sum_{l'_p \in I(l_p)} \frac{w_{l'_p}^{c(l'_p)} \cdot 1_A}{w_{l'_p}^{c(l'_p)}} \ge 1$.

We derive the Lyapunov drift, indicating that the expected change of Lyapunov function is negative if the sum of queue length is large, thus the network stability follows [19]. ■

Condition (18) can be viewed as the sufficient condition for network stability with multi-hop network flows. However, it is conservative sometimes since it neglects the possible parallel transmissions within an interference area. Ideally, there are

at most ξ tuple links that can be active simultaneously by turning off a tuple link. Therefore the necessary condition can be written:

$$\sum_{l'_p \in I(l_p)} \frac{M^f_{l'_p} \lambda^f}{w^{c(l'_p)}_{l'_p}} < \xi \tag{21}$$

Based on the condition (21), the efficiency ratio of the tuple-based maximal scheduling can be determined as $\frac{1}{\xi}$ (i.e., $\frac{1}{\mathcal{K}+2}$). That is the tuple-based maximal scheduling derives a capacity region no smaller than $\frac{1}{\xi}$ times the optimal capacity region. Recall that the efficiency ratio of maximal scheduling in the SR-SC network is $\frac{1}{\mathcal{K}}$. This fact substantiates that our tuple-based model successfully transforms the MR-MC network to a virtual SR-SC network.

C. Distributed Implementation

We design a randomized algorithm to implement the tuple-based maximal scheduling in distributed manner, where each node coordinates with its neighbors to avoid collision and arrange a set of interference-free transmissions based on the local information. The new algorithm could be proved to produce a maximal schedule set with high probability following the argument in [18]. The algorithm consists of three main procedures: 1. information exchange and backlog update; 2. randomly select the tuple link that is backlogged; 3. update the neighborhood information and tuple status. We show that the running time of the algorithm is $O\left(|\mathcal{N}|\Delta \log |\mathcal{N}|^2\right)$, where Δ is the maximum node degree in the network. Due to the limited space, we skip the details.

VII. A Cross-layer Control Minimizing End-to-End Delay

Though the back pressure algorithm computes the optimal capacity region, it has been shown that the classic back-pressure algorithm could lead to unnecessarily large delays [3], [15]. In this section, we aim to enhance the delay performance of tuple-based back pressure (13) in MR-MC networks through the cross-layer design. A decomposable cross-layer framework is developed to minimize the end-to-end delay integrating a network-layer path selection with the link-layer scheduling.

Assume that there are $R(f)$ paths from source to destination for flow f, and H_k is the number of hops for path k, where $\{k : k = 1, \ldots, R(f)\}$. Further, we define a routing matrix $[\Upsilon^f_{kp}]$; $\Upsilon^f_{kp} = 1$ indicates tuple p is on the path k of flow f, and 0 otherwise. Let A_{fk} denote the fraction of flow f injected to path k. Note that in this section, the input rate $\vec{\lambda}$ is given. We formulate the optimization problem (**P2**):

$$\min_{A_{fk}, \vec{r}} \quad \sum_f \sum_{k=1}^{R(f)} V H_k A_{fk} \tag{22}$$

Subject to:

$$\sum_{k=1}^{R(f)} A_{fk} \Upsilon^f_{kp} + \sum_{p':(p',p)\in\mathcal{L}_\mathcal{P}} r^f_{(p',p)} \le \sum_{p'':(p,p'')\in\mathcal{L}_\mathcal{P}} r^f_{(p,p'')} \quad \forall p,\ f \tag{23}$$

$$\sum_{k=1}^{R(f)} A_{fk} = \lambda_f,\ A_{fk} \ge 0, \quad \forall f \tag{24}$$

where V is a positive constant. The rest constraints are the same as (4), (5) and (8). The objective function (22) is to minimize the average number of hops to support all traffic. We do not need the variable $\vec{\mu}$ in (23) since the routing matrix has been given. Note that we use the simplified assumption in constraint (23) that the new arrival packets of flow f are injected to all tuple links along the path of flow f simultaneously. We introduce β^f_p as the Lagrange multiplier associated with constraint (23):

$$L(\beta) = \min_{A_{fk}} \left\{ \sum_f \sum_{k=1}^{R(f)} A_{fk} \left(V H_k + \sum_p \beta^f_p \Upsilon^f_{kp} \right) \right\} \tag{25}$$

$$- \max_{\vec{r}} \left\{ \sum_{f,(p,p')\in\mathcal{L}_\mathcal{P}} \left(\beta^f_p - \beta^f_{p'} \right) r^f_{(p,p')} \right\} \tag{26}$$

