Distributed Link Scheduling under SINR Model in Multi-hop Wireless Networks

Jin-Ghoo Choi, *Member, IEEE,* Changhee Joo, *Senior Member, IEEE,* Junshan Zhang, *Fellow, IEEE,* and Ness B. Shroff *Fellow, IEEE*

***Abstract*—Link adaptation technologies, such as Adaptive Modulation and Coding (AMC) and Multiple Input Multiple Output (MIMO), are used in advanced wireless communication systems to achieve high spectrum efficiency. Communication performance can be improved significantly by adaptive transmissions based on the quality of received signals, i.e., the signal-to-interference-plus-noise ratio (SINR). However, for multi-hop wireless communications, most link scheduling schemes have been developed under simplified interference models that do not account for accumulative interference, and cannot fully exploit the recent advances in PHY-layer communication theory. This paper focuses on developing link scheduling schemes that can achieve optimal performance under the SINR model. One key idea is to treat an adaptive wireless link as multiple parallel virtual links with different signal quality, building on which we develop throughput-optimal scheduling schemes using a two-stage queueing structure in conjunction with recently developed carrier-sensing techniques. Furthermore, we introduce a novel three-way handshake to ensure, in a distributed manner, that all transmitting links satisfy their SINR requirements. We evaluate the proposed schemes through rigorous analysis and simulations.**

***Index Terms*—Multi-hop wireless networks, link scheduling, CSMA, SINR model.**

I. Introduction

MULTI-HOP wireless networks have recently attracted significant attention due to their potential to achieve substantial gains over their single hop counterparts [1], [2]. Over the past few years, a cross-layer optimization-based framework has emerged that can be used to maximize the network utility (see [3] and the references therein). The solution to such utility maximization problems involves solving *congestion control*, *link scheduling*, and *routing* components. The link scheduler determines which link has to be active at what time and at what power level. The congestion control adjusts the amount of data injected into the network based on congestion information, which can be obtained by feedback from the receiver. The routing determines which path(s) should be taken by packets on a given *flow* (or source-destination pair). It has been shown that the queue length information plays an important role in the interactions between these components. Moreover, under this cross-layer optimization based framework, the scheduling component often turns out to have the highest complexity, and has thus been the focus of attention.

In their seminal work [4], Tassiulas and Ephremides studied joint routing and link scheduling for packet radio networks, assuming no a priori information of input traffic, and developed a throughput-optimal solution, so called the *back-pressure* algorithm. The back-pressure idea has since been applied to various fields such as input-queued switch [5] and cellular data networks [6]. Also, it has been reformulated as a dynamic power control scheme under time-varying channels [7], and shown to maximize the network utility as a scheduling component [3], [8].

Back-pressure, although throughput-optimal, is a centralized algorithm with high computational complexity, and can be shown to be an NP-hard problem for most interference models [9], [10]. In general, centralized schemes would not work well for multi-node wireless networks due to high complexity for exchanging control messages, and the lack of scalability. Thus, researchers have expended significant effort in developing distributed counterpart of high-performance centralized back-pressure schemes, e.g., [11]–[17]. To this end, many studies employ simple *combinatorial* models for wireless interference, where the interference relationship between wireless links is binary. Notably, the k-hop interference model, under which two links cannot transmit simultaneously if their distance is within k hops [9], has been extensively studied. The so-called maximal scheduling[1] scheme has been shown to attain a certain fraction of the optimal throughput for certain combinatorial channels [12], [15], and several provably efficient distributed random access schemes have been developed in [13], [14].

Recently, it has been shown that throughput optimality can be achieved using traditional Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocols in a new way. In [18], it had been shown that the stationary distribution of link activations can be derived in a product form. This result has been recently exploited in [19] to develop fully distributed scheduling schemes to achieve throughput optimality under

J. Choi is with Yeungnam University, Gyeongbuk 712-749, South Korea (e-mail: jchoi@yu.ac.kr).

C. Joo is with UNIST, Ulsan 689-798, South Korea (corresponding author, e-mail: cjoo@unist.ac.kr).

J. Zhang is with Arizona State University, Tempe, AZ 85287 USA (e-mail: junshan.zhang@asu.edu).

N. B. Shroff is with the Ohio State University, Columbus, OH 43210 USA (e-mail: shroff@ece.osu.edu).

Manuscript received XXXX; revised XXX. This work was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2012R1A1A1005972 and NRF-2011-0008549), and in part by the US National Science Foundation under Grants CNS-0905603, CNS-1218484, CNS-1065136, and CNS-1012700, and DoD MURI project No. FA9550-09-1-0643 and a MURI grant from the Army Research Office W911NF-08-1-0238.

[1]Also referred to as maximal matching

combinatorial channel models. The main idea has been generalized to more practical scenarios in a time-slotted system [20], and to utility maximization for multi-channel systems [21].

The combinatorial channel models greatly simplify the interference relationship and make the problem relatively tractable. However, they do not fully capture some fundamental features of wireless channels. Indeed, the link rate is usually a function of the received SINR that hinges heavily on the accumulative interference from all other transmitters. Under simple combinatorial models, this global property of interference is ignored and, thus, a transmitting link considered feasible under those channel models may experience strong interference in practice, and fail to deliver data. It has been shown that scheduling schemes that guarantee a certain fraction of the optimal throughput under the k-hop interference model may attain zero-throughput in the worst case under the SINR-based model [22].

In developing CSMA scheduling under the SINR interference model, one main difficulty is to ensure the feasibility of a schedule in a distributed manner while maintaining the desired product-form stationary distribution of schedules, whereas this can be easily done under the combinatorial interference models [19]–[21]. To this end, the authors in [23] have developed a conservative combinatorial model that conforms the SINR requirements, which results in performance degradation while requiring for each link to have a priori knowledge of the set of links that it can coexist with. In this work, we develop algorithms that provide feasibility test of a schedule under the SINR interference model, which is the key component of the CSMA scheduling to achieve throughput optimality without any priori knowledge on the network topology.

There are also a few works that have been developed non-CSMA scheduling under the SINR channel model. In [24], the greedy maximal scheduling has been modified to operate under SINR channels at the cost of performance degradation. It has been shown that the modified greedy scheme achieves non-diminishing performance. Power control for utility maximization in saturated networks has been developed in [25], which, however, does not take into account traffic dynamics and has substantial overhead of control message exchanges since each link requires global channel information.

In this paper, we develop resource allocation schemes in multi-node[2] wireless networks that not only achieve the optimal throughput under SINR channel models, but are also amenable to distributed implementation. The main contributions of our paper are summarized as follows.

- We model a single physical link as multiple virtual links with different SINR requirements, and are able to formulate the resource allocation as a standard optimization problem. By introducing a two-stage queueing structure, we successfully decompose the overall problem into two subproblems.
- By studying the duality problem, we develop resource allocation schemes that consist of two components: traffic splitter and link scheduler. We provide an optimal traffic splitter, and characterize the property of an optimal link scheduler.
- We develop practical solutions by using CSMA techniques that are amenable to distributed implementation. We introduce a novel three-way handshake to satisfy the SINR requirements of each active link in a distributed manner.
- Finally, we show through rigorous analysis that the proposed schemes follows the same Markov chain evolution as the optimal solution, and provide simulations to support our evaluations.

[2]Here, by multi-node wireless systems, we mean wireless systems, where multiple nodes can transmit to multiple destination in a single or multi-hop fashion.

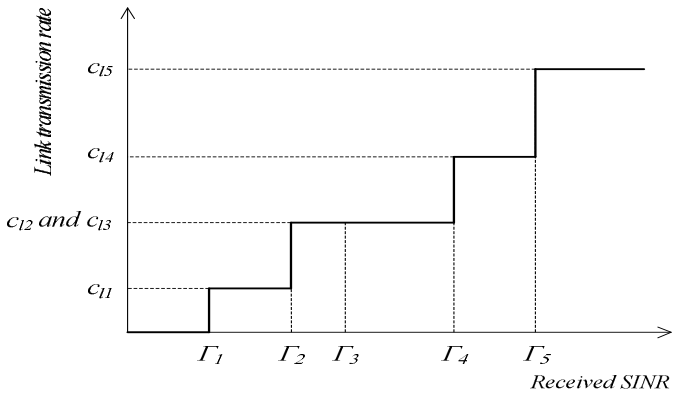

Fig. 1. Transmission rate of link l vs. SINR.

The rest of this paper is organized as follows. In Section II, we describe the system and channel models. In Section III, we develop a fully distributed throughput optimal resource allocation scheme using channel sensing. We evaluate our proposed scheme through simulations and make some interesting observations in Section IV. We conclude our paper in Section V.

II. Model description

A. System model

We model a wireless network by a graph $G = (N, L)$, where N is the set of nodes and L is the set of directed links. We assume the reciprocity of wireless channels for any pair of nodes a and b, i.e., if node a is connected to node b by a link, there always exists a link in the opposite direction from node b to node a. A session is a stream of data in the network from a source node to a destination node. We assume that all sessions traverse one hop only, and there is at most one session per link. *This one-hop model is only for ease of exposition and can be extended to multi-hop sessions following the approach in [19].*

The system operates in a time-slotted and synchronized fashion. At the beginning of time slot t, data bits are injected into link l at its source, and wait in the queue until they are served in a first-come first-served manner. When the link is activated, backlogged bits are transmitted and leave the network (since they have reached the destination). The amount of served bits is determined by the transmission rate of link l, depending on the SINR. Let $A_l(t)$ denote the amount of

data bits injected into link l in time slot t. We assume that the arrival process $\{A_l(t)\}_{t=0}^{\infty}$ is stationary and *i.i.d.* across time and, further, the first and second moments of $A_l(t)$ are finite. The average arrival rate of data bits at link l is denoted by $\lambda_l := \lim_{T\to\infty} \frac{1}{T} \sum_{\tau=0}^{T-1} A_l(\tau)$.

B. Channel model

Let P_l denote the transmission power of (the transmitting node of) link l. We assume that the transmission power is fixed. Let G_{lm} represent the channel gain from the transmitter of link l to the receiving node of link m, and let G_{ll} denote the channel gain of link l itself. When the transmitter of link l sends a signal, the corresponding receiver attains signal strength of $P_l G_{ll}$, and the receiver of another link $m(\neq l)$ experiences interference of $P_l G_{lm}$.

Let γ_l denote the received SINR at link l. We assume that each link can support V different transmission rates based on the value of γ_l. To elaborate, we group the domain of γ_l into $V+1$ fixed intervals using positive constants, $\Gamma_1 < \Gamma_2 < \cdots < \Gamma_{V+1}(=\infty)$. At the v-th interval for each $v \in \{1, 2, \ldots, V\}$, i.e., when $\gamma_l \in [\Gamma_v, \Gamma_{v+1})$, link l can achieve transmission rate of $c_{lv}(>0)$. When $\gamma_l \in [0, \Gamma_1)$, the transmission rate is 0. If link l transmits data with higher rate transmission mode than c_{lv}, then the probability the receiver can decode the data correctly decreases to an unacceptably low level. The v-th transmission mode with rate c_{lv} may correspond to a different AMC scheme and/or a different MIMO antenna configuration [23]. Fig. 1 depicts an example of the rate curve of a link, which is an (unequal) staircase function due to the quantization of the SINR level. Note that in our definition, the transmission rate monotonically increases, but does not strictly increase with the SINR level, as shown for $v = 2$ and 3 in the figure. This piecewise constant rate model is more realistic than continuous rate models such as the Shannon capacity formula, since practical communication systems support only a finite number of transmission modes. Moreover, by increasing the number of modes V, our model can closely approximate the Shannon capacity formula. We assume heterogeneous link rate curves, i.e., each link has its own curve.

