Arbitrary Throughput Versus Complexity Tradeoffs in Wireless Networks using Graph Partitioning

Saikat Ray and Saswati Sarkar

Abstract

Several policies have recently been proposed for attaining the maximum throughput region, or a guaranteed fraction thereof, through dynamic link scheduling. Among these policies, the ones that attain the maximum throughput region require a computation time which is linear in the network size, and the ones that require constant or logarithmic computation time attain only certain fractions of the maximum throughput region. In contrast, in this paper we propose policies that can attain any desirable fraction of the maximum throughput region and require a computation time that is largely independent of the network size. First, using a combination of graph partitioning techniques and lyapunov arguments, we propose a simple policy for tree topologies under the primary interference model that requires each link to exchange only 1 bit information with its adjacent links and approximates the maximum throughput region using a computation time that depends only on the maximum degree of nodes and the approximation factor. We subsequently develop a framework for attaining arbitrary close approximations for the maximum throughput region in arbitrary networks and interference models and use this framework to obtain any desired tradeoff between throughput guarantees and computation times for a large class of networks and interference models. Specifically, given any $\epsilon > 0$, the maximum throughput region can be approximated in these networks within a factor of $1 - \epsilon$ using a computation time that depends only on the maximum node degree and ϵ.

I. Introduction

Attaining the maximum throughput region, or a guaranteed fraction thereof, through dynamic link scheduling, is a key design goal in multihop wireless networks. The scheduling problem involves determination of which links transmit packets at any given time. Appropriate scheduling of links is key towards attaining throughput guarantees as the success of transmission in any given link depends on which other links transmit packets simultaneously. The transmission schedules can not be pre-computed, and needs to be determined at every transmission epoch, as the congestion levels in the nodes and the transmission conditions in the wireless medium vary with time, and the statistics of these temporal variations are oftentimes not known a-priori. Thus, the time required to determine which links would transmit at any transmission epoch is a key performance metric for any dynamic scheduling policy. The contribution of this paper is to characterize tradeoffs between throughput guarantees and computation times for scheduling policies for different classes of wireless networks.

Owing to the lack of a central controller, at every transmission epoch each link needs to determine whether it would transmit based on its own state and the information it acquires about the states of other nodes. The throughput guarantees usually improve with increase in the information each link (or rather a node which is the source of the link) acquires about the states of other links. The time required for each link to decide whether to transmit at any given time depends on the time required (a) to exchange messages with other links to learn their states and (b) to perform the computations required to arrive at an appropriate decision based on the information acquired. We refer to the total time required in both parts as the schedule computation time, or rather the computation time. The time required in each part increases with increase in the amount of information a link acquires about the states of other links. Thus, an important question is how much information a link should acquire about the states of other links.

The scheduling policies that have been widely investigated can be classified in two broad classes based on the above qualifier: the policies that require each link to know some attribute that depends on the states

The authors are at the Department of Electrical and Systems Engineering at the University of Pennsylvania. Their emails are {saikat,swati}@seas.upenn.edu.

of (a) all links in the network [4], [16], [17] and (b) only the links that interfere with it [2], [10], [11], [13], [18]. We refer to the two classes as INFORMATION(N) and INFORMATION(1) policies respectively, where N is the number of links in the network. By this nomenclature, INFORMATION(k) is the class of policies that require each link to learn the states of their k-hop interferers. A seminal result has established that policies in INFORMATION(N) class can attain the maximum possible throughput region in arbitrary wireless networks using $O(N)$ computation time per scheduling decision [16]. Recently, it has been shown that a policy in INFORMATION(1) class can attain a guaranteed fraction of the maximum throughput region using $O(\Delta_G \log N)$ computation time per scheduling decision where Δ_G is the maximum degree, or the maximum number of neighbors of any given node, in the network [2]. The contribution of this paper is to show that in certain important classes of wireless networks, for appropriate selection of k between 1 and N, policies can be designed in INFORMATION(k) class so as to obtain arbitrary close approximations for the maximum throughput region, while requiring a computation time that depends only on Δ_G and the desired approximation factor and is otherwise independent of the size of the network.

We first consider the primary interference model which mandates that any set of links can be simultaneously scheduled provided they do not have any common node. Under this interference model, when the network topology is a tree, given any positive constant ϵ, we obtain a distributed scheduling policy in INFORMATION(1) class that (a) approximates the throughput region within a factor of $1-\epsilon$ and (b) requires a computation time of $O(\Delta_G/\epsilon)$ (Section IV). The policy is simple to implement as the computation of each schedule needs each link to communicate only 1 bit of control message to each of the links that interferes with it.

We next present a general framework for designing INFORMATION(k) policies for approximating the throughput region arbitrarily closely for arbitrary networks and interference models (Section V). We subsequently use this framework for obtaining arbitrary tradeoffs between throughput guarantees and computation times for large classes of networks, e.g., graphs with limited cyclicity and primary interference models (Section V-B), geometric and quasi-geometric graphs under both primary and secondary interference models (Section V-C,V-D). For example, when nodes are embedded in a plane and there exists an edge between two nodes if and only if the distance between them is less than a given constant D (i.e., for geometric graphs), given any positive constant ϵ, we obtain a distributed scheduling policy in INFORMATION($O(\Delta_G/\epsilon^2)$) class that (a) approximates the throughput region within a factor of $1-\epsilon$ and (b) requires a computation time of $O(\Delta_G^2/\epsilon^2)$. The throughput and computation time guarantees extend to networks where sessions traverse multiple links (Section VI).

Under the primary interference constraints in tree topologies existing policies attain (a) the maximum throughput region using a computation time of $\Theta(N)$ [4], [16] (b) $2/3$ of the maximum throughput region using a computation time of $\Theta(\Delta_G (\log N)^2)$ [13] and (c) $1/3$ of the maximum throughput region using a computation time of $\Theta(\Delta_G)$ [10]. For geometric graphs and secondary interference model, existing policies attain (a) the maximum throughput region using a computation time of $\Theta(N)$ [16] (b) $1/8$ of the maximum throughput region using a computation time of $\Theta(\Delta_G \log N)$ [2] and (c) $1/\Delta_G$ of the maximum throughput region using a computation time of $\Theta(\Delta_G)$ [10]. Our policies therefore attain arbitrary desired tradeoffs between the best known guarantees for throughput and computation times. Specifically, for networks with bounded degree, our policies approximate the throughput region within any desired approximation factor using a computation time which depends only on the approximation factor and does not depend on the network size, whereas existing algorithms that require constant computation time attain an approximation guarantee that (a) is $1/2$ in the first case and (b) depends on the degree bound in the second case. For networks with degrees $O(\mathrm{logN})$ (which happens in several topologies), our policies approximate the throughput region within any constant factor using poly-logarithmic computation time, whereas existing algorithms that use poly-logarithmic computation time attain an approximation factor of at most $2/3$ and $1/8$ for the above cases respectively.

We now briefly describe the design of the proposed policies, and provide the intuition behind the performance guarantees. The proposed policies partition the network in a collection of components - the size of the components depend only on Δ_G and ϵ. The links that originate in a component but interfere with those in another component are "shut down" i.e., not scheduled. Thus, the links scheduled in each component will not interfere with those scheduled in other components irrespective of the scheduling policy in each component. Hence, the scheduling in different components can now be determined in parallel. Thus, the time required to compute the overall schedule now depends only on the size of each component and can be determined only by Δ_G and ϵ for appropriate partitioning schemes. The links are scheduled in each component so as to maximize the throughput region of the component. Thus, the reduction in the overall throughput region may only happen because links are occasionally shut down. But, the resulting reduction in throughput is small because each link is shut down only a small fraction of time since different partitioning schemes are used at different times and the size of the components in each partition is large enough. The proofs for the throughput guarantees rely on a combination of graph-partitioning techniques and lyapunov arguments.

II. Related Literature

The problem of maximizing the throughput region in wireless networks, or attaining a guaranteed fraction thereof, has received significant attention. Tassiulas *et al.* have characterized the maximum throughput region and provided a policy that attains this throughput region in an arbitrary wireless network [17]. This policy schedules the maximum weighted independent set of links in each slot, and hence requires $\Omega(e^N)$ computation time unless $P = NP$. Later, Tassiulas [16] provided randomized scheduling schemes that attain the maximum achievable throughput region, which can be implemented in fully distributed manner using gossip based algorithms [5]. In each slot, this policy randomly selects an independent set of links, compares its weight with the weight of the set of links scheduled in the previous slot and schedules the set that has the larger weight. This policy requires $\Theta(N)$ computation time. Recently, Dimakis *et al.* [4] have shown that a greedy maximal weight scheduling, which requires $\Theta(N)$ computation time, attains the maximum throughput region in several different networks. All these policies are in the Information(N) class.

Recently, provable throughput guarantees have been obtained with some policies in Information(1) class. Dai *et al.* [3], Lin *et al* [11] and Wu *et al.* [18] proved that a simple greedy scheduling scheme, maximal matching, attains half the maximum throughput region for the primary interference model; the computation time for maximal matching is $\Theta(\Delta_G \log N)$. Chaporkar *et al.* [2] proved that maximal matching can be generalized to attain guaranteed fraction of the maximum throughput region for arbitrary interference models, while retaining the logarithmic computation time. Sarkar *et al.* [13] proved that for primary interference model and tree graphs, a queue length dependent maximal matching attains $2/3$ of the throughput region while using $\Theta\left(\Delta_G(\log(N))^2\right)$ computation time. Lin *et al.* [10] proved that a random access scheme, where links access the medium with a probability that depends on their and their interferers' queue lengths, attains $1/3$ and $1/\Delta_G$ the throughput region for arbitrary networks under primary and secondary interference models respectively, while requiring a $O(\Delta_G)$ computation time.

Our contribution is to introduce the class of Information(k) policies and prove that for appropriate choices of k, policies can be designed in the Information(k) class so as to obtain arbitrary tradeoffs between the best throughput guarantees and the computation times obtained so far.

