University of Pennsylvania
ScholarlyCommons

Departmental Papers (ESE) | Department of Electrical & Systems Engineering

November 2008

Arbitrary Throughput Versus Complexity Tradeoffs in Wireless Networks Using Graph Partitioning

Saswati Sarkar
University of Pennsylvania, swati@seas.upenn.edu

Saikat Ray
University of Bridgeport

Follow this and additional works at: http://repository.upenn.edu/ese_papers

Recommended Citation

Saswati Sarkar and Saikat Ray, "Arbitrary Throughput Versus Complexity Tradeoffs in Wireless Networks Using Graph Partitioning", . November 2008.

Copyright 2008 IEEE. Reprinted from *IEEE Transactions on Automatic Control*, Volume 53, Issue 10, November 2008, pages 2307-2323.

This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of the University of Pennsylvania's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

This paper is posted at ScholarlyCommons. http://repository.upenn.edu/ese_papers/463
For more information, please contact libraryrepository@pobox.upenn.edu.

Arbitrary Throughput Versus Complexity Tradeoffs in Wireless Networks Using Graph Partitioning

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 using 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. Then we develop a framework for attaining arbitrary close approximations for the maximum throughput region in arbitrary networks, 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.

Keywords
tree-partition-mapping (TPM)

Comments
Copyright 2008 IEEE. Reprinted from *IEEE Transactions on Automatic Control*, Volume 53, Issue 10, November 2008, pages 2307-2323.

This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of the University of Pennsylvania's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

This journal article is available at ScholarlyCommons: http://repository.upenn.edu/ese_papers/463

Arbitrary Throughput Versus Complexity Tradeoffs in Wireless Networks Using Graph Partitioning

Saswati Sarkar, *Member, IEEE*, and Saikat Ray

***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 using 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. Then we develop a framework for attaining arbitrary close approximations for the maximum throughput region in arbitrary networks, 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.**

***Index Terms*—Tree-partition-mapping (TPM).**

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 should transmit packets at a given time so as to avoid packet collisions. Moreover, the transmission schedules cannot be precomputed as the number of packets waiting at nodes as well as the transmission conditions in the wireless medium vary with time, and the statistics of these temporal variations are oftentimes not known *a priori*. The transmission schedules need to be computed at every transmission epoch. Thus, the schedule computation time 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.

The lack of a central controller dictates that each link needs to determine at every transmission epoch whether or not it would transmit based on its own state and the information it acquires about the states of other nodes. The state of a node or link comprises of attributes that change in the time scale of packet transmission: e.g., queue lengths and scheduling decisions. The time required for each link (or rather the node which is the source of the link) to decide whether to transmit or not at any given time depends on the time required (a) to exchange messages with other links to learn their states and (b) to compute the decision based on the information acquired. We refer to the total time required in both parts as the computation time of each schedule, or simply the computation time. The throughput guarantees usually improve with increase in the information each link acquires about the states of other links, but fetching information about distant links (nodes) require longer time. 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 into two broad classes: the policies that require each link to know attributes that depend on the states of (a) all links in the network [4], [27], [28] and (b) only the links that interfere with it (one-hop interferers) [2], [16], [17], [23], [29]. We refer to the two classes as INFORMATION (N) and INFORMATION (1) policies respectively, where N refers to the number of links in the network. By this nomenclature, then, INFORMATION (k) is the class of policies that require each link to learn the states of their k-hop interferers. A seminal result shows that the INFORMATION (N) class contains policies that attain the maximum possible throughput region in arbitrary wireless networks while computing each schedule in $O(N)$ time [27]. Recently, it has been shown that a policy in INFORMATION (1) class can attain a guaranteed fraction of the maximum throughput region using $O(\Delta_G \log N)$ time for computing each schedule 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 computing each schedule in an amount of 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 where any set of links that contains no two links with a common node can

Manuscript received August 27, 2007; revised February 18, 2008. Current version published November 05, 2008. This paper was presented in part at the Information Theory and Applications WorkshopUniversity of California, San Diego, CA, February 2007. This work was supported by the National Science Foundation under Grants NCR 0238340, CNS-0435306, ECCS 0621782, and CNS 0721308. Recommended by Associate Editor I. Paschalidis.

S. Sarkar is with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: swati@seas.upenn.edu,).

S. Ray is with the Department of Electrical Engineering, University of Bridgeport, Bridgeport, CT 06601 USA (e-mail: saikatr@bridgeport.edu).