The Lagrange multiplier β^f_p is related to the queue length Q^f_p. We are inspired by (25) that the total amount λ^f of commodity f is injected into path k if path k minimizes the value of $VH_k + \sum_p Q^f_p \Upsilon^f_{kp}$. We are motivated to apply the tuple-based back pressure formulation (13) to solve (26) for the optimal $\vec{r}^*$, where the Lagrange multiplier β^f_p is interpreted as the tuple backlog. Thus, we design a joint routing and scheduling algorithm:

- *Step 1: Path selection.* At time slot t, for flow f, the input traffic λ^f is injected into the path k^* such that

$$\left\{ k^* : k^* = \text{argmin}_{1\le k\le R(f)} V H_k + \sum_p Q^f_p(t) \Upsilon^f_{kp} \right\} \tag{27}$$

- *Step 2: Scheduling.* Apply the tuple-based back pressure algorithm (13) for the throughput-optimal scheduling.

Note that the expression $VH_k + \sum_p Q^f_p \Upsilon^f_{kp}$ estimates the end-to-end delay in terms of transmission delay and queuing delay, respectively. The hop count H_k implies the aggregate transmission delay over all tuple links if the failure probability of transmission over each tuple link is similar. Furthermore, the summation $\sum_p Q^f_p \Upsilon^f_{kp}$ implies the total queuing delay along the path k without considering the MAC specification. It is obvious in (27) that the constant V also affects the performance of the algorithm. A large V gives higher priority to those paths having less number of hops, leading to the shorter average transmission delay. However, this may result in large queuing delay since the total number of queue length is given the smaller weight in the weighted average summation. There exists a trade-off when determining the value of V.

Remark 2: A hop-count based path selection [15] has been exploited in SR-SC networks, but the hop-count is not enough to determine the delay in the MR-MC context. For example, consider routing over two connected links with two available channels. There exist 4 possible paths considering the channel combinations, each of which may give different delay due to the channel-dependent link capacity. For satisfying delay

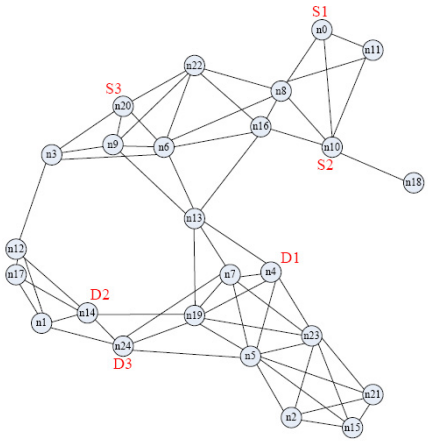
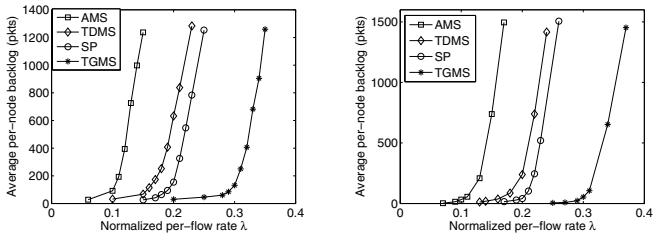


Fig. 2. Random topology.



(a) 2 radios and 4 channels.

(b) 3 radios and 7 channels.

Fig. 3. Average backlog with single-hop traffic under the different scheduling algorithms.

performance, we must exploit the channel-dependence in the network-layer path selection. Furthermore, it is important to have a decomposable cross-layer control framework for the convenience of implementation [15]. The distributed scheduling in [3] cannot provide such a decomposable cross-layer framework due to its coupled design interleaving the link scheduling and the channel/radio assignment. The coupled scheduling design [3] could only partially present the input queue information to the network layer, but inhibits the output queues from being considered in a decomposable manner, thus leading to inaccurate evaluation of the path delay.