The design of our policies rely on the use of graph partitioning techniques. Hunt *et al.* [8], Kuhn *et al.* [9], Nieberg *et al* [12] and Sharma *et al.* [14] have devised graph partitioning techniques for obtaining arbitrary close approximations of maximum weighted independent sets in polynomial growth bounded graphs. A graph is said to be polynomial growth bounded if the maximum number of pairwise independent nodes in any r-neighborhood of a node can be upper-bounded by a polynomial in r. Many of the graphs we consider, e.g., trees, are not polynomial growth bounded. Even in the polynomial growth bounded graphs we consider, i.e., geometric graphs, existing results [8], [12], [14] approximate maximum weighted independent sets within a factor of $1-\epsilon$ using policies in Information (N) class which have computation times of $\left(N + \Delta_G^{\Theta(1/\epsilon^2)}\right)$. Thus selecting the links using these approximation techniques require central control and $\Theta(N + \Delta_G^{\Theta(1/\epsilon^2)})$ time for computing each schedule. For geometric graphs, our framework yields a policy in the INFORMATION($O(\Delta_G/\epsilon^2)$) class that computes each schedule in $O(\Delta_G^2/\epsilon^2)$ time using a simpler partitioning technique, and still attains desired approximation guarantees for the maximum throughput region. The partitioning technique used in [9] however requires $\Delta_G^{\Theta(1/\epsilon^2)}$ time for computing a maximum size independent set which does not depend on N as well, but this technique approximates a maximum weighted independent set arbitrarily closely only when the weights are all equal. Since different links have different queue lengths in a network, this partitioning technique does not provide throughput guarantees. Finally, Brzezinski *et al.* have recently used graph partitioning techniques for providing throughput guarantees using $\Theta(N)$ scheduling schemes for networks with multiple channels [1]. Their goal is to divide the graph in subgraphs such that different subgraphs are assigned different channels, and a greedy maximal weight scheduling, which requires $\Theta(N)$ computation time, maximizes the throughput region in each subgraph. Driven by different goals, we use different partitioning schemes.

III. SYSTEM MODEL

We consider scheduling at the MAC layer in a wireless network. We assume that time is slotted. The topology in a wireless network can be modeled as a graph $G = (V, E)$, where V and E respectively denote the sets of nodes and links. A link exists from a node u to another node v if and only if both u and v can receive each others' signals. Let $|E| = N$. Each session represents a triplet (i, u, v) where i is the identifier associated with the session and u and v are source and destinations of the session. At the MAC layer, each session traverses only one link, but multiple sessions may traverse a link. We consider a network with $\hat{N}$ sessions.

We now introduce terminologies that we use throughout the paper. Some of these are well-known in graph theory; we mention these for completeness.

A node i is a *neighbor* of a node j, if there exists a link from i to j, i.e., $(i, j) \in E$. The *degree of a node* u is the number of neighbors of u. We denote the maximum degree of any node in G as Δ_G. Two links (sessions) are *adjacent* to each other if they have common nodes. By definition, a link is adjacent to itself.

A link i *interferes* with link j if j can not successfully transmit a packet when i is transmitting. A subset of links is said to be *independent* if if no link in the subset interferes with another link in the subset. Let $\mathcal{X}$ be the collection of independent sets of links.

We now describe the packet arrival process. We assume that at most $\hat{\alpha}_{\max} \geq 1$ packets arrive for any session in any slot. Let $\hat{A}_i(t)$ be the number of packets that session i generates in slot t. We assume that a packet arriving in a slot arrives at the end of the slot, and may not be transmitted in the slot. The arrival process $\{\hat{A}_i(t)\}$ is independent and identically distributed for all t.

A *scheduling policy* is an algorithm that decides in each slot the subset of sessions that would transmit packets in the slot. Clearly, a subset of sessions can transmit packets in any slot if no two sessions in the subset traverse the same link and the links the sessions traverse constitute an independent set X, i.e., if $X \in \mathcal{X}$. Every packet has length 1 slot. Thus, if a session is scheduled in a slot, it transmits a packet in the slot.

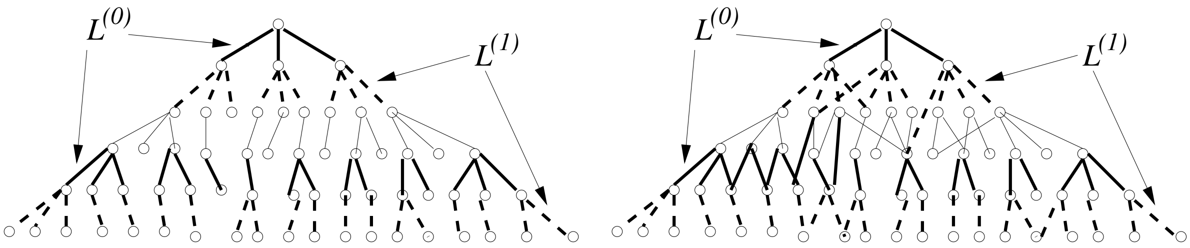
Let $\hat{D}_i(t)$ be the number of packets that session i transmits in slot t, $i = 1, \ldots, \ldots, \hat{N}$. Now, $\hat{D}_i(t) \in \{0, 1\}$ and depends on the scheduling policy.

Let $\hat{Q}_i(t)$ be the queue length before the arrivals and the transmissions in slot t. Then $\hat{Q}_i(t+1) = \hat{Q}_i(t) + \hat{A}_i(t) - \hat{D}_i(t)$.

Definition 1: The network is said to be *stable* if there exists a finite real number Γ such that with probability 1,

$$\limsup_{T\to\infty} \sum_{t=0}^{T-1} \hat{Q}_i(t)/T \leq \Gamma, \ \ i = 1, \ldots, \hat{N}. \tag{1}$$

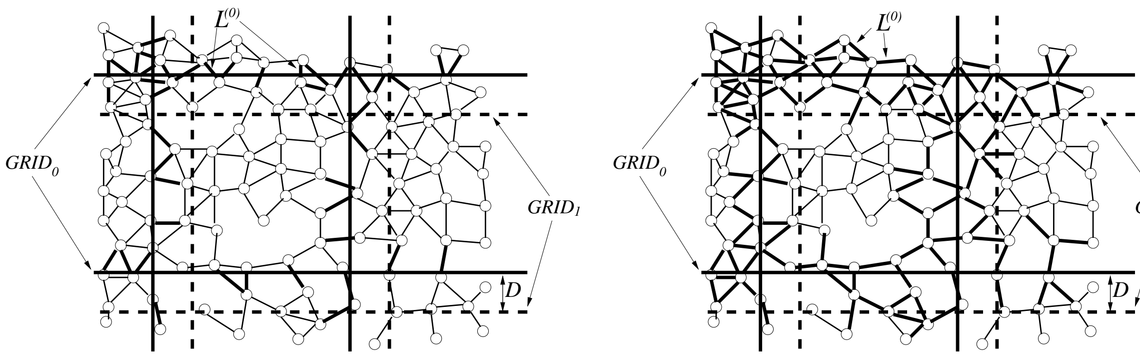
We consider a virtual-queue Q_l associated with link l that contains all packets waiting for transmission for all sessions that traverse l. Note that the virtual queue in a link l may contain packets of sessions



(a) Tree topologies

(b) Topologies with limited cyclicity

Fig. 1. The figures demonstrate the edge sets $L^{(0)}, L^{(1)}$ under the primary interference model for (a) a tree and (b) topology with limited cyclicity.



(a) Geometric graphs under primary interference

(b) Geometric graphs under secondary interference

Fig. 2. The figures demonstrate the edge sets $L^{(1)}, L^{(2)}$ for a geometric graph under (a) primary and (b) secondary interference models.

traversing l in both directions. Let $A_l(t)$ and $D_l(t)$ respectively denote the number of arrivals and departures in slot t in virtual queue Q_l. Clearly, the arrival process $\{A_l(t)\}$ is independent and identically distributed for all t and for all $l, t,\ A_l(t) \leq \alpha_{\max}$ where $\alpha_{\max} = \hat{N}\hat{\alpha}_{\max}$. Let $\mathbf{E}A_l(t) = \lambda_l$. The *arrival rate* of link i is $\lambda_i,\ i = 1, \ldots, |E|$. The *arrival rate vector* $\vec{\lambda}$ is an $|E|-$dimensional vector whose components are the arrival rates.

Now, $Q_l(t+1) = Q_l(t) + A_l(t) - D_l(t)$. Also, (1) holds if and only if $\limsup_{T\rightarrow\infty} \sum_{t=0}^{T-1} Q_i(t)/T$ is finite.

The *throughput region* Λ^π of a scheduling policy π is the set of arrival rate vectors $\vec{\lambda}$ for which the network is stable under π. An arrival rate vector $\vec{\lambda}$ is said to be *feasible* if it is in the throughput region of some scheduling policy. The *maximum throughput region* Λ is the set of feasible arrival rate vectors. A scheduling policy π is said to approximate the maximum throughput region within a factor $1-\epsilon$ if for each arrival rate vector $\vec{\lambda} \in \Lambda,\ (1-\epsilon)\vec{\lambda} \in \Lambda^\pi$.

IV. INFORMATION(1) POLICY FOR APPROXIMATING THE MAXIMUM THROUGHPUT REGION ARBITRARILY CLOSELY IN TREE TOPOLOGIES

We assume that G is a tree and consider the primary interference model. Under this interference model, two links interfere if and only if they have a common end-point. A matching is a set of links such that no two links in the set are adjacent to each other. Thus, a valid schedule in a slot is a matching in the basic graph G, and $\mathcal{X}$ is the set of all matchings in G. This interference model is encountered in networks like Bluetooth where each node has a single transceiver and a unique frequency in its neighborhood.

We now describe the scheduling policy which we refer to as TREE-PARTITION-MATCHING(k), and abbreviate as TPM(k). Here, k is a parameter which determines the throughput region and the computation time of the policy.

We first introduce the following notations. The level of a node $\text{level}(u)$ in a tree is its distance from the root of the tree. A link $l = (u, v)$ is the parent (child) of a link $l' = (v, w)$ if $\text{level}(u) < \text{level}(v) < \text{level}(w)$ ($\text{level}(w) < \text{level}(v) < \text{level}(u)$). Links $(u, v_1), (u, v_2), \ldots$ are siblings of each other if $\text{level}(v_1) = \text{level}(v_2)$. Also, different priorities are associated with different siblings such that between any two siblings one is older and the other is younger. Let $J_l = \{l' \in E : l' \text{ is a parent or older sibling of } l\}$. For $j = 0, \ldots, k-1$, let $L^{(j)}$ be the set of links (u, v) such that levels of u and v are j and $j+1$ modulo k (Figure 1(a)).

A formal description of TPM(k) follows.