We represent each v-th transmission mode of (physical) link l by a *virtual link* $[l, v]$, as first introduced in [23]. Virtual link $[l, v]$ is said to be active when link l transmits data using the v-th mode. Let $x_{lv} \in \{0, 1\}$ represent the activity of virtual link $[l, v]$, i.e., $x_{lv} = 1$ if virtual link $[l, v]$ is active, and $x_{lv} = 0$ otherwise. We call $\mathbf{x} := (x_{lv})$ as the *virtual link activation vector*. For a given $\mathbf{x}$, the received SINR of virtual link $[l, v]$ can be calculated as

$$\gamma_{lv}(\mathbf{x}) = \frac{x_{lv} P_l G_{ll}}{\sum_{m,w} x_{mw} P_m G_{ml} - x_{lv} P_l G_{ll} + \eta_l}, \qquad (1)$$

where η_l is the background noise power at the receiver of link l. Note that for a single link, only one virtual link can be activated at any given time, and the SINR of active virtual link $[l, v]$ is equivalent to the SINR of physical link l at that time. Hence, if virtual link $[l, v]$ is active and satisfies the SINR requirement $\gamma_{lv} \geq \Gamma_v$, the transmission rate of the virtual link is c_{lv}, and otherwise, 0. We denote the *average* transmission rate of virtual link $[l, v]$ by $r_{lv} := \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} c_{lv} x_{lv}(t)$, where $x_{lv}(t)$ represents the activity of $[l, v]$ at time slot t.

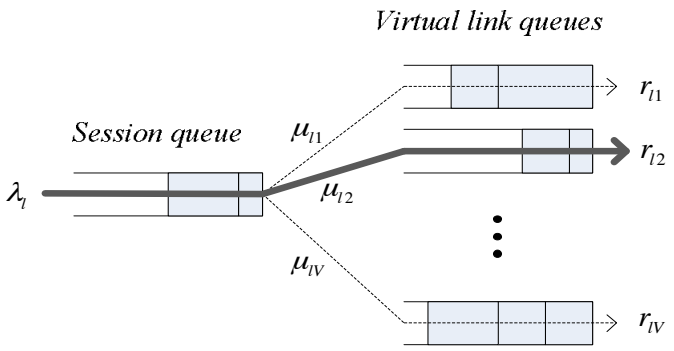

Fig. 2. Queueing structure of link l. The solid arrow indicates an example processing path of data bits.

C. Queueing structure

Based on the fact that a physical link accommodates multiple virtual links, we develop a two-stage queueing structure as shown in Fig. 2. A similar queueing structure has been used for scheduling with multi-channel wireless links [26]. All incoming bits are put into the first-stage *session queue*, and then distributed to V *virtual link queues* at the second stage, each of which is associated with a virtual link. Data bits arrive at the session queue at rate λ_l (bits/slot), and move from the session queue to the v-th virtual link queue at service rate μ_{lv}. Since the data movement between queues are *internal*, it does not rely on the SINR of the channel. The data bits in the v-th virtual link queue will be transmitted to the next node at rate c_{lv} when the associated virtual link is active.

This two-stage queueing structure may result in some undesirable effect: a packet should wait until a virtual queue is consumed, and it may experience long delay if the virtual queue that it moves to remains inactive for a long period. We can address the problem by implementing the queueing structure with a single FIFO queue and $(1+V)$ counter variables at each link. Each counter variable maintains the queue length of the session queue and the virtual queues. When a packet arrives at a link, the packet is physically enqueued into the FIFO queue and the counter variable for the session queue increases. Whenever a packet needs to move from the session queue to a virtual link queue, the counter variable is decreased accordingly for the session queue and the counter corresponding to the virtual link queue is increased by the same amount. Note that the physical packet does not move and still wait for service in the FIFO queue during this process. When a virtual link is scheduled to transmit, we transmit the head-of-line packet in the FIFO queue and decrease the counter for the virtual link queue.

D. Capacity region

We begin with specifying the *feasible* virtual link activation vectors. For a virtual link activation vector $\mathbf{x}$, the associated link activation vector is defined as $\hat{\mathbf{x}} := (\hat{x}_l)$, where $\hat{x}_l :=$

$\sum_v x_{lv}$, and link l is active if $\sum_v x_{lv} = 1$, and inactive if $\sum_v x_{lv} = 0$. The received SINR of link l can be calculated as

$$\gamma_l(\hat{\mathbf{x}}) = \frac{\hat{x}_l P_l G_{ll}}{\sum_m \hat{x}_m P_m G_{ml} - \hat{x}_l P_l G_{ll} + \eta_l}. \quad (2)$$

Based on the reciprocity of wireless channel, if link l connects node a to node b, there always exists a reverse link in the opposite direction from node b to a. For $\hat{\mathbf{x}}$, we denote its *reverse* link activation vector as $\mathbf{rev}(\hat{\mathbf{x}}) := (rev_l(\hat{\mathbf{x}}))$, where $rev_l(\hat{\mathbf{x}}) = 1$ if and only if link l is the reverse link of any active link in $\hat{\mathbf{x}}$, and 0 otherwise.

Definition 1: A virtual link activation vector $\mathbf{x}$ is *feasible* if

1) $\sum_v x_{lv} \le 1$ for all links l,
2) $\gamma_{lv}(\mathbf{x}) \ge \Gamma_v$ for all active virtual links $[l, v]$ of $\mathbf{x}$, and
3) $\gamma_l(\mathbf{rev}(\hat{\mathbf{x}})) \ge \Gamma_{ctrl}$ for all active links l in $\mathbf{rev}(\hat{\mathbf{x}})$, where $\hat{\mathbf{x}}$ is the link activation vector associated with $\mathbf{x}$, and Γ_{ctrl} is the minimum SINR requirement for the reverse links.

The first condition implies that at most one virtual link can be activated in a physical link. The second condition requires that all the active links have a large enough SINR to support the transmission mode used. The third condition means that the reverse links should satisfy the minimum requirement on SINR, and is needed to guarantee that each receiver can feedback control packets to the transmitter through the reverse link. Owing to the tiny size of control packets, small value of Γ_{ctrl} would be sufficient under the protection of strong channel coding schemes. Without loss of generality, we assume that $\Gamma_{ctrl} \le \Gamma_1$. The second and third conditions imply that under any 'feasible' activation vector, control packets can be successfully delivered through both the forward and reverse links.

Let F denote the collection of all the feasible virtual link activation vectors in a network. We consider a link scheduling policy that chooses a virtual link activation vector $\mathbf{x}$ among F and activates the virtual links $[l, v]$ with $x_{lv} = 1$, satisfying the SINR requirements of links. Since the active virtual link $[l, v]$ has the transmission rate of c_{lv}, we denote the virtual link transmission rate vector for $\mathbf{x}$ by $\mathbf{c} \cdot \mathbf{x}$, where $\mathbf{c} := (c_{lv})$, using the component-wise multiply, i.e., $\mathbf{c} \cdot \mathbf{x} = (c_{lv} x_{lv})$. Then, any achievable vector of average virtual link transmission rates $\mathbf{r} := (r_{lv})$ can be obtained by time-interleaving of feasible virtual link activations, i.e., by a certain combination of $\mathbf{c} \cdot \mathbf{x}$ over F. Now we define the *capacity region* Λ as the set of arrival rate vectors $\boldsymbol{\lambda} := (\lambda_l)$ that is no greater than an achievable transmission rate vector component-wise, i.e.,

$$\Lambda := \{\boldsymbol{\lambda} \mid \lambda_l \le \sum_v r_{lv} \ \forall l, \text{ for some } \mathbf{r} \in Co(\mathbf{c} \cdot \mathbf{x}), \ \mathbf{x} \in F\}, \quad (3)$$

where $Co(\cdot)$ represents the convex hull. Clearly, if an arrival rate vector is not in the set Λ, it cannot be accommodated in the network with any scheduling policy. If a link scheduling policy supports any traffic that is strictly within the capacity region, the scheduler is said to be *throughput optimal*. In this work, we are interested in developing throughput-optimal scheduling schemes that are amenable to distributed implementation.

III. Link Scheduling under SINR Model

We first formulate the problem formally and provide a throughput-optimal solution using the standard optimization techniques. For practical implementation, we develop a distributed resource allocation scheme with the two-stage queueing structure and a novel three-way handshake, and show that the proposed distributed link scheduling scheme still achieves the optimal throughput.

A. Problem formulation

Recall that λ_l is the arrival rate of data bits at link l, μ_{lv} is the internal service rate from its session queue to the v-th virtual link queue, and r_{lv} is the average service rate of the v-th virtual link queue. At each time slot, the link scheduler chooses a feasible virtual link activation vector and serves the chosen virtual links.

For feasible virtual link activation vectors in F, denote the i-th feasible vector by $\mathbf{x}^{(i)} := (x_{lv}^{(i)})$. Suppose that a scheduling policy chooses feasible virtual link activation vectors following a stationary distribution, and let α_i denote the selection probability of the i-th feasible vector. It has been shown in [19] that the scheduling policy achieves the optimal throughput performance if $\{\alpha_i\}$ is the solution to the following optimization problem $\mathbf{P}$:

$$\begin{aligned} \mathbf{P}: \ \underset{\boldsymbol{\alpha}\ge\mathbf{0},\ \boldsymbol{\mu}\ge\mathbf{0}}{\text{maximize}} \ & -\sum_i \alpha_i \log \alpha_i \\ \text{subject to} \ & \lambda_l \le \sum_v \mu_{lv}, \quad \forall l, \\ & \mu_{lv} \le r_{lv}, \quad \forall l, v, \\ & \sum_i \alpha_i = 1, \end{aligned} \quad (4)$$

where $\boldsymbol{\alpha} := (\alpha_i)$ and $\boldsymbol{\mu} := (\mu_{lv})$, and $\mathbf{0}$ is a zero vector. The first and second constraints are the stability conditions for session queues and for virtual link queues, respectively. The average service rate r_{lv} is a linear function of $\boldsymbol{\alpha}$ and can be written as $c_{lv} \sum_i \alpha_i x_{lv}^{(i)}$. The last constraint along with $\boldsymbol{\alpha} \ge \mathbf{0}$ comes from the fact that $\boldsymbol{\alpha}$ is a probability distribution. The objective of the problem is to maximize the entropy of $\boldsymbol{\alpha}$.