It is worth noting that the performance of the cross-layer method depends on the size of alternate path set in step 1. If we can enumerate all the possible paths, the optimal solution can be reached. In practice, it is too complex to compute all paths. It is shown [3] that a small set is enough, and a dynamic update process is used to discover the paths online, which is not suitable for the tuple-based model. The reason is that a path consists of several tuples rather than nodes under the tuple-based model, thus computing the tuple-based path is too complex. In this paper, we apply MCF-based method in [9] to compute a set of paths which provide a larger capacity region. The low-complexity algorithms developed in Section V could be applied in step 2 instead of the tuple-based back pressure algorithm (13), which guarantees a constant portion of the optimal capacity region.

VIII. Performance Evaluation

In this section, we investigate the performance of the proposed algorithms through simulations. We setup a random topology with 25 nodes as shown in Fig. 2. The transmission range and interference range of each node is set to 250m and 500m, respectively. Link capacity over each channel is uniformly selected in the range [0.1,1] in each slot to simulate the channel diversity.

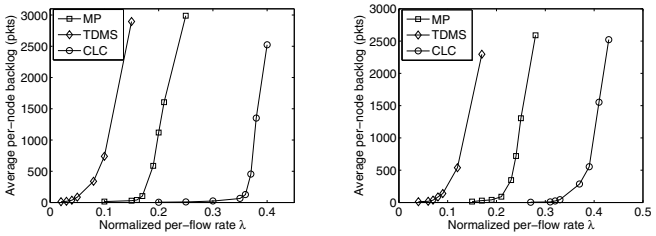
A. Comparison of Scheduling Algorithms

In this experiment, we investigate the effectiveness of the tuple-based greedy maximal scheduling (TGMS) and the tuple-based distributed maximal scheduling (TDMS), with comparison to the aggregated maximal scheduling (AMS) and the single path (SP) algorithm proposed in [3]. We developed C codes to implement all these algorithms. We randomly pick 20 single-hop source-destination pairs (i.e., directed links). The input rate of each flow is the same, denoted as λ with the unit of packets/slot. We compute the average queue length at each source node as the performance metric.

In Fig. 3, we plot the average backlog at all source nodes versus the input rate λ. The constant γ in the TDMS is set to 10. When input rate arrives to a certain value, the average queue length rapidly goes to very large or infinite. This implies that the input rate is approaching the boundary of capacity region. It is obvious that the TDMS achieves larger capacity region than that of AMS, since AMS assigns all channels to an active link for data transmission, which may lead to potential bandwidth waste if the link has less data to transmit than the total capacity of all the links. The TGMS obtains the best performance as expected, however it requires global information and centralized computation, which is not computationally efficient. It is interesting that SP algorithm has better performance than TDMS. Though the logical queue assignment procedure in SP algorithm may lead to longer waiting time based on the output queue information, it could utilize those high capacity channels. While TDMS just randomly picks up a tuple link to transmit within the interference area for each tuple link without considering channel diversity.

B. Delay Enhancement by Cross-layer Control

In this experiment, we exam the performance of cross-layer control algorithm in terms of delay. Three multi-hop commodity flows are considered as shown in Fig. 4, where the source and destination nodes for flow $i(i = 1, 2, 3)$ are denoted as S_i and D_i respectively. we gradually increase the flow input rates and observe the average node backlog (averaged over those nodes involved in flow transmissions) to examine the delay performance and capacity region of the multi-path algorithm (MP) [3] and our cross-layer control (CLC) algorithm in Section VII. For the two cross-layer control algorithms CLC and MP, the candidate paths are generated



(a) 2 radios and 4 channels.

(b) 3 radios and 7 channels.

Fig. 4. Average backlog with multi-hop traffic under the cross-layer control algorithm.

based on the MDCG-based MCF solution in [9]. Each node is assumed having an infinite buffer space.

As illustrated in Fig. 4, all three algorithms show the turning point of the average backlog at a certain input rate. The turning point is critical since it implies not only network stability but also the achievable capacity region. Before the turning point, the average backlog keeps small meaning that the network remains stable; when the input rate increases to a certain value, the average backlog rapidly goes to very large or even infinite, which means the total input traffic is out of the capacity region that an algorithm can achieve and the system is unstable. It is clear that TDMS leads to the smallest capacity region since it does not consider any routing performance. Our CLC algorithm achieves larger capacity region compared with the MP [3]. The reason is that MP algorithm does not have a decomposable structure, so that it could not obtain well coordinated scheduling and channel assignment as our CLC scheme, which achieves the good resource allocation through the tuple-based cross-layer framework. More specifically, it applies the link-based model, and separates the input queue and output queue. The input queue is used to make routing decision, and output queue information determines the scheduling result. The routing and scheduling problems cannot be decoupled due to the channel/radio assignment issue. Our CLC algorithm applies the tuple based model, and each tuple has a unique queue for input and output packet, which allows us completely decouple the routing and scheduling algorithm.