TREE-PARTITION-MATCHING (k)

In slot t, every link selects an integer in the range $[0, \ldots k-1]$; each integer is selected with probability $1/k$ and all links select the same integer. Let $i(t)$ be the integer selected in slot t. A link l contends if and only if $l \in E \setminus L^{(i(t))}$.

A link schedules itself if and only if (a) it contends (b) its virtual-queue has packets to transmit, and (c) links in J_l do not schedule themselves.

When a link is scheduled, the head of line packet in the corresponding virtual queue is served.

Note that TPM (k) belongs in the INFORMATION (1) class irrespective of the value of k, and is simple to implement since each link only needs to inform its children and younger siblings only about whether it has scheduled itself. We now evaluate the computation time for TPM(k). Note that in any slot the links that contend constitute a forest such that those in a tree of the forest do not interfere with those in a different tree of the forest. Thus, the scheduling in different components can be determined in parallel. The maximum length of a path in any tree in the forest is k. Each link that contends decides whether to schedule itself immediately after it knows the decisions of its parents and older siblings that contend. Thus, each link waits for the scheduling decision of at most $k\Delta_G$ links. Thus, the overall computation time is $O(k\Delta_G)$.

Theorem 1: If $\vec{\lambda} \in \text{Int}(\Lambda)$, then $(1 - 1/k)\vec{\lambda} \in \Lambda^{\text{TPM}(k)}$.

We first outline the intuition behind Theorem 1. First, intuitively a scheduling policy π that schedules a link l if and only if (a) it has a packet to transmit and (b) links in J_l do not schedule themselves, maximizes the throughput region in a tree. This is because whenever a link l has a packet to transmit, π schedules either l or a link in J_l; the optimum policy also schedules at most one link in $J_l \cup \{l\}$ in each slot. Clearly, the computation time for π is $O(d\Delta_G)$ where d is the depth of the tree, and d is $O(|E|)$. Now, by preventing the contention of a subset $L^{(i(t))}$ of links in each slot t, TPM (k) partitions the graph in a forest where the depth of each tree is at most k, and uses the above scheduling policy in each tree of the forest. This reduces the computation time of TPM (k) to $O(k\Delta_G)$. The choice of $L^{(0)}, \ldots, L^{(k-1)}$, and different selections of $i(t) \in \{0, \ldots, k-1\}$ in each slot t ensures that a link contends with probability $1 - 1/k$ in each slot t; this in turn ensures that the maximum throughput region reduces only by a factor of $1 - 1/k$.

Proof: The result clearly holds if $k = 1$. Thus, we assume that $k > 1$. The arrival rate vector is $(1 - 1/k)\vec{\lambda}$ where $\vec{\lambda} \in \text{Int}(\Lambda)$. Since $\vec{\lambda} \in \text{Int}(\Lambda)$ and X constitutes of all matchings of the links, $\sum_{l' \in J_l \cup \{l\}} \lambda'_l < 1$ [7], [17]. Let $\delta = \min\left(\frac{1 - \max \sum_{l' \in J_l \cup \{l\}} \lambda_{l'}}{2|E| \max_{l'} \lambda_{l'}}, 1\right)$. Clearly, $\delta > 0$. For a link $l = (u, v)$, χ_l denotes the sum of $\min(\text{level}(u), \text{level}(v))$ and the number of older siblings of l.

Observe that the queue lengths of the virtual queues constitute a Markov chain. We consider a lyapunov

function
$$V(\vec{Q}) = \sum_l \delta^{\chi_l} Q_l^2 + 2\sum_l \delta^{\chi_l} Q_l \sum_{l' \in J_l} Q_{l'}.$$
Note that the use of δ^{χ_l}s in the lyapunov function have been motivated by the a-symmetricity of J_ls (J_l is asymmetric in the sense that if l' is in J_l then l is not in J_l'). We prove that $\mathbf{E}\left(V\left(\vec{Q}\,(t+1)\right) - V\left(\vec{Q}(t)\right)|\vec{Q}(t) = \vec{Q}\right) < -1$ for all sufficiently large $||\vec{Q}||$, where $||\vec{Q}|| = \sqrt{V(\vec{Q})}$. Then, from Foster's theorem (Theorem 2.2.3 in [6]) the Markov chain representing the queue length process $\vec{Q}_l(t)$ is positive recurrent. Also, $\mathbf{E}\left(Q_l(t)\right) < \infty$ for each l under the steady state distribution for the above Markov chain. Thus, $\lim_{K\to\infty} \frac{\sum_{t=0}^{K-1} Q_l(t)}{K} < \infty$. The result follows.

$$\begin{aligned}
& V\left(\vec{Q}(t+1)\right) - V\left(\vec{Q}(t)\right) \\
= \; & \sum_l \delta^{\chi_l}\left(Q_l(t+1) - Q_l(t)\right)\left(Q_l(t+1) + Q_l(t)\right) + 2\sum_l \delta^{\chi_l} Q_l(t+1) \sum_{l' \in J_l} Q_{l'}(t+1) \\
& -2\sum_l \delta^{\chi_l} Q_l(t) \sum_{l' \in J_l} Q_{l'}(t) \\
\leq \; & 2\sum_l \delta^{\chi_l}\left(A_l(t) - D_l(t)\right) Q_l(t) + 2\sum_l \delta^{\chi_l}\left(A_l(t) - D_l(t)\right)^2 + 2\sum_l \delta^{\chi_l} Q_l(t) \sum_{l' \in J_l}\left(A_{l'}(t) - D_{l'}(t)\right) \\
& +2\sum_l \delta^{\chi_l}\left(A_l(t) - D_l(t)\right)\sum_{l' \in J_l} Q_{l'}(t) + 2\sum_l \delta^{\chi_l}\left(A_l(t) - D_l(t)\right)\sum_{l' \in J_l}\left(A_{l'}(t) - D_{l'}(t)\right) \\
\leq \; & 2\sum_l \delta^{\chi_l} Q_l(t)\left(\sum_{l' \in J_l \cup \{l\}}\left(A_{l'}(t) - D_{l'}(t)\right) + \sum_{l' : l \in J_{l'}} \delta^{\chi_{l'} - \chi_l} A_{l'}(t)\right) + 4N^2\alpha_{\max}^2 \\
\leq \; & 2\sum_l \delta^{\chi_l} Q_l(t)\left(\sum_{l' \in J_l \cup \{l\}}\left(A_{l'}(t) - D_{l'}(t)\right) + \delta\sum_{l' : l \in J_{l'}} A_{l'}(t)\right) + 4N^2\alpha_{\max}^2. \qquad (2)
\end{aligned}$$

The last inequality follows since $0 < \delta \leq 1$, $\chi_l < \chi_{l'}$ if $l \in J_{l'}$. From (2),

$$\begin{aligned}
& \mathbf{E}\left(V\left(\vec{Q}\,(t+1)\right) - V\left(\vec{Q}(t)\right)|\vec{Q}(t) = \vec{Q}\right) \\
\leq \; & (2/k)\sum_l \delta^{\chi_l}\sum_{m=0}^{k-1}\mathbf{E}\left(Q_l(t)\left(\sum_{l' \in J_l \cup \{l\}}\left(A_{l'}(t) - D_{l'}(t)\right) + \delta\sum_{l' : l \in J_{l'}} A_{l'}(t)\right)|\vec{Q}(t) = \vec{Q}, i(t) = m\right) \\
& +4N^2\alpha_{\max}^2 \\
\leq \; & (2/k)\sum_l \delta^{\chi_l} Q_l\left(k(1-1/k)\sum_{l' \in J_l \cup \{l\}}\lambda_{l'} - (k-1) + k(1-1/k)\delta\sum_{l' : l \in J_{l'}}\lambda_{l'}\right) + 4N^2\alpha_{\max}^2 \\
& \text{(since } l \in L^{(j)} \text{ for only one } j \in \{0, \ldots, k-1\} \\
& \text{and } D_{l'}(t) = 1 \text{ for some } l' \in J_l \cup \{l\} \text{ unless } Q_l(t) = 0 \text{ or } l \in L^{(i(t))}) \\
\leq \; & 2(1-1/k)\sum_l \delta^{\chi_l} Q_l\left(\sum_{l' \in J_l \cup \{l\}}\lambda_{l'} - 1 + \delta\sum_{l' : l \in J_{l'}}\lambda_{l'}\right) + 4N^2\alpha_{\max}^2 \\
\leq \; & -2(1-1/k)|E|\max_l \lambda_{l'}\delta\sum_l \delta^{\chi_l} Q_l \\
< \; & -1 \text{ for sufficiently large } ||\vec{Q}|| \text{ (since } \delta > 0 \text{ and } k > 1).
\end{aligned}$$

The result follows. ■

Thus, TPM ($\lceil 1/\epsilon \rceil$) approximates the maximum throughput region within a factor of $1-\epsilon$. The computation time for TPM ($\lceil 1/\epsilon \rceil$) is $O(\Delta_G/\epsilon)$.

V. INFORMATION(k) POLICIES FOR APPROXIMATING THE MAXIMUM THROUGHPUT REGION ARBITRARILY CLOSELY FOR ARBITRARY NETWORKS AND INTERFERENCE MODELS

We first provide a general framework for approximating the maximum throughput region arbitrarily closely in arbitrary networks and interference models using policies in INFORMATION(k) class (Section V-A). We use this framework to obtain arbitrary tradeoffs between throughput approximations and computation times in several important classes of networks and interference models (Section V-B, V-C, V-D). Specifically, we prove that in a geometric graph for both primary and secondary interference models the maximum throughput region can be approximated within a factor of $1-\epsilon$ using a policy in INFORMATION(Δ_G/ϵ^2) class and a computation time of $O\left(\Delta_G^2/\epsilon^2\right)$ (Section V-C). These results can be extended to arbitrary graphs with limited cyclicity (Section V-B) and quasi-geometric graphs (Section V-D). In Section V-E, we discuss how these policies can be implemented.

A. General Framework

We describe a policy $\pi(k)$ that approximates the maximum throughput region arbitrarily closely for appropriate choices of k in arbitrary networks and interference models (the network and interference models are as described in Section III). We consider k subsets of links $L^{(0)}, \ldots, L^{(k-1)}$ such that the links in a component of $G^{(j)} = (V, E \setminus L^{(j)})$ do not interfere with those in other components of $G^{(j)}$. In every slot t, every link selects an integer in the range $[0, \ldots k-1]$; each integer is selected with probability $1/k$ and all links select the same integer. In any slot t, the weight of a link is the number of packets waiting for transmission in the virtual queue associated with the link, and the links that constitute a maximum weighted independent set in the interference graph of any component of $G^{(i(t))}$ are scheduled. Without loss of generality, links with zero weight are not scheduled. When a link l is scheduled, the virtual queue associated with l transmits a packet.