Problem $\mathbf{P}$ has a unique optimizer as long as it is feasible because the objective function is strictly convex and the constraint set is convex. Suppose that an arrival rate vector $\boldsymbol{\lambda}$ is within the capacity region. According to (3), we can find non-negative numbers $(\tilde{\alpha}_i)$ such that $\lambda_l \le \sum_v \left(\sum_i \tilde{\alpha}_i c_{lv} x_{lv}^{(i)} \right)$ and $\sum_i \tilde{\alpha}_i = 1$. Let $\tilde{\mu}_{lv} := \sum_i \tilde{\alpha}_i c_{lv} x_{lv}^{(i)}$. It is easy to see that $(\tilde{\alpha}_i)$ and $(\tilde{\mu}_{lv})$ satisfy all the constraints of problem $\mathbf{P}$ and the constraint set is not empty. Hence, for any $\boldsymbol{\lambda} \in \Lambda$, problem $\mathbf{P}$ is solvable since it is a convex optimization problem.

B. Throughput-optimal scheduling

We next solve the dual problem of $\mathbf{P}$ rather than solving the more challenging primal problem directly. Let $\mathbf{q} := (q_l)$ denote the vector of Lagrange multipliers associated with the first constraint, and $\mathbf{q_v} := (q_{lv})$ represent the vector of Lagrange multipliers associated with the second constraint. From

the standard optimization techniques, a partial Lagrangian can be defined as

$$L(\boldsymbol{\alpha}, \boldsymbol{\mu}; \mathbf{q}, \mathbf{q_v}) = -\sum_i \alpha_i \log \alpha_i + \sum_{l,v} q_{lv} r_{lv} + \sum_{l,v} \mu_{lv}(q_l - q_{lv}) - \sum_l q_l \lambda_l, \quad (5)$$

and the dual problem **D** of problem **P** can be formulated as

$$\textbf{D:} \ \text{minimize} \sup_{\boldsymbol{\alpha}\geq\mathbf{0}, \boldsymbol{\mu}\geq\mathbf{0}, \sum_i \alpha_i = 1} L(\boldsymbol{\alpha}, \boldsymbol{\mu}; \mathbf{q}, \mathbf{q_v}) \quad \text{subject to} \ \ \mathbf{q} \geq \mathbf{0}, \mathbf{q_v} \geq \mathbf{0}. \quad (6)$$

Given $\mathbf{q}$ and $\mathbf{q_v}$, the objective function of problem **D** can be decoupled into the following two subproblems, since the last term $\sum_l q_l \lambda_l$ is a constant.

$$\textbf{SubD1:} \quad \underset{\boldsymbol{\mu}\geq\mathbf{0}}{\text{maximize}} \quad \sum_{l,v} \mu_{lv}(q_l - q_{lv}), \quad (7)$$

$$\textbf{SubD2:} \quad \underset{\boldsymbol{\alpha}\geq\mathbf{0}, \sum_i \alpha_i = 1}{\text{maximize}} \quad -\sum_i \alpha_i \log \alpha_i + \sum_{l,v} q_{lv} r_{lv}. \quad (8)$$

In **SubD1**, we introduce an additional constraint on the average internal service rates μ_{lv}, i.e., $\sum_v \mu_{lv} \leq C$, where C is a constant no smaller than $V \cdot \max_{l,v} c_{lv}$. Note that from the second constraint of **P**, any solution will satisfy that

$$\sum_v \mu_{lv} \leq \sum_v r_{lv} \leq V \cdot \max_{l,v} r_{lv} \leq V \cdot \max_{l,v} c_{lv} \leq C, \quad (9)$$

indicating that the new constraint does not affect the solution space. We can now rewrite **SubD1** as

$$\underset{\boldsymbol{\mu}\geq\mathbf{0}, \sum_v \mu_{lv} \leq C}{\text{maximize}} \sum_l \sum_v \mu_{lv}(q_l - q_{lv}). \quad (10)$$

The solution to this subproblem can be obtained as follows. First, each link l finds the virtual link v^* such that

$$v^* = \underset{v}{\text{argmax}} \ (q_l - q_{lv}), \quad (11)$$

and sets each internal transmission rate μ_{lv} as

$$\mu_{lv} = \begin{cases} C, & \text{if } v = v^* \text{ and } q_l > q_{lv^*}, \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

It is noteworthy that problem **SubD1** can be solved in a distributed manner by each link with only local information.

In **SubD2**, the optimal solution will maximize the entropy of the probability distribution $\boldsymbol{\alpha}$ for the feasible virtual link activation vectors, plus the queue weighted rate sum $\sum_{l,v} q_{lv} r_{lv}$. We consider a stationary link scheduling policy that chooses the i-th member $\mathbf{x}^{(i)}$ of feasible set F with probability

$$\pi_i(\mathbf{q_v}) = \frac{1}{Z} \exp\big(\sum_{l,v} c_{lv} q_{lv} x_{lv}^{(i)}\big), \quad (13)$$

where Z is a normalization constant such that $\sum_i \pi_i(\mathbf{q_v}) = 1$. It has been shown in [19] that a scheduling policy that achieves the stationary distribution (13) is an optimal solution to **SubD2**.

Therefore, at each time slot t, we can use (12) and (13) to optimally solve **SubD1** and **SubD2** given $\mathbf{q}(t)$ and $\mathbf{q_v}(t)$, and maximize the value of the objective function of problem **D**. Subgradient algorithm for a convex optimization problem asserts that we can attain the optimal solution of problem **D** by adjusting $\mathbf{q}(t)$ and $\mathbf{q_v}(t)$ as follows: For small enough κ, each Lagrange multiplier is adjusted as

$$q_l(t+1) = \Big[q_l(t) + \kappa\big(\lambda_l - \sum_v \mu_{lv}\big)\Big]_+, \quad (14)$$

$$q_{lv}(t+1) = \Big[q_{lv}(t) + \kappa\big(\mu_{lv} - r_{lv}\big)\Big]_+, \quad (15)$$

where $[x]_+ := \max(x, 0)$.

Finally, we confirm that the *strong* duality holds for problems **P** and **D**, when the arrival rate vector $\boldsymbol{\lambda}$ is *strictly* within the capacity region Λ. From (3), there are non-negative $(\tilde{\alpha}_i)$ such that $\lambda_l = \sum_v \big(\sum_i \tilde{\alpha}_i c_{lv} x_{lv}^{(i)}\big)$ and $\sum_i \tilde{\alpha}_i < 1$. Let h be a constant between 1 and $(\sum_i \tilde{\alpha}_i)^{-1}$. We set $\alpha_i = \tilde{\alpha}_i / \sum_i \tilde{\alpha}_i$, and $\mu_{lv} = h \sum_i \tilde{\alpha}_i c_{lv} x_{lv}^{(i)}$. Clearly, these $\boldsymbol{\alpha}$ and $\boldsymbol{\mu}$ satisfy all the inequality constraints of **P** with *strict inequality* as well as the equality constraint. Hence the Slater's condition holds, and the strong duality ensures that the optimal value of problem **P** equals to the optimal value of problem **D**.

Algorithm 1 Traffic splitter of link l at time slot t.

1: Find $v^* = \text{argmax}_v(Q_l(t) - Q_{lv}(t))$.
2: **if** $Q_l(t) - Q_{lv^*}(t) > 0$ **then**
3: /* Move $\min(C, Q_l(t))$ bits from $Q_l(t)$ to $Q_{lv^*}(t)$ */
4: $Q_l(t+1) = [Q_l(t) - C]_+$;
5: $Q_{lv^*}(t+1) = Q_{lv^*}(t) + \min(C, Q_l(t))$.
6: **end if**

C. Distributed link scheduling

Based on the studies above, we next devise practical distributed algorithms that solve the problems **SubD1** and **SubD2** in a dynamic setting. The solution is divided into two parts: *Traffic splitter* associated with **SubD1** distributes the data bits from the session queue to virtual link queues, and *link scheduler* associated with **SubD2** achieves the stationary distribution (13) for the virtual link activation vectors.

Traffic splitter can be performed at each link in a distributed fashion with only local information; $Q_l(t)$ and $Q_{lv}(t)$, where $Q_l(t)$ denotes the length of session queue at link l and $Q_{lv}(t)$ denotes the length of the v-th virtual link queue at link l at time slot t. At each time slot t, the internal service rate $\mu_{lv}(t)$ is set to either 0 or constant C of equation (12), which is larger than $V \cdot \max_{l,v} c_{lv}$. The detailed algorithm is outlined in Algorithm 1. Basically, the algorithm implements (12). First, each physical link finds the virtual queue with the smallest queue length. Then, if the session queue has a larger queue length than this smallest virtual link queue, it transfers backlogged data bits to the virtual link queue by at most C bits. Otherwise, it does not transfer data during t.

To describe our distributed *link scheduler*, we begin with the time slot structure for scheduling in our time-slotted system. We divide a single slot into a *control* subslot and a *data* subslot, as shown in Fig. 3. During the control subslot, which is further divided into RTS, CTS, and REJECT parts, nodes exchange control packets with each other to determine

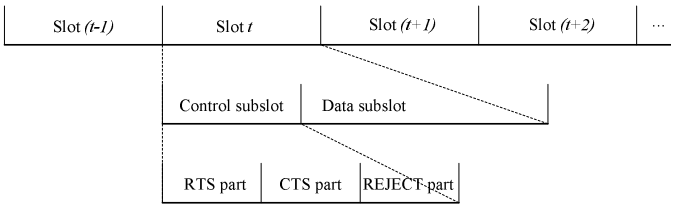

Fig. 3. Time slot structure.

the set of active virtual links distributedly. Then during the data subslot, a sender can transmit data to the corresponding receiver using the activated virtual link. We assume the control subslot is substantially small compared to the data subslot and thus throughput loss due to the control subslot is negligible. We also ignore the processing time and the overhead from the packetization of data bits.

A key step is to choose a feasible virtual link activation vector $\mathbf{x}(t)$ in a distributed manner such that its distribution follows (13). To this end, we combine two virtual link activation vectors, i.e., the virtual link activation vector $\mathbf{x}(t-1)$ at the previous slot $t-1$ and a randomly chosen vector at the beginning of time t, to determine a feasible virtual link activation vector at time slot t. A challenge here is to ensure that the resultant virtual link activation vector is feasible under the SINR interference model. We solve the problem by *three-way handshake* of control packets: Ready-To-Send (RTS), Clear-To-Send (CTS), and REJECT.

The overall procedure for distributed link scheduling can be described as follows. At the beginning of time slot t, each link decides if it changes its activity or not with some probability. Once a link wants to change its activity state, it randomly chooses a virtual link. The set of the selected virtual links in the network is called as the *decision vector*. Then the active virtual links in $\mathbf{x}(t-1)$[3] and a subset of virtual links in the decision vector participate in feasibility tests by exchanging control packets. As depicted in Fig. 3, the transmitter of each participating virtual link first sends an RTS in the control subslot. The RTS (and also CTS) packet contains link *id* l and virtual link *id* v to be activated. The receiver of the RTS returns an CTS in the CTS part, only if the received SINR is large enough to support the designated virtual link. The CTS packet is also used to test if the SINR of the reverse link is acceptable. If any active virtual link in $\mathbf{x}(t-1)$ fails to meet the required SINR, it broadcasts a REJECT signal in the REJECT part and invalidates the new link activity state. If no REJECT signal is broadcasted, we obtain new schedule $\mathbf{x}(t)$ from the virtual links participated in the feasibility tests, and otherwise, the schedule remains unchanged, i.e., $\mathbf{x}(t) = \mathbf{x}(t-1)$. Once $\mathbf{x}(t)$ is determined, during the following data subslot, the virtual links in $\mathbf{x}(t)$ transmit data packets.