IX. Conclusion

In this paper, we develop a tuple-based modeling to systematically study the throughput-optimal control in the MR-MC network by developing the tuple-based back pressure algorithm. The tuple-based model allows us to directly extend the maximal scheduling algorithm with guaranteed performance. We show that the tuple-based maximal scheduling can be implemented with the provable efficiency ratio, taking the channel diversity into account. We also propose a decomposable cross-layer framework based on the tuple-model to enhance the end-to-end delay performance of tuple-based back pressure, integrating the routing and scheduling problem and considering transmission delay and queuing delay.

References

[1] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Automat. Contr.*, vol. 4, pp. 1936-1948, December 1992.

[2] M. Alicherry, R. Bhatia, and L. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *Proc. ACM MobiCom*, Aug. 2005, pp. 58–72.

[3] X. Lin and S. Rasool, "A distributed joint channel-assignment, scheduling and routing algorithm for multi-channel ad hoc wireless networks," in *Proc. IEEE INFOCOM*, May 2007, pp. 1118–1126.

[4] S. Merlin, N. H. Vaidya, and M. Zorzi, " Resource allocation in multi-radio multi-channel multi-hop wireless networks" in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 610–618.

[5] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *Proc. ACM MobiCom*, Aug. 2005, pp. 73–87.

[6] M. J. Neely, "Delay analysis for maximal scheduling in wireless networks with bursty traffic," in *Proc. IEEE INFOCOM*, Apr. 2008.

[7] X. Lin and B. Shroff, "The impact of imperfect scheduling on cross-layer rate control in wireless networks," in *Proc. IEEE INFOCOM*, Mar 2005, pp. 1118–1126.

[8] G. R. Gupta and N. B. Shroff, "Delay Analysis for Wireless Networks with Single Hop Traffic and General Interference Constraints," *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 393-405, April 2010.

[9] H. Li, Y. Cheng, C. Zhou, and P. Wan, "Multi-dimensional conflict graph based computing for optimal capacity in MR-MC wireless networks", in *Proc. IEEE ICDCS*, June, 2010.

[10] P. Chaporkar, K. Kar, and S. Sarkar, "Achieving Queue Length Stability Through Maximal Scheduling in Wireless Networks,"in *Proceedings of Information Theory and Applications Inaugural Workshop*, Feb. 2006.

[11] P. Gupta and P. R. Kumar, "The cpacity of wireless networks," *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 388-404, Mar. 2000.

[12] M. J. Neely, "Optimal backpressure routing for wireless networks with multi-receiver diversity," in *Proc. Conf. on Information Sciences and Systems (CISS)*, 2006, pp. 18-25.

[13] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput Guarantees in Maximal Scheduling in Wireless Networks," in *Proc. 43rd Annual Allerton Conference on Communication, Control and Computing*, September 2005.

[14] X. Wu and R. Srikant, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," in *Proc. IEEE Infocom*, 2006.

[15] L. Ying, S. Shakkottai and A. Reddy, "On Combining Shortest-Path and Back-Pressure Routing Over Multihop Wireless Networks," in *Proc. IEEE Infocom*, 2009.

[16] C. Joo, X. Lin, and N. B. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multi-hop wireless networks," in *Proc. IEEE INFOCOM*, Apr, 2008, 1103-1111.

[17] W. Cheng, X. Cheng, T. Znati, X. Lu, and Z. Lu, "The complexity of channel scheduling in multi-radio multi-channel wireless networks," in *Proc. IEEE INFOCOM* , Apr. 2009, pp. 1512-1520.

[18] G. Sharma, C. Joo, and N. B. Shroff, "Distributed scheduling schemes for throughput guarantees in wireless networks," in *Proc. 43rd Annual Allerton Conference on Communication, Control and Computing*, September 2006.

[19] L. Georgiadis, M. J. Neely, and L. Tassiulas, *Resource Allocation and Cross-Layer Control in Wireless Networks*, Foundations and Trends in Networking, Vol. 1, no. 1, pp. 1-144, 2006.