Note that $\pi(k)$ is completely specified once $L^{(0)}, \ldots, L^{(k-1)}$ are specified. We show that for appropriate choices of $L^{(0)}, \ldots, L^{(k-1)}$, $\pi(k)$ approximates the maximum throughput region within an approximation factor that depends only on k. We first introduce the following definitions.

$$\text{Let } S_l = \{l' : l' = l \text{ or } l' \text{ interferes with } l \text{ or } l \text{ interferes with } l'\}.$$

Definition 2: A collection of subsets $E_1, \ldots, E_q$ of E is said to be *c-approximate* if for (a) any given $|E|$-dimensional vector of non-negative real numbers $\vec{W} = (W_1, \ldots, W_{|E|})$ and (b) any collection of subsets of E, $X_1, \ldots X_q$ such that $X_i \in \mathcal{X}$ and $X_i \subseteq E_i$

$$\sum_{i=1}^{q} \sum_{l \in X_i} W_l \leq c \max_{X \in \mathcal{X}} \sum_{l \in X} W_l.$$

We now present the key technical lemma that allows us to obtain desired throughput guarantees.

Lemma 1: Let $L^{(0)}, \ldots, L^{(k-1)}$ be c-approximate. Then,

$$\mathbf{E}\left(\sum_i Q_i(t) D_i(t) | \vec{Q}(t) = \vec{Q}\right) \geq (1 - c/k) \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t).$$

We first provide the intuition behind the result. Now, the weight of the links scheduled by $\pi(k)$ differs from the maximum weight of any schedule in the slot by at most the weight of the maximum weight independent set among links that do not contend in the slot. Now, if $L^{(0)}, \ldots, L^{(k-1)}$ are $c-$approximate, the expected weight of the maximum weight independent set in $L^{(j)}$ for $j = 0 \ldots k-1$ turns out to be at most c/k times that of the weight of the maximum weight independent set in the slot. Thus, the expected weight of the scheduled links is at least $(1 - c/k)$ times that of the weight of the maximum weight of any schedule in the slot.

Proof: Let $i(t)$ be the integer selected by links in slot t, and

$$B(t) = \arg \max_{\substack{X \in \mathcal{X} \\ X \subseteq L^{(i(t))}}} \sum_{l \in X} Q_l(t).$$

Now, $\sum_i Q_i(t) D_i(t) \geq \left(\max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t) - \sum_{i \in B(t)} Q_i(t) \right)$. Now,

$$\begin{aligned}
\mathbb{E}\left(\sum_{l \in B(t)} Q_l(t) | \vec{Q}(t) = \vec{Q}\right) &= \sum_{j=0}^{k-1} \mathbf{P}\left(i(t) = j | \vec{Q}(t) = \vec{Q}\right) \mathbb{E}\left(\sum_{l \in B(t)} Q_l(t) | \vec{Q}(t) = \vec{Q}, i(t) = j\right) \\
&= (1/k) \sum_{j=0}^{k-1} \mathbb{E}\left(\sum_{l \in B(t)} Q_l(t) | \vec{Q}(t) = \vec{Q}, i(t) = j\right) \\
&= (1/k) \sum_{j=0}^{k-1} \max_{\substack{X \in \mathcal{X} \\ X \subseteq L^{(j)}}} \sum_{l \in X} Q_l(t) \\
&\leq (c/k) \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t) \text{ (since } L^{(0)}, \ldots, L^{(k-1)} \text{ are } c - \text{approximate).}
\end{aligned}$$

Thus, $\mathbb{E}\left(\sum_i Q_i(t) D_i(t) | \vec{Q}(t) = \vec{Q}\right) \geq (1 - c/k) \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)$. ■

Lemma 2: Let $L^{(0)}, \ldots, L^{(k-1)}$ be c-approximate. Then, if $\vec{\lambda} \in \text{Int}(\Lambda)$ and $k \geq c > 0$, $(1 - c/k)\vec{\lambda} \in \Lambda^{\pi}$. We first provide the intuition behind the above result. When $L^{(0)}, \ldots, L^{(k-1)}$ are c-approximate, from lemma 1 it follows that $\pi(k)$ schedules links such that the expected weight of the scheduled links in any slot is at least $(1 - c/k)$ times that of the maximum weight independent set of links in the slot. The throughput guarantee now follows using lyapunov arguments similar to those in [11], [17]. We prove this lemma at the end of this subsection.

Once we prove that the collection $L^{(0)}, \ldots, L^{(k-1)}$ is c-approximate, Lemma 2 allows us to approximate the maximum throughput region within a factor of $1 - \epsilon$ for any $\epsilon > 0$ using $\pi(k)$ for $k = \lceil c/\epsilon \rceil$. In the next subsections we will prove that in large classes of networks the collection $L^{(0)}, \ldots, L^{(k-1)}$ can be selected so as to render it c-approximate for different constant factors c.

Note that different components in each $G^{(j)}$ can schedule the links in parallel as the links in different components do not interfere. Thus, $\pi(k)$ can be implemented provided in each slot and in each component either one, or all links, know the weights of all links in the component. In either case, $\pi(k)$ is in INFORMATION($\hat{k}$) class where $\hat{k}$ is the maximum diameter of any component of $G^{(j)}$ for any $j \in \{0, \ldots, k-1\}$* which is upper bounded by the number of nodes in any component of $G^{(j)}$ for any $j \in \{0, \ldots, k-1\}$. The computation time for $\pi(k)$ will again be determined by the maximum size (number of links or number of nodes or both) of a component in $G^{(j)}$ for $j \in \{0, \ldots, k-1\}$. We will show that for a large class of networks, the size of each component and therefore the overall computation time depends only on Δ_G and k.

We now prove lemma 2.

Proof: The result clearly holds when $k = c$. We now assume that $k > c > 0$. Let the arrival rate vector be $(1 - c/k)\vec{\lambda}$ where $\vec{\lambda} \in \text{Int}(\Lambda)$. Clearly, under $\pi(k)$, $\vec{Q}(t)$ constitute an aperiodic irreducible Markov chain. We will consider the lyapunov function $V(\vec{Q}) = \sum_i Q_i^2$, and prove that under $\pi(k)$, $\mathbf{E}\left(V\left(\vec{Q}(t+1)\right) - V\left(\vec{Q}(t)\right) | \vec{Q}(t) = \vec{Q}\right) < -1$ for all sufficiently large $||\vec{Q}||$, where $||\vec{Q}|| = \sqrt{V(\vec{Q})}$.

*The tacit assumption we make here is that two adjacent links always interfere with each other which usually holds in all wireless networks. Note that we allow links to interfere even if they are not adjacent.

Then, from Foster's theorem (Theorem 2.2.3 in [6]) the Markov chain representing the queue length process is positive recurrent. Also, $\mathbf{E}\left(Q_i(t)\right) < \infty$ for each i under the steady state distribution for the above Markov chain. Thus, $\lim_{K\to\infty} \frac{\sum_{t=0}^{K-1} Q_i(t)}{K} < \infty$. The result follows.

Let $\vec{I}^X$ denote the indicator vector for set $X \in \mathcal{X}$. Note that $\phi \in \mathcal{X}$. Then, $\mathrm{Int}(\Lambda)$ can be characterized as follows [17]:

$$\mathrm{Int}(\Lambda) = \{\vec{\lambda} : \vec{\lambda} = \sum_{\vec{X}\in\mathcal{X}} \beta_X \vec{I}^X, \text{ where } \sum_{\vec{X}\in\mathcal{X}} \beta_X = 1 \text{ and } \beta_X \geq 0 \text{ for each } X \in \mathcal{X} \text{ and } \beta_\phi > 0\}. \tag{3}$$

Now,

$$\begin{aligned}
\mathbf{E}\left(\left(\vec{A}(t)\right)^T \vec{Q}(t) | \vec{Q}(t) = \vec{Q}\right) &= (1 - c/k)\vec{\lambda}^T \vec{Q} \\
&= (1 - c/k) \sum_{X\in\mathcal{X}} \beta_X \left(\left(\vec{I}^X\right)^T \vec{Q}\right) \text{ where } \beta_\phi > 0 \text{ (from (3))} \\
&= (1 - c/k) \sum_{X\in\mathcal{X}\setminus\{\phi\}} \beta_X \sum_{i\in X} Q_i \\
&\leq (1 - c/k) \left(\sum_{X\in\mathcal{X}\setminus\{\phi\}} \beta_X\right) \max_{X\in\mathcal{X}} \sum_{i\in X} Q_i \\
&= (1 - c/k)(1 - \beta_\phi) \max_{X\in\mathcal{X}} \sum_{i\in X} Q_i. \qquad (4)
\end{aligned}$$

$$\begin{aligned}
&\mathbf{E}\left(V\left(\vec{Q}(t+1)\right) - V\left(\vec{Q}(t)\right) | \vec{Q}(t) = \vec{Q}\right) \\
= \; &\mathbf{E}\left(\left(\vec{Q}(t+1) - \vec{Q}(t)\right)^T \left(\vec{Q}(t+1) + \vec{Q}(t)\right) | \vec{Q}(t) = \vec{Q}\right) \\
= \; &\mathbf{E}\left(\left(\vec{A}(t) - \vec{D}(t)\right)^T \left(2\vec{Q}(t) + \vec{A}(t) - \vec{D}(t)\right) | \vec{Q}(t) = \vec{Q}\right) \\
\leq \; &2\mathbf{E}\left(\left(\vec{A}(t) - \vec{D}(t)\right)^T \vec{Q}(t) | \vec{Q}(t) = \vec{Q}\right) + N\alpha_{\max}^2 \\
\leq \; &-2(1 - c/k)\beta_\phi \max_{X\in\mathcal{X}} \sum_{i\in X} Q_i + N\alpha_{\max}^2 \text{ (from Lemma 1 and (4))} \\
< \; &-1 \text{ for all sufficiently large } ||\vec{Q}|| \text{ (since } \beta_\phi > 0, 0 < c < k).
\end{aligned}$$

■

B. Graphs with Limited Cyclicity

Using the above general framework, we generalize the tradeoffs between throughput and computation times to networks with limited cyclicity. Specifically, we assume that there exists a constant H such that the maximum length of a cycle in G is upper bounded by H. We still consider the primary interference model.