Detailed descriptions for the selection of the decision vector and the feasibility tests using the three-way handshake are as follows. (Also refer to Algorithms 2 and 3, where $\mathrm{Tx}(l)$ and $\mathrm{Rx}(l)$ denote the transmitter and the receiver of link l, respectively.) At the beginning of the control subslot, each link decides whether it changes the activity in a probabilistic way. This probability $p_{trial}(>0)$, called the *trial probability*, can be different for each link. A link l, if it is to change the activity, chooses a virtual link at random. The selection probability need not be uniform for all the virtual links in the link. However, each virtual link should have a non-zero probability, and the probability should be fixed across time slots. We denote the decision vector by $\mathbf{m}(t) := (m_{lv}(t))$, where $m_{lv}(t) \in \{0,1\}$ specify the selected virtual links, i.e., $m_{lv}(t) = 1$ if virtual link $[l,v]$ is chosen, and $m_{lv}(t) = 0$ otherwise. The virtual links in the decision vector can potentially change its activity from 'active' to 'inactive' or vice versa. If no virtual link of link l was active at time $t-1$, virtual link $[l,v]$ in the decision vector will be active with probability p_{lv} $(0 < p_{lv} < 1)$ at time t, and remains silent with probability $\bar{p}_{lv} := 1 - p_{lv}$. If a virtual link of link l was active at time $t-1$, virtual link $[l,v]$ in the decision vector checks whether it is the active virtual link during the previous time slot. If it is, it remains active with probability p_{lv} and will be inactive with probability $\bar{p}_{lv}$. If it is not, the virtual link in the decision vector remains silent and the previously active *sibling* virtual link will be active in the current slot, where we say two virtual links are siblings if they belong to the same physical link.

Now we have a set of virtual link candidates that can potentially be active during time slot t, which consist of the virtual links in the decision vector that decides to be active and the virtual links in $\mathbf{x}(t-1)$ that are not in the decision vector. We proceed to test feasibility of these virtual links candidates. We call the candidates that were not active at $t-1$ as *new applicants*. To represent new applicants, we use the indicator function $n_{lv}(t)$ and its vector $\mathbf{n}(t) := (n_{lv}(t))$. Naturally, $n_{lv}(t) = 1$ when virtual link $[l,v]$ is a new applicant, and $n_{lv}(t) = 0$ otherwise. All the active virtual links in $\mathbf{x}(t-1)$ as well as all the new applicants in $\mathbf{n}(t)$ participate in the feasibility test. Note that some in $\mathbf{x}(t-1)$ are not candidates, but participate in the test. The inclusion of all active virtual links of $\mathbf{x}(t-1)$ in the test is crucial to ensure the reversibility of Markov chain described in the next section.

In the RTS part of the control subslot, each participating virtual link's transmitter sends an RTS, and its corresponding receiver measures the SINR from the RTS. Suppose that a virtual link $[l,v]$ transmits an RTS. If the receiver passes the test, i.e., if the SINR is large enough to support virtual link $[l,v]$, the receiver returns a CTS to the transmitter in the subsequent CTS part. If the receiver fails the test, i.e., if the SINR requirement is violated, the receiver takes a different action depending on its activity during the previous time slot. If virtual link $[l,v]$ was not active at time $t-1$ (equivalently, if virtual link $[l,v]$ is in $\mathbf{n}(t)$), the receiver does nothing[4], and it remains inactive during time t. If virtual link $[l,v]$ was active at time $t-1$ (equivalently, if virtual link $[l,v]$ is in $\mathbf{x}(t-1)$), the receiver will broadcast a REJECT signal[5] in the REJECT part later, and invalidates the new activation vector. This ends the *forward* feasibility test by RTS.

Note that the above feasibility test is conservative, since not

[3]We say that virtual link $[l,v]$ is *in the vector*, e.g., $\mathbf{x}$ if the associated indicator function x_{lv} is 1.

[4]Line 22 in Algorithm 2.

[5]Lines 24 and 32 in Algorithm 3.

TABLE I
TYPES OF CONTROL PACKETS AND SIGNAL THAT CAN BE GENERATED BY LINK l. (YES) IMPLIES THAT THE CONTROL PACKET OR SIGNAL WILL BE GENERATED CONDITIONALLY, BASED ON THE PASS OR FAIL OF THE FEASIBILITY TESTS.

	RTS	CTS	REJECT
$\exists v$ s.t. $[l,v] \in \mathbf{x}(t-1), [l,v] \notin \mathbf{n}(t)$	Yes	(Yes)	(Yes)
$\exists v$ s.t. $[l,v] \notin \mathbf{x}(t-1), [l,v] \in \mathbf{n}(t)$	Yes	(Yes)	No
$\forall v$ s.t. $[l,v] \notin \mathbf{x}(t-1), [l,v] \notin \mathbf{n}(t)$	No	No	No

all the senders of RTS will transmit data during the data subslot (even when there are no REJECT signals in the REJECT part): Some virtual links in $\mathbf{n}(t)$ may fail the feasibility test and stay inactive during the data subslot, and some virtual links that were active in $\mathbf{x}(t-1)$ will voluntarily give up being active and turn off their radio signal. This implies the receivers of active links may achieve higher SINR than that in the feasibility test.

In the CTS part, the *reverse* feasibility test, which is similar to the forward feasibility test of the RTS, proceeds and the transmitter checks whether it receives a SINR higher than Γ_{ctrl} and successfully decodes the CTS. If the transmitter passes the test and receives no REJECT signal before the data subslot, it will transmit data bits during the data subslot. If the transmitter fails the test, depending on its activity during the previous time slot, the transmitter remains silent[6] (if $n_{lv}(t) = 1$), or broadcasts a REJECT signal[7] in the following REJECT part (if $x_{lv}(t-1) = 1$). This reverse feasibility test is also conservative for the same reason as in the forward feasibility test.

In the REJECT part, the transmitter and receiver of links in $\mathbf{x}(t-1)$ may send REJECT signals if the SINR requirement is violated. If a REJECT signal is broadcasted, all the links will set their activities to the value of the previous time slot, i.e., $x_{lv}(t) = x_{lv}(t-1)$ for all l, v. In this sense, the REJECT signal *invalidates* the new virtual link activation vector. Since the REJECT signal does not necessarily carry any specific information, it would suffice to receive a signal with high enough strength compared to the background noise level. Similar ideas can be found in a different context of wireless local area networks [27] and wireless mesh networks [28]. In a small network, we can assume that all nodes perceive a signal from any node. In a larger network, we may need to employ a *flooding* method to relay the signal over the network, inevitably with additional control overheads. However, since the signal does not carry data, multiple nodes can transmit the REJECT signals at the same time, which enables quick propagation of the signal throughout the network. In this work, we assume that the REJECT signal is quickly relayed to all nodes in the network within the period of the REJECT part. Detailed implementation of the signal flooding is beyond the scope of the paper.

Finally, in the data subslot, virtual link $[l,v]$ transmits data bits at rate c_{lv} if $x_{lv}(t) = 1$. Table I summarizes the types of control packets that can be generated by a link during the control subslot, based on its membership in $\mathbf{x}(t-1)$ and $\mathbf{n}(t)$. A virtual link cannot belong to both $\mathbf{x}(t-1)$ and $\mathbf{n}(t)$ at the same time. Note that i) if a receiver sends a CTS packet, it will not broadcast a REJECT signal, ii) it is possible for both the transmitter and receiver of a link to broadcast REJECT signals simultaneously, and iii) new applicants are not allowed to generate REJECT signals, which is the key difference between Algorithms 2 and 3.

Note that the flooding-like control (i.e., REJECT signal) incurs additional overhead. However, a certain level of global information is required for scheduling under the SINR-based interference model, since even a small change of the transmission power may cause a constraint violation of a far-away transmission. In this work, we reduce the amount of control overhead for link activity coordination under the SINR model to a minimal level. Exchanging control messages often incurs significant overhead due to inclusion of addresses, delimiters, checksum, etc, in each message. We avoid such large overhead by using a simple 'signal', which carries only a single bit information. Thus it can be easily implemented without much complexity, even without decoding, e.g., by using energy detection, and can propagate through the network quickly by simply repeating. All the other complex operations have been designed to operate in a distributed manner.

D. Throughput optimality of distributed link scheduler

In this section, we show that the proposed link scheduler indeed achieves optimal throughput. Specifically, we show that the scheduler yields the same steady-state probability distribution as (13), which implies an optimal solution to **SubD2**. We represent the set of active virtual links in a virtual link activation vector as the corresponding capital letter. For example, we denote the decision *set* associated with decision *vector* $\mathbf{m} = (m_{lv})$ by $M := \{\forall [l,v] \mid m_{lv} = 1\}$.

We begin with some properties of the stochastic process $\{\mathbf{x}(t)\}_{t=0}^{\infty}$.

Lemma 1: $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ is a Discrete-Time Markov Chain (DTMC).

Proof: The inputs of Algorithms 2 and 3 are the previous virtual link activation vector $\mathbf{x}(t-1)$, trial probability (p_{trial}), activation probabilities (p_{lv}), and the fixed probabilities for constructing the decision vector. Since the current state does not depend on the history before time $t-1$, the discrete time process $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ is a Markov Chain. ■

The following two lemmas assert that the Markov Chain walks through all and only the feasible virtual link activation vectors of the network.

Lemma 2: With a feasible initial virtual link activation vector, e.g., $\mathbf{x}(0) = \mathbf{0}$, the virtual link activation vector $\mathbf{x}(t)$ under our proposed scheduler is feasible, i.e., $\mathbf{x}(t) \in F$, for all $t \geq 0$.

Proof: A new set of virtual links is permitted to be active at the same time only if it passes both the forward and reverse feasibility tests. The tests guarantee that each virtual link of the new activation vector satisfies the SINR requirements. Moreover, the proposed algorithm allows for only a single virtual link in a physical link to be active in a time slot.

[6]Line 25 in Algorithm 2.
[7]Lines 27 and 32 in Algorithm 3.

Algorithm 2 Link scheduling for previously *inactive* link l.