We now describe $L^{(0)}, \ldots, L^{(k-1)}$ for the scheduling policy which we refer to as H-LIMITED-CYCLICITY-PARTITION-MATCHING(k), and abbreviate as H-LCPM(k). Consider a spanning tree T of G. For H-LCPM(k), $L^{(j)}$ is the set of links (u, v) such that the levels of u and v in T are (a) less than or equal to jH modulo kH and (b) greater than jH modulo kH respectively (Figure 1(b)). Intuitively, for H-LIMITED-CYCLICITY-PARTITION-MATCHING(k), when $i(t) = j$, levels jH, $jH + kH$, $jH + 2kH, \ldots$

partition the graph, and $L^{(j)}$ consists of the links that cross these levels. Clearly, the components of $G^{(j)}$ are such that the links in a component do not interfere with those in other components.

We now evaluate the computation time for H-LCPM(k). Let the set of edges in T be $\hat{E}$. Note that the maximum length of a path in $T^{(j)} = (V, \hat{E} \setminus L^{(j)})$ is kH. Thus each component in $T^{(j)}$ has $O(\Delta_G^{kH})$ nodes. Each component of $G^{(j)}$ consists of several components of $T^{(j)}$. Let u and v be nodes that are in different components of $T^{(j)}$ but are in the same component of $G^{(j)}$. Then the sum of the distances of u and v in T from their common ancestor in T is at most $H-1$. Thus, at most Δ_G^H components of $T^{(j)}$ can constitute the same component in $G^{(j)}$. Thus, each component in $G^{(j)}$ has $O(\Delta_G^{(k+1)H})$ nodes. Now, each independent set X of links in each component of $G^{(j)}$ is a matching in the corresponding component of $G^{(j)}$. The time needed to compute a maximum weighted matching in each such component is therefore $O(\Delta_G^{3(k+1)H})$. Thus, the overall computation time is $O(\Delta_G^{3(k+1)H})$. If G is a bipartite graph, the overall computation time is $O(\Delta_G^{2(k+1)H})$.

The diameter of any component of $T^{(j)}$ is $O(kH)$. Since a component of $G^{(j)}$ consists of at most Δ_G^H components of $T^{(j)}$, the diameter of any component of $G^{(j)}$ is $O(kH\Delta_G^H)$. Thus, H-LCPM (k) is in INFORMATION($kH\Delta_G^H$) class.

We now prove the following key result which will be used in obtaining throughput guarantees for H-LCPM (k).

Lemma 3: $L^{(0)}, \ldots, L^{(k-1)}$ is $4-$approximate.

Proof: Let $\vec{W}$ be an arbitrary N-dimensional vector of non-negative real numbers, $X^* = \arg\max_{X \in \mathcal{X}} \sum_{l \in X} W_l$, and $X_0, \ldots, X_{k-1}$ be arbitrary subsets of links such that $X_j \in \mathcal{X}$ (i.e., X_i is a matching) and $X_j \subseteq L^{(j)}$, $j = 0, \ldots, k-1$. We need to prove that $\sum_{j=0}^{k-1} \sum_{l \in X_j} W_l \leq 4 \sum_{l \in X^*} W_l$. Note that for any link l,

$$W_l \leq \sum_{i \in X^* \cap S_l} W_i.$$

Let $\eta_l^{(j)} = |X_j \cap S_l|$. Thus,

$$\sum_{j=0}^{k-1} \sum_{l \in X_j} W_l \leq \sum_{l \in X^*} \left(\sum_{j=0}^{k-1} \eta_l^{(j)} \right) W_l. \tag{5}$$

Hence, we need to show that $\left(\sum_{j=0}^{k-1} \eta_l^{(j)} \right) \leq 4$ for each $l \in X^*$.

Consider $l = (u, v) \in E$. Wlog, let level of u in T be less than or equal to that of v in T. There exists a unique j_l such that level of u in T is in $((j_l - 1)H, j_l H] \bmod kH$. Note that l is not adjacent to any link in $L^{(q)}$ where $q < (j_l - 1) \bmod k$ or $q > (j_l + 1) \bmod k$. Since X_js are matchings, at most 1 link in X_j is adjacent to l when $j \in \{(j_l - 1) \bmod k, (j_l + 1) \bmod k\}$, and at most 2 links in X_{j_l} are adjacent to l. Thus, $\left(\sum_{j=0}^{k-1} \eta_l^{(j)} \right) \leq 4$ for each $l \in X^*$. The result follows. ■

Theorem 2: If $\vec{\lambda} \in \text{Int}(\Lambda)$ and $\epsilon \in (0, 1)$, then $(1 - \epsilon)\vec{\lambda} \in \Lambda^{H\text{-LCPM}(\lceil 4/\epsilon \rceil))}$.

Using $k = \lceil 4/\epsilon \rceil$, $c = 4$, Theorem 2 follows from lemmas 3 and 2. Now, H-LCPM ($\lceil 4/\epsilon \rceil$) is in INFORMATION$\big(O(H\Delta_G^H/\epsilon)\big)$ class and requires $\Delta_G^{O(H/\epsilon)}$ computation time. Thus, H-LCPM will be useful for small values of H.

Note that both TPM ($\lceil 1/\epsilon \rceil$) and 1-LCPM ($\lceil 4/\epsilon \rceil$) approximate the maximum throughput region of trees within a factor of $1 - \epsilon$. But, they need different computation times: $O(\Delta_G/\epsilon)$ and $\Delta_G^{O(1/\epsilon)}$ respectively. Thus, H-LCPM ($\lceil 4/\epsilon \rceil$) should only be used for cyclic graphs.

C. Geometric Graphs

A graph is said to be geometric if nodes are embedded in the first quadrant of the 2-dimensional plane, and a link exists between nodes u and v if and only if the distance between them is less than a certain value say D. The distance D is referred to as the transmission range. We first consider the primary interference model (Section V-C.1) and subsequently consider the secondary interference model (Section V-C.2).

1) Geometric Graphs with primary interference model: We consider a geometric graph G with primary interference model. We now describe the $L^{(0)}, \ldots, L^{(k-1)}$ for the policy GEOMETRIC-GRAPH-PARTITION-MATCHING(k) which we abbreviate as GGPM(k). We will consider k different grids each of which consists of a series of horizontal and vertical lines parallel to the x and y axes respectively and the distance between any two closest horizontal (vertical) lines is kD. Each grid is specified by its first horizontal and vertical lines. The first horizontal and vertical lines of grid j are given by $y = jD$ and $x = jD$ respectively for $j = 0, \ldots k-1$. Now, $L^{(j)}$ is the set of links which either cross, or have at least one end node on, a vertical or a horizontal line of grid j (Figure 2(a)). Note that the links in a component of $G^{(j)}$ do not interfere with those in other components.

We first evaluate the computation time for GGPM(k). The overall computation time equals the worst case computation time in a component. Let ν be the maximum number of nodes in any component of $G^{(j)} = (V, E \backslash L^{(j)})$ for any j. We next show that ν is $O(\Delta_G k^2)$. Thus, the computation time for GGPM(k) is the time required to compute a maximum weighted matching in a component with $O(\Delta_G k^2)$ nodes, which is $O(\Delta_G^3 k^6)$. Also, GGPM(k) is in INFORMATION$(O(\Delta_G k^2))$ class.

Lemma 4: For any $j = 0, \ldots, k-1$, a component in $G^{(j)} = (V, E \setminus L^{(j)})$ has $O(\Delta_G k^2)$ nodes and $O(\Delta_G^2 k^2)$ links.

Proof: Consider some $j \in 0, \ldots, k-1$. A component in $G^{(j)}$ consists of nodes in a square enclosed by the closest horizontal and vertical lines of the jth grid. The side of such a square is at most kD units. Such a square can be filled with $O(k^2)$ small squares with sides slightly less than $D/\sqrt{2}$. Let I be a maximal independent set of nodes in the component, i.e., there does not exist an edge between any two nodes in I and every node in the component is either in I or has an edge to some node in I. Since the distance between any two points in any small square is less than D, at most one node in I is present in any small square. Therefore, $|I|$ is $O(k^2)$. Clearly, $\nu \leq |I|\Delta_G$. Thus, ν is $O(\Delta_G k^2)$. Also, the maximum number of links in any component of $G^{(j)}$ is at most $\nu\Delta_G$ which is $O(\Delta_G^2 k^2)$. ■

We now prove the following key result which will be used in obtaining throughput guarantees for GGPM(k).

Lemma 5: $L^{(0)}, \ldots, L^{(k-1)}$ is $12-$approximate.

Proof: The proof is similar to that for lemma 3. We point out the differences. We need to prove that $\sum_{j=0}^{k-1} \sum_{l \in X_j} W_l \leq 12 \sum_{l \in X^*} W_l$. Relation (5) holds in this case as well. Hence, we need to show that $\left(\sum_{j=0}^{k-1} \eta_l^{(j)}\right) \leq 12$ for each $l \in X^*$.

Now, note that the k grids do not share any common line. Let SUPERGRID consist of all lines of all grids. Then SUPERGRID is a grid where the distance between any two closest horizontal (vertical) lines is D.

Clearly, $\eta_l^{(j)} = 1$ for any $l \in X_j \cap X^*$. If $l \in X^* \setminus X_j$, $\eta_l^{(j)}$ is the number of links in X_j that interferes with l. Since these links are in X_j, they do not interfere with each other. Thus, $\eta_l^{(j)} \leq 2$ since at most 2 links can be adjacent to l but are not adjacent to each other. Thus, $\eta_l^{(j)} \leq 2$ for any $l \in X^*$.

Next, for each $l \in X^*$ we upper-bound the number of js in $\{0, \ldots, k-1\}$ such that $\eta_l^{(j)} > 0$. Now, $\eta_l^{(j)} > 0$ if either $l \in L^{(j)}$ or $l \notin L^{(j)}$ but l interferes with a link in $L^{(j)}$. Note that for any l, $l \in L^{(j)}$ for at most 2 js in $\{0, \ldots, k-1\}$. This holds because a link l can either cross or have an end node on at most 1 horizontal and vertical line of SUPERGRID. Next, for any l, $l \notin L^{(j)}$ but l interferes with (i.e., is adjacent to) a link in $L^{(j)}$ for at most 4 js in $\{0, \ldots, k-1\}$. This holds because if $l \notin L^{(j)}$ both end nodes of l are inside one square of the SUPERGRID, say square a. But, then l can be adjacent to links in $L^{(j)}$ if a side of square a is aligned with at least one horizontal or vertical line of grid j, which can happen for at most 4 values of j. Thus, for each $l \in X^*$, $\eta_l^{(j)} > 0$ for 6 js in $\{0, \ldots, k-1\}$. Hence, $\left(\sum_{j=0}^{k-1} \eta_l^{(j)}\right) \leq 2 \times 6 = 12$ for each $l \in X^*$. ■

Theorem 3: If $\vec{\lambda} \in \text{Int}(\Lambda)$ and $\epsilon \in (0,1)$, then $(1-\epsilon)\vec{\lambda} \in \Lambda^{\text{GGPM}(\lceil 12/\epsilon \rceil, \epsilon))}$.

Using $k = \lceil 12/\epsilon \rceil$, $c = 12$, Theorem 3 follows from lemmas 2 and 5. GGPM $(\lceil 12/\epsilon \rceil)$ is in INFORMATION$(O(\Delta_G/\epsilon^2))$ class and requires $O(\Delta_G^3/\epsilon^6)$ computation time. In the next subsection, we propose a technique that computes each schedule in $O(\Delta_G^2/\epsilon^2)$ time while approximating the maximum throughput region within a factor of $(1-\epsilon)$.

2) Geometric Graphs with Secondary Interference Model: We consider a geometric graph G and the secondary interference model. In this interference model, a link i interferes with link j if one end point of j is within distance D from an end point of i. Note that if two links interfere under the primary interference model they also interfere under the secondary interference model but the converse is not true. This model is an abstraction of bidirectional wireless links where all transmissions use a single channel and equal power. Note that an independent set of links is no longer a matching in G.

We now describe the $L^{(0)}, \ldots, L^{(k-1)}$ for policy GEOMETRIC-GRAPH-PARTITION-INDEPENDENT-SET(k) which we abbreviate as GGPIS(k). Just as in Section V-C.1, we consider k different grids. Now, $L^{(j)}$ is the set of links for which at least one end point is within a distance D of a vertical or horizontal line of grid j (Figure 2(b)). Note that the links in a component of $G^{(j)}$ do not interfere with those in other components.

We now evaluate the computation time for GGPIS(k). From lemma 4, each component of $G^{(j)}$ has $O(\Delta_G^2 k^2)$ links. Consider two links $l=(u,v), l'=(w,x)$ that do not interfere. Then no small square in the proof of lemma 4 can contain both u, w, or both u, x or both v, w or both v, x. Thus, the maximum size of any independent set of links in a component of $G^{(j)}$ is upper-bounded by the number of such small squares which again is $O(k^2)$. Thus, in any component of $G^{(j)}$, the maximum weighted interference set can be computed in $(\Delta_G^2 k^2)^{O(k^2)}$. Thus, the computation time for GGPIS(k) is $(\Delta_G^2 k^2)^{O(k^2)}$. Again, like GGPM (k), GGPIS(k) is in INFORMATION$(O(\Delta_G k^2))$ class.

We make the following observations about $L^{(0)}, \ldots, L^{(k-1)}$.

- Let $\psi_l = \{j : l \in L^{(j)}\}$. Then, $|\psi_l| \leq 6$ for any $l \in E$ (*Observation* 1). This holds because the end nodes of a link can be at a distance of D from at most 3 vertical and 3 horizontal lines of SUPERGRID.
- For any l, $l \notin L^{(j)}$ but l interferes with a link in $L^{(j)}$ for at most 4 js in $\{0, \ldots, k-1\}$ (*Observation* 2). This happens only if one of the end nodes of l is within $2D$ units of a horizontal or vertical line of grid j but none of the end nodes of l are within a distance of D from any line of grid j. This can happen at most 2 times for vertical lines and 2 more times for horizontal lines of SUPERGRID.

We now prove the following key result which will be used in obtaining throughput guarantees for GGPIS(k).

Lemma 6: $L^{(0)}, \ldots, L^{(k-1)}$ is $80-$approximate.

Proof: The proof is similar to that for lemma 5. Like in lemma 5, we need to prove that $\left(\sum_{j=0}^{k-1} \eta_l^{(j)}\right) \leq 80$ for each $l \in X^*$. Now, $\eta_l^{(j)} \leq 8$ for any $l \in X^*$ as the number of links that interfere with l but do not interfere with each other is at most 8 [2]. Next, from observations 1 and 2, for each $l \in X^*$, $\eta_l^{(j)} > 0$ for 10 js in $\{0, \ldots, k-1\}$ Hence, $\left(\sum_{j=0}^{k-1} \eta_l^{(j)}\right) \leq 8 \times 10 = 80$ for each $l \in X^*$. ■

Theorem 4: If $\vec{\lambda} \in \text{Int}(\Lambda)$ and $\epsilon \in (0,1)$, then $(1-\epsilon)\vec{\lambda} \in \Lambda^{\text{GGPIS}(\lceil 80/\epsilon \rceil))}$.

Using $k = \lceil 80/\epsilon \rceil$, $c = 80$, Theorem 4 follows from lemmas 2 and 6. GGPIS$(\lceil 80/\epsilon \rceil)$ is in INFORMATION$(O(\Delta_G/\epsilon^2))$ class and requires $(\Delta_G/\epsilon)^{O(1/\epsilon^2)}$ computation time.

We now present a policy, which we denote as GEOMETRIC-GRAPH-PARTITION-GRADUAL-IMPROVEMENT(k) and abbreviate as GGPGI(k), that for appropriate choice of k attains the same throughput guarantee as GGPGI$(\lceil 80/\epsilon \rceil)$ but requires a computation time of only $O(\Delta_G^2/\epsilon^2)$. Note that GGPGI(k) does not belong in the general class of policies $\pi(k)$ described in Section V-A. The main difference between GGPGI(k) and $\pi(k)$ (and hence GGPIS(k)) is that the former does not compute the maximum weight independent set of links in any component but in each component selects an independent set of links which has a higher weight than that selected in a previous epoch. Note that Tassiulas [16] proved that the stability region can be maximized by using a similar selection strategy in the entire graph. We prove that, by appropriately partitioning the graph, the stability region can be approximated arbitrary closely if the above selection policy is used in each component. This combination of graph partitioning and improvement based selection schemes reduce the computation time from $\Theta(N)$ attained using only the improvement based selection schemes in [16] to $O(\Delta_G^2/\epsilon^2)$.

In GGPGI(k) each link l is associated with $k-6$ secondary virtual queues: $Q_{li}^{(S)}$, $i \in \{0,\ldots,k-1\}\setminus\hat{\psi}_l$ where $\hat{\psi}_l$ is the union of ψ_l and $\max(0, 6-|\psi_l|)$ arbitrary elements of $\{0,\ldots,k-1\}\setminus\psi_l$. Whenever a packet arrives in the virtual queue Q_l it is routed to one of the secondary virtual queues with equal probability. The policy divides the time axis in frames of k slots. In the jth slot of each frame, for different links $l \in E$, the secondary virtual queues $Q_{lj}^{(S)}$ contend. Only the secondary virtual queues that contend can be scheduled for transmission and those that are scheduled for transmission transmit their head of line packets if they are non-empty.

We now describe which contending secondary virtual queues are scheduled for transmission in the jth slot of each frame. Note that $Q_{lj}^{(S)}$ does not exist if $l \in L^{(j)}$ as then $j \in \psi_l \subseteq \hat{\psi}_l$. Thus, in the jth slot of each frame, no secondary virtual queue associated with any link $l \in L^{(j)}$ contends and at most one secondary virtual queue associated with each link $l \in E\setminus L^{(j)}$ contends. A link is said to contend if one secondary virtual queue associated with it contends. Thus, for each j the links that contend in the jth slot of each frame constitute components such that links in different components do not interfere, and the links in each component are a-priori rank ordered in some manner. Links in the ordered list sequentially select themselves with a probability $p \in (0,1)$ and those that interfere with the selected links remove themselves from the list. The weight of each contending link is the number of packets waiting for transmission in the contending secondary virtual queue associated with it. The selected links are scheduled in each component if their total weight exceeds the total weight of the links scheduled in the same component in the jth slot of the previous frame; otherwise the links scheduled in the same component in the jth slot of the previous frame are scheduled again. The contending secondary virtual queues associated with the scheduled links are scheduled.

The computation time of GGPGI(k) is clearly $O(\gamma)$ where γ is the maximum number of links in any component of $G^{(j)}$; hence this computation time is $O(\Delta_G^2 k^2)$. Also, GGPGI(k) is in INFORMATION$(O(\Delta_G k^2))$ class.

Theorem 5: If $\vec{\lambda} \in \mathrm{Int}(\Lambda)$ and $k \geq 6$, then $(1-6/k)\vec{\lambda} \in \Lambda^{\mathrm{GGPGI}(k))}$.

Proof: The result clearly holds for $k=6$. We therefore assume that $k>6$.

Consider a fictitious system that consists of only the secondary virtual queues Q_{lj} for all l. Let $\hat{\Lambda}^{(j)}$ be the maximum throughput region of this fictitious system. Then [17]

$$\mathrm{Int}(\hat{\Lambda}^{(j)}) = \{\vec{\lambda} : \vec{\lambda} = \sum_{\substack{\vec{X}\in\mathcal{X} \\ X\subseteq E\setminus\{l:j\notin\hat{\psi}_l\}}} \beta_X \vec{I}^X, \text{ where } \sum_{\substack{\vec{X}\in\mathcal{X} \\ X\subseteq E\setminus\{l:j\notin\hat{\psi}_l\}}} \beta_X = 1, \beta_X \geq 0 \text{ for each } X\in\mathcal{X} \text{ and } \beta_\phi > 0\}.$$

Consider a policy π that schedules secondary virtual queues that satisfy the following properties.

1) $Q_{lj}(t)$ constitutes an irreducible aperiodic markov chain.
2) In each slot t there is a positive probability associated with scheduling the secondary virtual queues associated with links l in $X^*(t)$ where
$$X^*(t) = \arg \max_{\substack{X\in\mathcal{X} \\ X\subseteq E\setminus\{l:j\notin\hat{\psi}_l\}}} \sum_{l\in X} Q_{lj}(t).$$
3) If X_0 and X_1 are the sets of links associated with the secondary virtual queues scheduled in slots $t-1$ and t then $\sum_{l\in X_1} Q_{lj}(t) \geq \sum_{l\in X_0} Q_{lj}(t)$.