1) At the beginning of control subslot:
Decide if link l will try to change its activity with probability p_{trial}.
if Link l is to change the activity **then**
  Select $v^* \in \{1, 2, ..., V\}$ at random.
  $x_{lv^*}(t) = 1$ with probability p_{lv^*};
  $x_{lv^*}(t) = 0$ with probability $\bar{p}_{lv^*} = 1 - p_{lv^*}$.
  $x_{lv}(t) = x_{lv}(t-1)$ for $v \neq v^*$.
else
  $x_{lv}(t) = x_{lv}(t-1)$ for all v's.
end if

2) In RTS part of control subslot:
if $x_{lv}(t) = 1$ **then**
  Tx(l) sends RTS(l, v).
  Rx(l) measures SINR$_l$ from RTS(l, v).
end if

3) In CTS part of control subslot:
if SINR$_l \geq \Gamma_v$ **then**
  Rx(l) sends CTS(l, v).
else
  $x_{lv}(t) = 0$.
end if
if Tx(l) fails to decode CTS(l, v) **then**
  $x_{lv}(t) = 0$.
end if

4) In REJECT part of control subslot:
if Tx(l) detects REJECT **then**
  $x_{lv}(t) = x_{lv}(t-1)$ for all v's.
end if

5) In data subslot:
if $x_{lv}(t) = 1$ **then**
  Tx(l) transmits data bits at the rate of c_{lv}.
end if

Algorithm 3 Link scheduling for previously *active* link l.

1) At the beginning of control subslot:
Decide if link l will try to change its activity with probability p_{trial}.
if Link l is to change the activity **then**
  Select $v^* \in \{1, 2, ..., V\}$ at random.
  if $x_{lv^*}(t-1) = 1$ **then**
    $x_{lv^*}(t) = 1$ with probability p_{lv^*};
    $x_{lv^*}(t) = 0$ with probability $\bar{p}_{lv^*} = 1 - p_{lv^*}$.
  else
    $x_{lv^*}(t) = 0$.
  end if
  $x_{lv}(t) = x_{lv}(t-1)$ for $v \neq v^*$.
else
  $x_{lv}(t) = x_{lv}(t-1)$ for all v's.
end if

2) In RTS part of control subslot:
Tx(l) sends RTS(l, v).
Rx(l) measures SINR$_{lv}$ from RTS(l, v).

3) In CTS part of control subslot:
if SINR$_l \geq \Gamma_v$ **then**
  Rx(l) sends CTS(l, v).
else
  Rx(l) constructs and holds REJECT.
end if
if Tx(l) fails to decode CTS(l, v) **then**
  Tx(l) constructs and holds REJECT.
end if

4) In REJECT part of control subslot:
if Tx(l) or Rx(l) is holding REJECT **then**
  REJECT is broadcasted.
end if
if Tx(l) detects REJECT **then**
  $x_{lv}(t) = x_{lv}(t-1)$ for all v's.
end if

5) In data subslot:
if $x_{lv}(t) = 1$ **then**
  Tx(l) transmits data bits at the rate of c_{lv}.
end if

Therefore, the selected virtual link activation vector should be feasible if it passes the feasibility tests. If the new set is rejected, then the activation vector at time t should be also feasible since $\mathbf{x}(t) = \mathbf{x}(t-1)$. Therefore, $\mathbf{x}(t)$ is feasible for all $t \geq 0$ by induction. ■

Lemma 3: In $\{\mathbf{x}(t)\}_{t=0}^{\infty}$, a state $\mathbf{x}$ can reach any $\mathbf{x}' \in F$ with positive probability in a finite number of steps.

Proof: We show that by the proposed scheduling, the null state $\mathbf{0}$ can reach an arbitrary feasible link activation vector with some positive probability in a single step, and vice versa, where the null state implies that all links in the network are inactive.

1) Suppose that the network is currently in the null state. For any $\mathbf{x}' \in F$, if the decision set coincides with X' and all the virtual links in the decision set determine to be active, then they will pass the feasibility tests and clearly, $\mathbf{x}'$ is accepted as the next virtual link activation vector.
2) Conversely, suppose that $\mathbf{x}$ is the current virtual link activation vector. If the decision set coincides with X and all the virtual links in the decision set intend to turn off, the null state will become the virtual link activation vector of the next slot.

From 1) and 2), a state $\mathbf{x}$ can reach any $\mathbf{x}' \in F$ possibly via the null state with a positive probability in a finite number of steps. ■

We next show that the corresponding Markov Chain has the same distribution as (13) under our proposed scheduler. Since the Markov Chain is aperiodic and irreducible from Lemma 3, it has a unique stationary distribution. Focusing on the transition of two virtual link activation vectors, we

characterize the probability distribution that satisfies detailed balance conditions. Then the reversibility of the Markov Chain indicates that it is the stationary distribution.

We consider a state transition at time slot t, and omit the parameter t (or $t-1$) if there is no confusion. If any REJECT signals are transmitted during the control subslot of time slot t, then there is no state transition. Henceforth, we consider only the case when new virtual link activation vector is accepted. Suppose that at time slot t, the virtual link activation vector changes from $\mathbf{x}$ to $\mathbf{x}'$. Let X and X' denote the set of all the active virtual links in $\mathbf{x}$ and $\mathbf{x}'$, respectively. Let M denote the decision set chosen in the process of the transition. Note that for virtual link $[l,v] \in X$, all its sibling virtual links are *blocked* from being active at time slot t under our scheduling scheme, i.e., $[l,w] \notin X'$ for $w \neq v$. (See lines $3-14$ in Algorithm 3.) We denote the blocked virtual links by $B([l,v])$, and extend the notation by letting $B(A)$ denote the set of virtual links blocked by the set of virtual links A. Further, N represents the set of new applicants for activation, i.e., virtual links that were not active in the previous time slot but participate in the feasibility tests in this time slot. Let subset $D \subset N$ denote the set of the virtual links discouraged from being activated in the process of the feasibility tests, i.e., the new applicants that could not meet the SINR requirement.

In this setting, the virtual links in the network can be classified as shown in Table II, based on their action in the time slot. Each virtual link belongs to either the decision set M or its complement M^c. The virtual links in M can be further subdivided as follows. We also provide a diagram in Fig. 4 to help readers understand the table, where the set M is represented by grey areas and the set N is by slanted areas, respectively.

- $M_{X\setminus X'}$: Virtual links that were active at $t-1$ and are inactive at t. These links will exchange RTS/CTS in the control subslot but do not transmit data in the data subslot. Under our scheduling algorithms, for a virtual link to change its state from 'active' to 'inactive', it should be included in the decision set M. Hence, $X \setminus X' \subset M$, and thus $M_{X\setminus X'} = X \setminus X'$.
- $M_{X\cap X'}$: Virtual links that were active at $t-1$ and remain active at t. These links will exchange RTS/CTS and do transmit data bits in the data subslot. They can be also written as $M_{X\cap X'} = M \cap (X \cap X')$.
- $M_{X'\setminus X}$: Virtual links that were inactive at $t-1$ and are active at t. These links are new applicants, participate in RTS/CTS exchange, and transmit data in the data subslot. From $X' \setminus X \subset M$, we have that $M_{X'\setminus X} = X' \setminus X$.
- $M_{B(X\cup X')}$: Virtual links in M that are blocked to be active at t due to some virtual link in $X \cup X'$. They can be written as $M_{B(X\cup X')} = M \cap B(X \cup X')$.
- M_D: Virtual links that fail the feasibility tests. All the virtual links in this set are new applicant since the state transition is assumed to occur. They may fail either the forward or reverse feasibility test, hence they can exchange either RTS only or RTS/CTS depending on cases. From $D \subset N \subset M$, we have $M_D = D$.
- M_{others}: The rest virtual links in $M \setminus ((X \cup X') \cup B(X \cup X') \cup D)$ that voluntarily give up trying to be active with probability $\bar{p}_{lv}$. They do not participate in the feasibility tests, and thus do not contribute to the state transition.

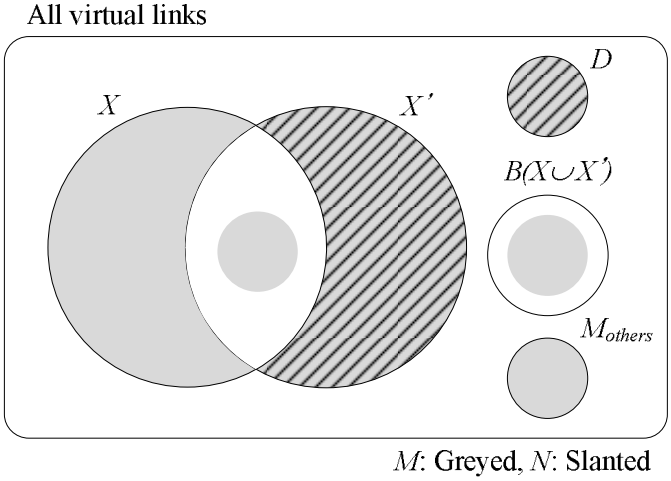

Fig. 4. An illustration for virtual link classification.

For a virtual link in M^c, if the link was active at $t-1$, it will participate in the RTS/ CTS exchange and remain active at t. We denote these virtual links by $(M^c)_X$, which equals to $M^c \cap X$. Since $X \setminus X'$ does not share any members with M^c, i.e., $M^c \cap (X \setminus X') = \emptyset$, we can rewrite it as $(M^c)_X = M^c \cap (X \cap X')$. On the other hand, if a virtual link in M^c was not active at $t-1$, it does not exchange the control packets and remains silent.

Given X and X', a decision set that triggers a transition from X to X' may not be unique. Let $\mathcal{M}(X,X')$ denote the collection of all such decision sets that can make the transition. Also, for a decision set $M \in \mathcal{M}(X,X')$, there can be multiple sets of new applicants that incur the transition from X to X'. Since N consists of two disjoint sets $X' \setminus X$ and D, it suffices to determine N using D, X, and X'. Hence, we describe the transition from X to X' using D instead of N. Let $\mathcal{D}(X,X';M)$ denote the collection of all possible D's that make the transition from X to X' given M.

Now, we calculate the transition probability from X to X' as follows. Let $\beta(M)$ denote the probability of a decision set $M \in \mathcal{M}(X,X')$ being chosen randomly. Each virtual link in $M_{X\setminus X'}$ switches to the inactive state voluntarily with probability $\bar{p}_{lv}$ and each virtual link in $M_{X\cap X'}$ stays active with probability p_{lv}. Among M, virtual links in $B(X \cup X')$ are blocked (with probability 1) and does not affect the state transition probability. Lastly, each of the rest virtual links in $M \setminus (X \cup B(X \cup X'))$ becomes a new applicant with probability p_{lv} or voluntarily remains inactive with probability $\bar{p}_{lv}$. All the new applicants try to be activated in the current slot. Among them, virtual links in $M_{X'\setminus X}$ pass the feasibility tests, while the others in $D \in \mathcal{D}(X,X';M)$ fail the tests and are not allowed to be active. We have denoted the virtual links that voluntarily give up by M_{others}. We emphasize the dependency of M_{others} on D by $M_{others}(D)$. Naturally, virtual links of M^c have the same activity states as the previous time slot with probability 1. Hence, the state transition probability from $\mathbf{x}$ to $\mathbf{x}'$ can be obtained as (16) in Fig. 5.

The following lemma and corollary lead to the main result

TABLE II
CLASSIFICATION OF VIRTUAL LINKS.