Then π stabilizes the fictitious system for any arrival rate vector $\vec{\lambda}' \in \mathrm{Int}(\hat{\Lambda}^{(j)})$ [16], [5].

Let $(1-6/k)\vec{\lambda}$ be the arrival rate vector in the system and let $\vec{\lambda} \in \mathrm{Int}(\Lambda)$. Let $\vec{\lambda}^{(j)}$ consist of those components l of $\vec{\lambda}$ for which $j \notin \hat{\psi}_l$. From (3), $\vec{\lambda}^{(j)} \in \mathrm{Int}(\Lambda^{(j)})$.

We now sample the secondary virtual queues Q_{lj} for all l at slots $j, k+j, 2k+j, \ldots$ in the actual system. Note that in the actual system these secondary virtual queues are scheduled only in these slots.

We assume that the number of arrivals in slot $mk + j$ in the secondary virtual queue Q_{lj} in the sampled system is the number of arrivals in Q_{lj} in the actual system between slots $((m-1)k + j, mk + j]$ ($[0, j]$) for a positive integer m ($m = 0$). Note that the expected number of arrivals in secondary virtual queue Q_{lj} in the sampled system in slot $mk + j$ is now $k(1/(k-6))(1-6/k)\lambda_l = \lambda_l$. Thus, the arrival rate vector for these secondary virtual queues is $\vec{\lambda}^{(j)} \in \mathrm{Int}(\Lambda^{(j)})$. Now, observe that GGPGI(k) satisfies properties (1) to (3) for these secondary virtual queues in the sampled system, since links that contend in different components of $G^{(j)}$ do not interfere. Thus, the sampled system is stable for each j. The result follows. ∎

Thus, for $k = \lceil 6/\epsilon \rceil$, a policy GGPGI($k$) in INFORMATION($O(\Delta_G^2/\epsilon^2)$) class, approximates the maximum throughput region within a factor of $1 - \epsilon$ using a computation time of $O(\Delta_G^2/\epsilon^2)$. Note that GGPM(k) can be similarly modified to attain the same throughput guarantee, using $k = \lceil 4/\epsilon \rceil$ and a computation time of $O(\Delta_G^2/\epsilon^2)$. More generally, for r-ary interference models, i.e., when two links interfere provided an end node of one is within a distance of $(r-1)D$ of an end node of the other, similar techniques can be used to approximate the maximum throughput region within a factor of $1 - \epsilon$ using a computation time of $O\left(f(r)\Delta_G^2/\epsilon^)\right)$ where $f(r)$ increases with increase in r.

Finally, as compared to GGPIS(k), GGPGI(k) attains substantially better tradeoffs between throughput and computation time guarantees. But, at the same time GPGIIS(k) is likely to have substantially higher delay as compared to GGPGI(k). This is because since, unlike GGPGI(k), GGPGI(k) seggregates the incoming traffic in each link in multiple queues and in each slot allows at most one queue in each link to contend, when the contending queue is empty it does not schedule the link even if the link's interferers are not scheduled and other queues in the same link are non-empty. Also, unlike GGPGI(k), GGPGI(k) does not schedule the queues whose expected weight is close to that of the maximum weight independent set of queues, and instead attains stability by gradually improving the weight of the scheduled queues. This behavior is known to significantly increase the delay, e.g., simulations have demonstrated that the policy proposed by Tassiulas *et al.* [17] that schedules the maximum weight independent set in each slot has substantially lower delay as compared to the randomized policy proposed again by Tassiulas [16] that attains stability through similar improvements as above. An interesting topic for future research is to investigate the tradeoffs between delay and computation times of scheduling policies.

D. Quasi-Geometric Graphs

A graph is said to be quasi-geometric if nodes are embedded in the first quadrant of the 2-dimensional plane, and a link (a) exists between nodes u and v if the distance between them is less than ιD where $\iota < 1$ (b) may exist between nodes u and v if the distance between them is between ιD and D and (c) does not exist between nodes u and v if the distance between them is greater than or equal to D. Under primary interference constraints, as before, two links interfere if and only if they are adjacent. Under secondary interference constraints, two links l, l' interfere if and only if (a) they are adjacent and (b) there is an edge between at least one end node of l and another end node of l'.

We first consider the secondary interference model. Now, links $L^{(0)}, \ldots, L^{(k-1)}$ are selected as in the previous subsection, and GGPGI (k) attains a throughput region which is $1-6/k$ of the maximum throughput region as before. However, each component of $G^{(j)}$ has $O(\Delta_G k^2/\iota^2)$ nodes, and $O(\Delta_G^2 k^2/\iota^2)$ links. Thus, the computation time for GGPGI (k) is $O(\Delta_G^2 k^2/\iota^2)$. Also, GGPGI ($k$) is in INFORMATION($O(\Delta_G k^2/\iota^2)$) class. Thus, GGPGI ($\lceil 6/\epsilon \rceil$) approximates the maximum throughput region within a factor of $1 - \epsilon$ while requiring a computation time of $O\left(\Delta_G^2/(\iota^2\epsilon^2)\right)$ and is in INFORMATION($O\left(\Delta_G/\left(\iota^2\epsilon^2\right)\right)$) class. Similarly, under the primary interference model, a throughput region of $1-\epsilon$ of the maximum throughput region can be attained using a policy in INFORMATION($O(\Delta_G/\left(\iota^2\epsilon^2\right))$) class which requires $O(\Delta_G^2/(\epsilon^2\iota^2))$ computation time.

E. Distributed Implementation of the Scheduling Policies

We discuss two possible distributed implementations for $\pi(k)$. In one, in each slot one "master" link in each component determines which links will be scheduled in the component and broadcasts the decisions in the entire component, and in another every link does this computation. For the first each contending link communicates its weight to the master link in its component, and for the second each contending link broadcasts its weight in its entire component. The problem with both implementations is that the size of the packets can not be bounded by any function of the network size since the queue lengths exceed any given number with positive probability. A better solution is to have each link broadcast the increase in its weight since the previous epoch in which the link was in the same component (that is the same random number was selected). Now, the expectation of the magnitude of this increase is $O(k)$ as the expected difference between the consecutive epochs at which the same random number is selected is $O(k)$. Furthermore, the time consumed by the broadcasts in each component is a linear function of the number of links in the component. Thus, the overall expected computation times for the policies are as given in subsections V-B to V-D.

VI. Multi-hop sessions

We now allow sessions to traverse multiple hops. We first describe the modifications required in the system model and performance goals for accommodating this generalization. We subsequently generalize the framework presented in Section V for attaining arbitrary tradeoffs between throughput guarantees and computation times.

A. Generalized System Model

We now assume that the network consists of $\hat{N}$ end-to-end sessions, indexed as $1, \ldots, \hat{N}$. Each end-to-end session can be viewed as a collection of several hop-by-hop connections, one for each link it traverses; each of these hop-by-hop connections is called a *session-link* of the session considered. Each session-link is of the form (u, v), where u and v represent the transmitter and the receiver, respectively, of the session-link. We assume that there are M session-links in the network (over all sessions), and these are indexed by $1, \ldots, M$. The interference relations are as in Section III.

Each session-link corresponds to a separate virtual queue and the number of virtual queues associated with each link equals the number of session-links traversing it; we assume that this number is at most μ. The packet arrival process is the same as before, and only the first session-link of each session receives the exogenous arrivals. Now, the queue-length and departure vectors, $\vec{Q}(t), \vec{D}(t)$, are M-dimensional vectors respectively representing the queue lengths of the session-links and which session-links are served in slot t.

Let R be a $M \times M$ dimensional matrix such that (a) $R_{ij} = 1$ if $i = j$ (b) $R_{ij} = -1$ if i and j are session-links of the same session and i constitutes the hop after j and (c) $R_{ij} = 0$ otherwise.

$$\vec{Q}(t+1) = \vec{Q}(t) - R\vec{D}(t) + \vec{A}(t).$$

The definition for stability is the same except that session-links are considered instead of sessions. The definitions for the throughput regions are the same as before.

B. Scheduling policies for approximating the maximum throughput region arbitrary closely

We now generalize the policy $\pi(k)$ presented in Section V. The modified policy, denoted as $\pi^{\text{MH}}(k)$, differs from $\pi(k)$, in only the assignment of link weights. For $\pi^{\text{MH}}(k)$ in any slot t, the weight of a session-link (or a virtual-queue) $l = (u, v)$ of session i, $G_i(t)$, is (a) the difference between the queue lengths of session-links l and m where m is the session-link of i originating from v, if v is not the destination for i and (b) $Q_i(t)$ otherwise. The weight of a link is the maximum weight of a session-link traversing the link. Note that in the special case that each session traverses one link, for any virtual-queue $i = (u, v)$, v is the destination of the session and hence its weight $G_i(t)$ equals $Q_i(t)$ as in Section V. Whenever a link is scheduled, the session-link that has the maximum weight among those that traverse the link is served. The policies $\pi^{\text{MH}}(k)$ and $\pi(k)$ are otherwise the same.

Lemma 7: Let $L^{(0)}, \dots, L^{(k-1)}$ be c-approximate. Then, if $\vec{\lambda} \in \mathrm{Int}(\Lambda)$ and $k \geq c > 0$, then $(1-c/k)\vec{\lambda} \in \Lambda^{\pi^{\mathrm{MH}}(k)}$.

We prove lemma 7 in appendix.

We now consider the throughput guarantees of π^{MH} for different classes of networks considered in subsections V-B to V-D. The choice of $L^{(0)}, \dots, L^{(k-1)}$ for different classes of networks remain the same as in subsections V-B to V-D. Using $k = \lceil 4/\epsilon \rceil$, $c = 4$, Theorem 2 follows from lemmas 2 and 3 for H-LCPM$^{\mathrm{MH}}(k)$. Using $k = \lceil 12/\epsilon \rceil$, $c = 12$, Theorem 3 follows from lemmas 2 and 5 for GGPM$^{\mathrm{MH}}(k)$. Using $k = \lceil 80/\epsilon \rceil$, $c = 80$, Theorem 4 follows from lemmas 2 and 6 for GGPIS$^{\mathrm{MH}}(k)$.

Clearly, the computation times in each case increase only by an additive term of μ; this increase is necessary to compute the weight of each link as the maximum of weights of μ virtual queues associated with it.

VII. Conclusion

The contribution of this paper is to provide a framework for attaining arbitrary close approximations of the maximum throughput region in a large class of wireless networks using a computation time that depends only on the approximation factor and the maximum node degree in the network. The throughput guarantees have been proved under the assumption that the arrival process for each session is independent and identically distributed across different slots. Using a combination of graph-partitioning and the lyapunov techniques proposed in [15], the proofs can be generalized to accommodate markov modulated arrival processes. Also, under the weaker notion of rate stability which only ensures that input rates equal the output rates, the graph partitioning techniques may be combined with fluid-limit arguments so as to obtain similar tradeoffs between throughput and computation times for all stationary ergodic arrival processes that satisfy the strong law of large numbers.

VIII. Acknowledgement

We would like to thank Professors Sudipto Guha at University of Pennsylvania and Kamesh Munagala at Duke University for numerous discussions on graph partitioning techniques and algorithms for approximating maximum weight independent sets.

Appendix

Proof for lemma 7

We first state and prove lemma 8 for $\pi^{\mathrm{MH}}(k)$ which will be useful in proving lemma 7.

Lemma 8: Let $L^{(0)}, \dots, L^{(k-1)}$ be c-approximate. Then,

$$\mathbf{E}\left(\sum_i G_i(t) D_i(t) | \vec{Q}(t) = \vec{Q}\right) \geq (1 - c/k) \max_{X \in \mathcal{X}} \sum_{i \in X} G_i(t).$$

Proof:

$$\text{Let } B(t) = \arg \max_{\substack{X \in \mathcal{X} \\ X \subseteq L^{(i(t))}}} \sum_{l \in X} G_l(t).$$

Again, $\sum_i G_i(t)D_i(t) \geq \left(\max_{X\in\mathcal{X}} \sum_{i\in X} G_i(t) - \sum_{i\in B(t)} G_i(t)\right)$. Now,

$$\begin{aligned}
\mathbb{E}\left(\sum_{l\in B(t)} G_l(t)/\vec{Q}(t)\right) &= (1/k)\sum_{j=0}^{k-1} \max_{\substack{X\in\mathcal{X}\\ X\subseteq L^{(j)}}} \sum_{l\in X} G_l(t) \\
&\quad \text{(using same arguments as in the proof for lemma 1)} \\
&= (1/k)\sum_{j=0}^{k-1} \max_{\substack{X\in\mathcal{X}\\ X\subseteq L^{(j)}}} \sum_{l\in X} \max\left(G_l(t), 0\right) \\
&\leq (c/k)\max_{X\in\mathcal{X}} \sum_{i\in X} G_i(t) \text{ (since } L^{(0)}, \ldots, L^{(k-1)} \text{ are } c-\text{approximate).}
\end{aligned}$$

The result follows. ∎

We now prove lemma 7. This proof follows from lemma 8 using techniques similar to those used by Tassiulas *et al.* in [17].

Proof: The result clearly holds if $k = c$. We therefore assume that $k > c$. Let the arrival rate vector be $(1-c/k)\vec{\lambda}$ where $\vec{\lambda} \in \text{Int}(\Lambda)$. Clearly, under $\pi^{\text{MH}}(k)$, $\vec{Q}(t)$ constitutes an aperiodic irreducible Markov chain. We will consider the lyapunov function $V(\vec{Q}) = \sum_i Q_i^2$, and prove that under π, $\mathbf{E}\left(V\left(\vec{Q}\left(t+1\right)\right) - V\left(\vec{Q}(t)\right) | \vec{Q}(t) = \vec{Q}\right) < -1$ for all sufficiently large $||\vec{Q}||$, where $||\vec{Q}|| = \sqrt{V(\vec{Q})}$. Then, from Foster's theorem (Theorem 2.2.3 in [6]) the Markov chain representing the queue length process is positive recurrent. Also, $\mathbf{E}\left(Q_i(t)\right) < \infty$ for each i under the steady state distribution for the above Markov chain. Thus, $\lim_{K\rightarrow\infty} \frac{\sum_{t=0}^{K-1} Q_i(t)}{K} < \infty$. The result follows.

Let $q(j)$ denote the session of session-link j. Let $\vec{f}$ be an M-dimensional vector such that $f_i = \lambda_{q(i)}$. Then, $\text{Int}(\Lambda)$ can be characterized as follows [17]:

$$\text{Int}(\Lambda) = \{\vec{\lambda} : \vec{\lambda} = \sum_{\vec{X}\in\mathcal{X}} \beta_X R\vec{I}^X, \text{ where } \sum_{\vec{X}\in\mathcal{X}} \beta_X = 1 \text{ and } \beta_X \geq 0 \text{ for each } X \in \mathcal{X} \text{ and } \beta_\phi > 0\}. \quad (6)$$

Now,

$$\begin{aligned}
\mathbf{E}\left(\left(\vec{A}(t)\right)^T \vec{Q}(t)/\vec{Q}(t) = \vec{Q}\right) &= (1-c/k)\vec{\lambda}^T\vec{Q} \\
&= (1-c/k)\sum_{X\in\mathcal{X}} \beta_X \left(\left(R\vec{I}^X\right)^T \vec{Q}\right) \text{ where } \beta_\phi > 0 \text{ (from (6))} \\
&\leq (1-c/k)(1-\beta_\phi)\max_{X\in\mathcal{X}} \sum_{i\in X} G_i \qquad (7)
\end{aligned}$$

$$\begin{aligned}
&\mathbf{E}\left(V\left(\vec{Q}(t+1)\right) - V\left(\vec{Q}(t)\right) | \vec{Q}(t) = \vec{Q}\right) \\
= \;& \mathbf{E}\left(\left(\vec{A}(t) - R\vec{D}(t)\right)^T \left(2\vec{Q}(t) + \vec{A}(t) - R\vec{D}(t)\right) | \vec{Q}(t) = \vec{Q}\right) \\
\leq \;& 2\mathbf{E}\left(\left(\vec{A}(t) - R\vec{D}(t)\right)^T \vec{Q}(t) | \vec{Q}(t) = \vec{Q}\right) + M\hat{\alpha}_{\max}^2 \\
\leq \;& 2\mathbf{E}\left(\vec{A}^T(t)Q\vec{(t)} | \vec{Q}(t) = \vec{Q}\right) - 2\mathbf{E}\left(\sum_i G_i(t)D_i(t) | \vec{Q}(t) = \vec{Q}\right) + M\hat{\alpha}_{\max}^2 \\
\leq \;& -2(1-c/k)\beta_\phi \max_{X\in\mathcal{X}} \sum_{i\in X} G_i + M\hat{\alpha}_{\max}^2 \text{ (from Lemma 8 and (7))} \\
< \;& -1 \text{ for all sufficiently large } ||\vec{Q}|| \text{ (since } \beta_\phi > 0, 0 < c < k).
\end{aligned}$$

∎

References

[1] A. Brzezinski, G. Zussman, and E. Modiano. Distributed throughput maximization in wireless mesh networks - a partitioning approach. In *Proceedings of ACM MOBICOM*, Los Angeles, CA, September 2006.

[2] P. Chaporkar, K. Kar, and S. Sarkar. Throughput guarantees through maximal scheduling in multihop wireless networks. In *Proceedings of 43d Annual Allerton Conference on Communication, Control and Computing*, Allerton, Monticello, Illinois, September 28-30 2005.

[3] J. Dai and B. Prabhakar. The throughput of data switches with and without speedup. In *Proceedings of INFOCOM*, pages 556–564, Tel Aviv, Israel, Mar 2000.

[4] A. Dimakis and J. Walrand. Sufficient conditions for stability of longest queue first scheduling: second order properties using fluid limits. *Advances of applied Probability*, 38(2):505–521, June 2006.

[5] D. Shah E. Modiano and G. Zussman. Maximizing throughput in wireless networks via gossiping. In *Proc. ACM SIGMETRICS / IFIP Performance'06*, June 2006.

[6] G. Fayolle, V. A. Malyshev, and M. V. Menshikov. *Topics in the Constructive Theory of Countable Markov Chains*. Cambridge University Press, 1995.

[7] B. Hajek and G. Sasaki. Link scheduling in polynomial time. *IEEE Transactions on Information Theory*, 34(5):910–917, Sep 1988.

[8] H. Hunt, M. Marathe, Radhakrishnan V, S. Ravi, D. Rosenkrantz, and R. Stearns. Nc-approximation schemes for NP- and PSPACE-HARD problems in geometric graphs. *Journal of Algorithms*, 26(2), February 1998.

[9] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer. Local approximation schemes for ad hoc and sensor networks. In *Proceedings of DIALM-POMC*, Cologne, Germany, September 2 2005.

[10] X. Lin and S. Rasool. Constant-time distributed scheduling policies for ad hoc wireless networks. In *Proceedings of IEEE CDC-ECC'05*, San Diego, CA, Dec 2006.

[11] X. Lin and N. Shroff. The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks. In *Proceedings of INFOCOM*, Miami, FL, Mar 2005.

[12] T. Nieberg, J. Hurink, and W. Kern. A robust ptas for maximum weight independent sets in unit disk graphs. In *Graph Theoretic Concepts in Computer Science*, pages 214–221, Heidelberg, Germany, 2004. Springer Berlin.

[13] S. Sarkar and K. Kar. Achieving 2/3 throughput approximation with sequential maximal scheduling under primary interference constraints. In *Proceedings of 44th Annual Allerton Conference on Communication, Control and Computing*, Allerton, Monticello, Illinois, September 27-29 2006.

[14] G. Sharma, R.R. Mazumdar, and N.B. Shroff. On the complexity of scheduling in wireless networks. In *Proceedings of ACM MOBICOM*, Los Angeles, CA, September 2006.

[15] L. Tassiulas. Scheduling and performance limits of networks with constantly changing topology. *IEEE Transactions on Information Theory*, 43(3):1067–1073, May 1997.

[16] L. Tassiulas. Linear complexity algorithms for maximum throughput in radio networks and input queued switches. In *Proceedings of INFOCOM*, pages 533–539, 1998.

[17] L. Tassiulas and A. Ephremidis. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, Dec 1992.

[18] X. Wu and R. Srikant. Regulated maximal matching: a distributed scheduling algorithm for multihop wireless networks with node-exclusive spectrum sharing. In *Proceedings of IEEE CDC-ECC'05*, Seville, Spain, Dec 2005.