Sets of virtual links		Control packets	Data transmission	Equivalent form
M	$M_{X\setminus X'}$	RTS/CTS	·	$X \setminus X'$
	$M_{X\cap X'}$	RTS/CTS	Yes	$M \cap (X \cap X')$
	$M_{X'\setminus X}$	RTS/CTS	Yes	$X' \setminus X$
	$M_{B(X\cup X')}$	·	·	$M \cap B(X \cup X')$
	M_D	RTS or RTS/CTS	·	D
	M_{others}	·	·	$M \setminus ((X \cup X') \cup B(X \cup X') \cup D)$
M^c	$(M^c)_X$	RTS/CTS	Yes	$M^c \cap (X \cap X')$
	$(M^c)_{others}$	·	·	$M^c \setminus (X \cap X')$

$$\begin{aligned} P(\mathbf{x},\mathbf{x}') &= \sum_{M\in\mathcal{M}(X,X')} \left(\beta(M) \prod_{[l,v]\in M_{X\setminus X'}} \bar{p}_{lv} \prod_{[l,v]\in M_{X\cap X'}} p_{lv} \prod_{[l,v]\in M_{X'\setminus X}} p_{lv} \cdot \sum_{D\in\mathcal{D}(X,X';M)} \Big(\prod_{[l,v]\in D} p_{lv} \prod_{[l,v]\in M_{others}(D)} \bar{p}_{lv} \Big) \right) \\ &= \sum_{M\in\mathcal{M}(X,X')} \left(\beta(M) \prod_{[l,v]\in X\setminus X'} \bar{p}_{lv} \prod_{[l,v]\in M\cap(X\cap X')} p_{lv} \prod_{[l,v]\in X'\setminus X} p_{lv} \cdot \sum_{D\in\mathcal{D}(X,X';M)} \Big(\prod_{[l,v]\in D} p_{lv} \prod_{[l,v]\in M_{others}(D)} \bar{p}_{lv} \Big) \right). \end{aligned} \tag{16}$$

$$\begin{aligned} P(\mathbf{x}',\mathbf{x}) &= \sum_{M\in\mathcal{M}(X',X)} \left(\beta(M) \prod_{[l,v]\in X'\setminus X} \bar{p}_{lv} \prod_{[l,v]\in M\cap(X'\cap X)} p_{lv} \prod_{[l,v]\in X\setminus X'} p_{lv} \cdot \sum_{D\in\mathcal{D}(X',X;M)} \Big(\prod_{[l,v]\in D} p_{lv} \prod_{[l,v]\in M_{others}(D)} \bar{p}_{lv} \Big) \right) \\ &= \sum_{M\in\mathcal{M}(X,X')} \left(\beta(M) \prod_{[l,v]\in X\setminus X'} p_{lv} \prod_{[l,v]\in M\cap(X\cap X')} p_{lv} \prod_{[l,v]\in X'\setminus X} \bar{p}_{lv} \cdot \sum_{D\in\mathcal{D}(X,X';M)} \Big(\prod_{[l,v]\in D} p_{lv} \prod_{[l,v]\in M_{others}(D)} \bar{p}_{lv} \Big) \right). \end{aligned} \tag{17}$$

Fig. 5. Forward and reverse state transition probabilities.

of this section.

Lemma 4: $\mathcal{M}(X,X') = \mathcal{M}(X',X)$.

Proof: Suppose that state X changes to X' with decision schedule $M \in \mathcal{M}(X,X')$. The virtual links of M can be classified as follows.

- Virtual links in $M_{X\cap X'}$ stays active.
- Virtual links in $M_{X\setminus X'}$ becomes in active.
- Virtual links in $M_{X'\setminus X}$ becomes active.
- Virtual links in $M_{B(X\cup X')}$ remains inactive since blocked.
- The set of the rest virtual links is denoted by M_{rest}.

We now focus on M_{rest}. Among these virtual links, we denote the set of virtual links that voluntarily become inactive by M_{others}, and the others should be inactive by failing the feasibility test, in order to make the state transition from X to X', and denoted by M_D, i.e., $M_{rest} = M_{others} \cup M_D$. Note that the virtual links participate in the feasibility test under our algorithm include X, $X' \setminus X$, and $M_D = M_{rest} \setminus M_{others}$, and all the virtual links in M_D fail the feasibility test.

Let us consider the reverse transition, i.e., from X' to X. Suppose that we have the same decision schedule $M' = M$ and the same set $M'_{others} = M_{others}$ of virtual links becomes inactive voluntarily. Since $X \cup X' \subset M' \setminus M'_{others}$, with a certain probability:

- Virtual links in $M_{X'\cap X}$ stays active.
- Virtual links in $M_{X'\setminus X}$ becomes in active.
- Virtual links in $M_{X\setminus X'}$ becomes active.
- Virtual links in $M_{B(X'\cup X)}$ remains inactive since blocked.
- The set of the rest virtual links is denoted by M'_{rest}.

In this case, we have $M'_{rest} = M_{rest}$ because $M' = M$ and $M'_{others} = M_{others}$. Also, the set of the virtual links participate in the feasibility test would be $X' \cup (X \setminus X') \cup (M'_{rest} \setminus M'_{others})$, which is identical to the set of virtual links participated in the feasibility test for transition from X to X'. Hence, due to the identical interference relationship, all the virtual links in $M'_{rest} \setminus M'_{others} = M_{rest} \setminus M_{others}$ should fail the feasibility test. This completes the transition from X' to X. ■

The proof of Lemma 4 immediately implies that in both transitions, the set of the virtual links that fail the feasibility test is also identical, which leads to the following corollary.

Corollary 1: $\mathcal{D}(X,X';M) = \mathcal{D}(X',X;M)$.

We now can obtain the following proposition.

Proposition 1: The probability distribution

$$\pi(\mathbf{x};\mathbf{q_v}) = \frac{1}{Z} \prod_{[l,v]\in X} \frac{p_{lv}}{\bar{p}_{lv}}, \tag{18}$$

where Z is the normalization factor, is the stationary state distribution of the Markov Chain $\{\mathbf{x}(t)\}_{t=0}^{\infty}$. Further, by setting $p_{lv} = \frac{\exp(c_{lv}q_{lv})}{1+\exp(c_{lv}q_{lv})}$, we obtain the state probability

$$\pi(\mathbf{x};\mathbf{q_v}) = \frac{1}{Z} \exp\big(\sum_{l,v} c_{lv} q_{lv} x_{lv} \big). \tag{19}$$

Proof: We prove the proposition by showing that it satisfies the local balance equation between two arbitrary feasible virtual link activation vectors $\mathbf{x}$ and $\mathbf{x}'$, i.e.,

$$\pi(\mathbf{x})P(\mathbf{x},\mathbf{x}') = \pi(\mathbf{x}')P(\mathbf{x}',\mathbf{x}). \tag{20}$$

If (20) holds, then the DTMC $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ is reversible, which would imply that $\pi(\mathbf{x};\mathbf{q_v})$ is its stationary distribution [29], since it is irreducible and aperiodic from Lemma 3.

For $M \in \mathcal{M}(X,X')$, we have $M \in \mathcal{M}(X',X)$ by Lemma 4. Since the decision set is selected independently of the Markov state, the selection probability is the same $\beta(M)$ for the both transitions.

Also, according to Corollary 1, $\mathcal{D}(X,X';M) = \mathcal{D}(X',X;M)$. Thus, the state transition probability from $\mathbf{x}'$ to $\mathbf{x}$ can be calculated as (17) in Fig. 5.
For given M, D, X, and X', it is clear that $M_{others}(D) = M \setminus ((X \cup X') \cup B(X \cup X') \cup D)$ is the same set of virtual links for both the state transitions, since the set depends on $X \cup X'$ rather than individual sets X or X'. Combining (16), (18) and (17) yields that

$$\pi(\mathbf{x})P(\mathbf{x},\mathbf{x}') = \pi(\mathbf{x}')P(\mathbf{x}',\mathbf{x}). \tag{21}$$

■

Revisiting the adaptation algorithm (15) reveals that the parameter $q_{lv}(t)$ is a scaled version of queue length of virtual link $[l,v]$. Thus, in setting the probability p_{lv} in Proposition 1, we can replace the Lagrangian multiplier $q_{lv}(t)$ with the actual queue length $Q_{lv}(t)$. Actually we can use any function $f_{lv}(Q_{lv}(t))$ instead of $q_{lv}(t)$ to achieve the optimal throughput, if $f_{lv}(q)$ satisfies the following two conditions [20], [30]:

1) $f_{lv}(q)$ is a non-decreasing continuous function with $\lim_{q\to\infty} f_{lv}(q) = \infty$.
2) Given any M_1, $M_2 > 0$ and $0 < \epsilon < 1$, there exists $Q(<\infty)$ such that for $q > Q$,

$$(1-\epsilon)f_{lv}(q) \le f_{lv}(q-M_1) \le f_{lv}(q+M_2) \le (1+\epsilon)f_{lv}(q). \tag{22}$$

We now set probability p_{lv} to

$$p_{lv}(t) = \frac{\exp(c_{lv}f_{lv}(Q_{lv}(t)))}{1+\exp(c_{lv}f_{lv}(Q_{lv}(t)))} \tag{23}$$

with function $f_{lv}(q)$ that satisfies the above conditions, and obtain the product-form stationary distribution (18), which immediately implies queueing stability under *time-scale separation assumption* [20]. The assumption means that the probability $p_{lv}(t)$ is relatively constant with respect to convergence speed of the Markov Chain. It has been shown in recent works [31], [32] that 'without time-scale separation assumption', the system is stable for $f_{lv}(q) = \log\log(q)$ and for $f_{lv}(q) = \log(q)/g(q)$ with some slowly increasing function $g(q)$. Other empirical results also suggest that the time-scale separation assumption is still valid for $f_{lv}(q) = \log(q)$ [20]. In the rest of the paper, we will use the function $f_{lv}(q) = \log(1+kq)$ with some constant k.

Remarks:

- We have assumed that each physical link has a single session queue and V virtual link queues. This can be extended to multiple sessions per physical link case by increasing the number of queues in proportion to the number of sessions.
- In choosing the set of active links, we take a different approach from the traditional CSMA/CA protocol, where the random back-off and the carrier-sensing functionality are combined together. We first select a decision vector using probability p_{trial}, which is similar to the random back-off of $1/p_{trial}$ in the CSMA/CA protocol, and obtain next schedule while checking its feasibility with the carrier-sensing functionality. This two-step approach is necessary to obtain the desired balance equations of Markov Chain, and for the same reason, even the link with no queue still needs to be selected in the decision schedule.
- Multiple control subslots can be adopted in a single slot to improve delay performance. We recall that the stochastic process $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ describes the progress of virtual link activation vectors under our scheduling policy. With S control subslots in a slot, the evolution of the virtual link activation vectors will be represented by $\{\mathbf{x}(S(t+1)-1)\}_{t=0}^{\infty}$, which is a DTMC because it is a periodic sampling of another DTMC $\{\mathbf{x}(t)\}_{t=0}^{\infty}$ [29]. Repeating the steps of this section, it follows that our link scheduler is still throughput optimal with multiple control subslots.

IV. Simulation Studies

In this section, we evaluate the performance of the proposed scheduling algorithm from the perspective of throughput and delay through simulations.

A. Simulation setup

We consider a network in a square area with unit-length sides. We place total 12 nodes in the area, and connect any two nodes by a link if their distance is no greater than 0.5. We divide the area into four small squares of 0.5×0.5 as shown in Fig. 6 (dotted lines), and in each subsquare, three nodes are placed at random, reducing the likelihood of network partitioning. We identify the links by *link id*, which is shown beside each link. The links are two-way, but one-way traffic sessions are generated for each link. The direction of traffic is randomly chosen and shown by an arrow at each link. In our simulations, every link has a session, and its reverse link is implicitly assumed to exist for feedback.

The transmitting node of a link, when active, sends signal at unit power, and the signal attenuates with the path loss exponent of 4. At the receiving node, the background noise is assumed to be negligible. Each link supports various transmission rates as shown in Table III, depending on the received SINR.

We generate traffic as follows. Let $\hat{\mathbf{x}}$ denote a link activation vector, and let $\mathbf{r}(\hat{\mathbf{x}})$ denote the link transmission rate vector associated with $\hat{\mathbf{x}}$. We arbitrarily choose the following eight activation vectors,

$$\begin{array}{ll} \hat{\mathbf{x}}_1 = (1,10,13,18), & \hat{\mathbf{x}}_2 = (1,12,17,21), \\ \hat{\mathbf{x}}_3 = (2,13,16,21), & \hat{\mathbf{x}}_4 = (3,8,13,19), \\ \hat{\mathbf{x}}_5 = (4,9,11,22), & \hat{\mathbf{x}}_6 = (5,14,17,23), \\ \hat{\mathbf{x}}_7 = (6,11,16,23), & \hat{\mathbf{x}}_8 = (7,10,13,20), \end{array} \tag{24}$$

TABLE III
TRANSMISSION RATES OF LINK l.

Virtual link id	SINR requirement	Transmission rate (units per slot)
1	$\gamma_l \in [0.5, 1.0)$	1.0
2	$\gamma_l \in [1.0, 2.0)$	2.0
3	$\gamma_l \in [2.0, 4.0)$	3.0
4	$\gamma_l \in [4.0, \infty)$	4.0

where each vector includes four active links. For each vector, we estimate the SINR at the receiver, and can map it to a transmission rate using Table III. The obtained link transmission rate vectors present as follows.

$$\begin{aligned} \mathbf{r}(\hat{\mathbf{x}}_1) &= (0,2,4,4), \quad \mathbf{r}(\hat{\mathbf{x}}_2) = (0,4,4,4), \\ \mathbf{r}(\hat{\mathbf{x}}_3) &= (4,4,3,4), \quad \mathbf{r}(\hat{\mathbf{x}}_4) = (1,0,3,2), \\ \mathbf{r}(\hat{\mathbf{x}}_5) &= (0,2,0,4), \quad \mathbf{r}(\hat{\mathbf{x}}_6) = (2,3,4,4), \\ \mathbf{r}(\hat{\mathbf{x}}_7) &= (0,1,0,4), \quad \mathbf{r}(\hat{\mathbf{x}}_8) = (1,3,4,3), \end{aligned} \tag{25}$$

where each element is the transmission rate of corresponding active link in (24). We omit the elements of inactive links since they are all zeros. For example, for $\hat{\mathbf{x}}_1$, the vector $\mathbf{r}(\hat{\mathbf{x}}_1)$ shows that link 1 has rate 0, link 10 has rate 2, link 13 has rate 4, link 18 has rate 4, and all the other links have rate 0. Note that any convex combinations of these link transmission rate vectors will be within the capacity region Λ. To guarantee the offered traffic is strictly in the capacity region, we set the arrival rate vector $\boldsymbol{\lambda}$ as

$$\begin{aligned} \boldsymbol{\lambda} = \rho \cdot \big(& 0.1\mathbf{r}(\hat{\mathbf{x}}_1) + 0.2\mathbf{r}(\hat{\mathbf{x}}_2) + 0.1\mathbf{r}(\hat{\mathbf{x}}_3) + 0.2\mathbf{r}(\hat{\mathbf{x}}_4) \\ & + 0.1\mathbf{r}(\hat{\mathbf{x}}_5) + 0.1\mathbf{r}(\hat{\mathbf{x}}_6) + 0.1\mathbf{r}(\hat{\mathbf{x}}_7) + 0.1\mathbf{r}(\hat{\mathbf{x}}_8) \big), \end{aligned} \tag{26}$$

where $\rho \in (0,1)$ denotes the *traffic intensity*. As ρ increases, the arrival rate vector gets closer to the boundary of the capacity region. For a given arrival rates, each session generates data bits according to the geometric distribution at each time slot.

We first simulate our schemes assuming that RTS/CTS packets do not experience transmission errors and REJECT signals are perceived by all the nodes in the network. We later remove these assumptions in our extended simulations . The internal transmission rate μ_{lv} are set sufficiently large to 10^9 (bits per slot) for all virtual links. Each simulation runs for 10^7 time slots to ensure convergence.

B. Performance evaluation

We use the time-scale separable function $f_{lv}(Q_{lv}(t)) = \log(1 + kQ_{lv}(t))$ for simulations. The parameter k affects the system performance: in general, smaller k implies less changes of p_{lv} from (23) and thus a better time-scale separation. Since our analysis relies on the time-scale separation assumption, a smaller k will be preferred to improve throughput performance. Fig. 7 illustrates average queue size for $k = 0.01, 0.001, 0.0001$ with two different numbers of control subslots in a slot, i.e., 10 and 40. The trial probability p_{trial} is set to 0.1 for all the links. The results show that under light traffic, the queue lengths are kept lower for larger k. However, under heavy traffic, the queue length increases more quickly for larger k. In particular, when $k = 0.01$, the queues become unstable for $\rho \geq 0.8$ with 10 control subslots, and for $\rho \geq 0.9$ with 40 control subslots. This is because the time-scale separation assumption no longer holds for $k = 0.01$. Hence, there is a trade-off between the throughput and the delay performance. Hereafter, we will use $k = 0.001$ considering both delay and throughput.

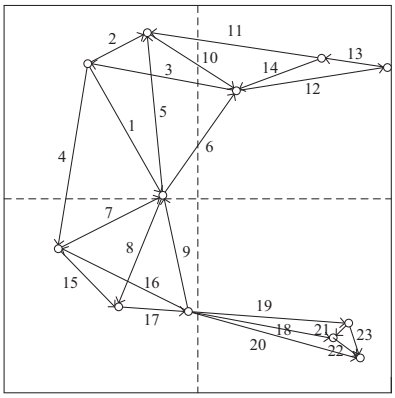

Fig. 6. Network topology for simulations. The number asides of each link is the link id, and the arrow indicates the direction of the link (or the session on it).

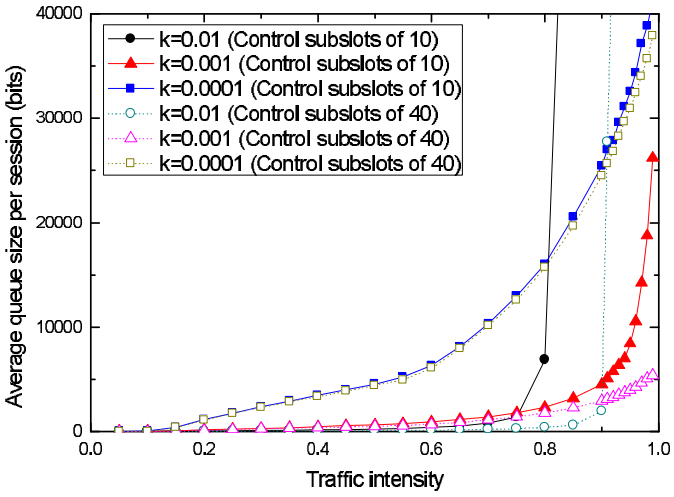

Fig. 7. Impact of time-separable function $f_{lv}(\cdot)$ on the average queue size. The trial probability is set to 0.1.

Next, we investigate the impact of control subslots on the performance. We simulate our schemes changing the number of control subslots from 10 to 40. We set the trial probability p_{trial} to 0.1 for all the links. As the number of control subslots increases, the underlying DTMC have richer connectivity between states, which contributes to improvement in the delay performance as shown in Fig. 8. We also conduct simulations changing the trial probability p_{trial}. The results are shown in Fig. 9. We can attain the best delay performance when $p_{trial} = 0.05$ and $p_{trial} = 0.1$ in our settings. For smaller p_{trial} (≤ 0.01), the state transition is not less likely to occur

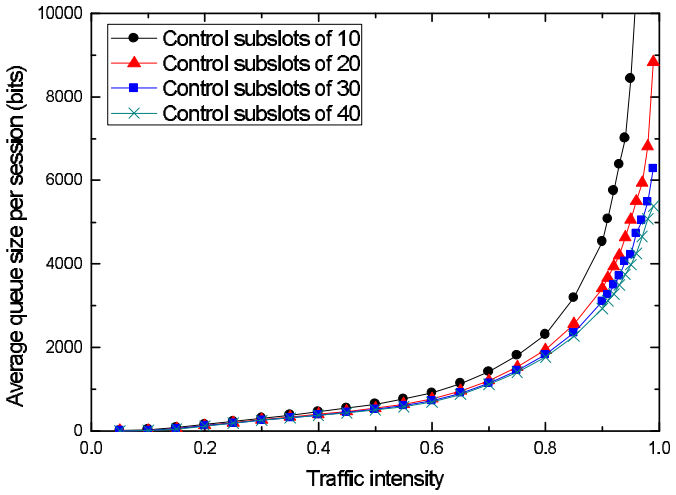

Fig. 8. Impact of the number of control subslots in a slot on the average queue size. The trial probability is 0.1.

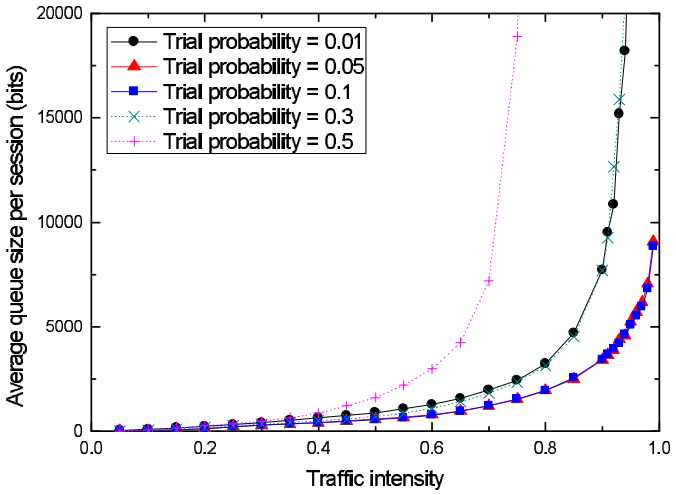

Fig. 9. Impact of the trial probability on the average queue size. The number of control subslots is 20 at each slot.

in the DTMC due to lack of links that want to change its state. This deteriorates the delay performance. On the other hand, when $p_{trial} \geq 0.3$, the transition is less likely to occur due to frequent failures in the feasibility tests. Hence, to improve the delay performance, we have to set the trial probability p_{trail} as well as the number of control subslots accordingly. Precise setting of these parameters remains open for future research.

V. Conclusion

We develop fully distributed link scheduling that achieves throughput optimality under the SINR model. One key idea is to model a single physical link as multiple virtual links with different SINR requirements, and to associate this virtual link model with a two-stage queueing structure. By employing the recently developed CSMA techniques on this queueing structure, we develop practical resource allocation schemes that are amenable to implementation in a distributed manner while achieving the optimal throughput. A fundamental challenge was to ensure that schedule changes (or state transitions) occur between feasible schedules such that each virtual link satisfies its own SINR requirement at any time, without a centralized control. To this end, we develop a novel three-way handshake of control messages and signals. We evaluate the proposed schemes through rigorous analysis, and show that the schemes achieve the same stationary distribution as the optimal solution. The analytical results are verified through extensive simulations under various parameter settings.

References

[1] P. Gupta and P. Kumar, "The capacity of wireless networks," IEEE Trans. Inf. Theory, vol. 46, no. 2, pp. 388-404, Mar. 2000.
[2] IEEE Std 802.16j-2009, "Air interface for broadband wireless access systems – Amendment 1: Multihop relay specification," 2009.
[3] X. Lin, N. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," IEEE J. Sel. Areas Commun., vol. 24, no. 8, pp. 1452-1463, Aug. 2006.
[4] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," IEEE Trans. Autom. Control, vol. 37, no. 12, pp. 1936-1948, Dec. 1992.
[5] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," IEEE Trans. Commun., vol. 47, no. 8, pp. 1260-1267, Aug. 1999.
[6] M. Adnrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting, "Scheduling in a queueing system with asynchronously varying service rates," Probability in the Engineering and Informational Sciences, vol. 18, no. 2, pp. 191-217, Apr. 2004.
[7] M. Neely, E. Modiano, and C. Rohrs, "Dynamic power allocation and routing for time-varying wireless networks," IEEE J. Sel. Areas Commun., vol. 23, no. 1, pp. 89-103, Jan. 2005.
[8] M. Neely, E. Modiano, and C. Li, "Fairness and optimal stochastic control for heterogeneous networks," IEEE/ACM Trans. Netw., vol. 16, no. 2, pp. 396-409, Apr. 2008.
[9] C. Joo, G. Sharma, R. Mazumdar, and N. Shorff, "On the complexity of scheduling in wireless networks," EURASIP Journal of Wireless Communications and Networking, Oct., 2010.
[10] O. Goussevskaia, Y. Oswald, and R. Wattenhofer, "Complexity in geometric SINR," in Proc. *ACM MobiHoc*, 2007.
[11] A. Eryilmaz, A. Ozdaglar and E. Modiano, "Polynomial complexity algorithms for full utilization of multi-hop wireless networks," in Proc. *IEEE INFOCOM*, 2007.
[12] X. Wu, R. Srikant, and J. Perkins, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," IEEE Trans. Mobile Comput., vol. 6, no. 6, pp. 595-605, 2007.
[13] X. Lin and S. Rasool, "Constant-time distributed scheduling policies for ad hoc wireless networks," IEEE Trans. Autom. Control, vol. 54, no. 2, pp. 231-242, Feb. 2009.
[14] C. Joo and N. Shroff, "Performance of random access scheduling schemes in multi-hop wireless networks," IEEE/ACM Trans. Netw., vol. 17, no. 5, pp. 1481-1493, Oct. 2009.
[15] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," IEEE/ACM Trans. Netw., vol. 14, no. 2, pp. 302-315, Apr. 2006.
[16] E. Modiano, D. Shah, and G. Zussman, "Maximizing throughput in wireless networks via gossiping," in Proc *ACM SIGMETRICS*, 2006.
[17] S. Sanghavi, L. Bui, and R. Srikant, "Distributed link scheduling with constant overhead," in Proc. *ACM SIGMETRICS*, 2007.
[18] R. Boorstyn, A. Kershenbaum, B. Maglaris, and V. Sahin, "Throughput analysis in multihop CSMA packet radio networks," IEEE Trans. Commun., vol. 35, no. 3, pp. 267-274, Mar. 1987.
[19] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," IEEE/ACM Trans. Netw., vol. 18, no.3, pp. 960 - 972, Jun. 2010.
[20] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-length based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks," in Proc. *IEEE INFOCOM*, 2010.
[21] A. Proutiere, Y. Yi, T. Lan, and M. Chiang, "Resource Allocation over Network Dynamics without Timescale Separation," in Proc. *IEEE INFOCOM*, 2010.
[22] C. Joo, X. Lin, and N. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multihop wireless networks," IEEE/ACM Trans. Netw., vol. 17, no. 4, pp. 1132-1145, Aug. 2009.

[23] D. Qian, D. Zheng, J. Zhang, and N. Shroff, "CSMA-based distributed scheduling in multi-hop MIMO networks under SINR model," in Proc. *IEEE INFOCOM*, 2010.
[24] L. Le, E. Modiano, C. Joo, and N. Shroff, "Longest-queue-first scheduling under SINR interference model," in Proc. *ACM MobiHoc*, 2010.
[25] Li Ping Qian, Ying Jun Zhang, and Mung Chiang, "Globally Optimal Distributed Power Control for Nonconcave Utility Maximization," in Proc. *IEEE GlobeCom*, 2010.
[26] S. Merlin, N. Vaidya, and M. Zorzi, "Resource allocation in multi-radio multi-channel multi-hop wireless networks," in Proc. *IEEE INFOCOM*, 2008.
[27] Z. Haas and J. Deng, "Dual busy tone multiple access DBTMA: A multiple access control scheme for ad hoc networks," IEEE Trans. Commun., vol. 50, no. 6, pp. 975-985, 2002.
[28] G. Brar, D. Blough, and P. Santi, "The SCREAM approach for efficient distributed scheduling with physical interference in wireless mesh networks," in Proc. *IEEE ICDCS*, 2008.
[29] F. Kelly, *Reversibility and Stochastic Networks*, Wiley, Chichester, 1979.
[30] A. Eryilmaz, R. Srikant, and J. Perkins, "Stable scheduling policies for fading wireless channels," IEEE/ACM Trans. Netw., vol. 13, no. 2, pp. 411-424, Apr. 2005.
[31] S. Rajagopalan, D. Shah, and J. Shin, "Network adiabatic theorem: an efficient randomized protocol for contention resolution," in Proc. *ACM SIGMETRICS*, 2009.
[32] J. Ghaderi and R. Srikant, "On the Design of Efficient CSMA Algorithms for Wireless Networks," in Proc. *IEEE CDC*, 2010.

Jin-Ghoo Choi received the B.S., M.S., and Ph. D. degree in the school of Electrical Engineering & Computer Science, Seoul National University in 1998, 2000, and 2005, respectively. From 2006 to 2007, he worked for Samsung Electronics as a senior engineer. In 2009, he was with the Department of Electrical & Computer Engineering in The Ohio State University as a visiting scholar. He joined the Department of Information and Communication Engineering in Yeungnam University as a faculty member in 2010. His research interests span performance analysis and control of communication networks, resource management in wireless networks including packet scheduling, and wireless sensor networks.

Changhee Joo received his Ph.D degree from the school of EECS at Seoul National University in 2005. Before he joined Korea University of Technology and Education as a faculty member in 2010, he had been with the Center of Wireless Systems and Applications, Purdue University, and at the Ohio State University. He is now with the school of ECE at Ulsan National Institute of Science and Technology (UNIST), Korea. His research interests include multi-hop wireless networks, cross-layer optimization, wireless sensor networks, and resource allocation. He is an Associated Editor for Journal of Communications and Networks (JCN), was Social Media Chair for SECON 2012, and served as a TPC member for several primary conferences, including INFOCOM, MOBIHOC, GLOBECOM, PIMRC, and WCNC. He is a Senior Member of IEEE, and a recipient of the INFOCOM 2008 Best Paper Award.

Junshan Zhang received his Ph.D. degree from the School of ECE at Purdue University in 2000. He joined the School of ECEE at Arizona State University in August 2000, where he has been Professor since 2010. His interests include cyber-physical systems, communications networks, and network science. His current research focuses on fundamental problems in information networks and energy networks, including modeling and optimization for smart grid, network optimization/control, mobile social networks, crowdsourcing, cognitive radio, and network information theory.

Prof. Zhang is a fellow of the IEEE, and a recipient of the ONR Young Investigator Award in 2005 and the NSF CAREER award in 2003. He received the Outstanding Research Award from the IEEE Phoenix Section in 2003. He was TPC co-chair for a number of major conferences in communication networks, including INFOCOM 2012, WICON 2008 and IPCCC'06, and TPC vice chair for ICCCN'06, and served as a TPC member for INFOCOM, SECON, GLOBECOM, ICC, MOBIHOC, BROADNETS, and SPIE ITCOM. He was the general chair for IEEE Communication Theory Workshop 2007. He was an Associate Editor for IEEE Transactions on Wireless Communications, an editor for the Computer Network journal and an editor IEEE Wireless Communication Magazine. He co-authored a paper that won IEEE ICC 2008 best paper award, and one of his papers was selected as the INFOCOM 2009 Best Paper Award Runner-up. He is currently a Distinguished Lecturer of the IEEE Communications Society.

Ness B. Shroff received his Ph.D. degree in Electrical Engineering from Columbia University in 1994. He joined Purdue university immediately thereafter as an Assistant Professor in the school of Electrical and Computer Engineering. At Purdue, he became Full Professor of ECE in 2003 and director of CWSA in 2004, a university-wide center on wireless systems and applications. In July 2007, he joined The Ohio State University, where he holds the Ohio Eminent Scholar endowed chair in Networking and Communications, in the departments of ECE and CSE. From 2009-2012, he served as a Guest Chaired professor of Wireless Communications at Tsinghua University, Beijing, China, and currently holds an honorary Guest professor at Shanghai Jiaotong University in China. His research interests span the areas of communication, social, and cyberphysical networks. He is especially interested in fundamental problems in the design, control, performance, pricing, and security of these networks. Dr. Shroff is a past editor for IEEE/ACM Trans. on Networking and the IEEE Communication Letters. He currently serves on the editorial board of the Computer Networks Journal, IEEE Network Magazine, and the Networking Science journal. He has chaired various conferences and workshops, and co-organized workshops for the NSF to chart the future of communication networks. Dr. Shroff is a Fellow of the IEEE and an NSF CAREER awardee. He has received numerous best paper awards for his research, e.g., at IEEE INFOCOM 2008, IEEE INFOCOM 2006, Journal of Communication and Networking 2005, Computer Networks 2003 (two of his papers also received runner-up awards at IEEE INFOCOM 2005 and INFOCOM 2013), and also student best paper awards (from all papers whose first author is a student) at IEEE WiOPT 2013, IEEE WiOPT 2012 and IEEE IWQoS 2006.