Digital Object Identifier 10.1109/TAC.2008.2006820

0018-9286/$25.00 © 2008 IEEE

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

be simultaneously scheduled. Under this interference model and tree network topology, given any positive constant ϵ, we obtain a 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 $O(\Delta_G/\epsilon)$ for each schedule (Section IV). This policy requires no actual computation! Each link with a packet to transmit simply waits until its parent and older siblings (all of which are adjacent to the link) take scheduling decisions, and if all of them decide not to transmit, it transmits. Thus, a link need only communicate its scheduling decision to its children and younger siblings, and no queue length information is communicated.

Next we present a general framework for designing INFORMATION (k) policies for approximating the throughput region arbitrarily closely (Section V). We subsequently use this framework for obtaining arbitrary tradeoffs between throughput guarantees and computation times for large classes of networks: (i) graphs with limited cyclicity under the primary interference model (Section V-B) and (ii) geometric and quasi-geometric graphs under both primary and secondary interference models (Sections V-C, V-D). For example, for geometric graphs, given $\epsilon > 0$, we obtain a scheduling policy in INFORMATION $(O(\Delta_G/\epsilon^2))$ class that (a) approximates the throughput region within a factor of $1-\epsilon$ and (b) computes each schedule in $O(\Delta_G^2/\epsilon^2)$ time. We upper bound the expected delays attained by these policies and prove that the bounds are comparable to the best known guarantees in these networks. The throughput and computation time guarantees extend to networks where sessions traverse multiple links (Section VI).

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 in one component that interfere with those in another component are "shut down" i.e., not scheduled. Hence, scheduling among the residual links 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. The links that are scheduled in each component maximize the throughput region of the component; the reduction in the overall throughput region may happen only because of the "shut down" links. This reduction in throughput is kept small using different partitioning schemes at different times that ensure that each link is shut down only a small fraction of time 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. A major challenge in proving the analytical results has been that standard results in graph partitioning and approximation of throughput regions do not apply owing to this combination. For example, the following result is often used for approximating the throughput region: if a scheduling policy ensures that the sum of the queue lengths of the links that transmit packets is within a factor c of the maximum sum of the queue lengths of the links in any valid schedule, then the throughput region of the policy is within a factor c of the maximum throughput region [17]. Since a valid schedule in a network with N links can oftentimes be represented as an independent set in a graph with N links, such schedules can be computed if the maximum weighted independent set in such graphs can be approximated within a factor of c. Existing graph partitioning schemes can be used for attaining the above in geometric graphs and secondary interference model for c arbitrarily close to 1, and existing matching algorithms can attain the above in trees under primary interference model for $c = 1$. But, all such schemes need a $\Omega(N)$ computation time [9], [13], [20]. Thus, such schemes can not be directly used to obtain arbitrary tradeoffs between throughput guarantees and computation times for each schedule. We circumvent this difficulty by proving that in a large class of networks, given any $\epsilon > 0$, simple randomized partitioning schemes can be used to (a) obtain independent sets such that the expected weight of such an independent set is within $1-\epsilon$ of the maximum weight of an independent set for any allocation of non-negative weights, (b) while requiring a computation time that depends only on Δ_G and ϵ. The above property may be useful for approximating maximum weighted independent sets in expected sense in other contexts as well, and is therefore an interesting result in its own right (Appendix B). It also turns out that if the scheduling policy ensures that the expected sum of the queue lengths of the links that transmit packets is within a factor c of the maximum sum of the queue lengths of the links in any valid schedule, then the throughput region of the policy is within a factor c of the maximum throughput region. Together, these results have enabled the design of scheduling policies that obtain arbitrary tradeoffs between throughput guarantees and computation times for each schedule. Finally, note that the simple scheduling scheme we proposed for trees does not approximate, even in expected sense, the maximum weighted schedule within any factor in any slot. The proof in this case relies on an appropriate choice of a Lyapunov function that captures artifacts introduced by the policy and the graph partitioning techniques.

II. RELATED LITERATURE

Tassiulas *et al.* characterized the maximum throughput region and provided a policy that attains this throughput region in an arbitrary wireless network [28]. This policy schedules the maximum weighted independent set of links in each slot, and hence requires $\Theta(2^N)$ time for computing each schedule unless $P = NP$. A minor modification of the proof shows that if the schedule is computed as above once every T slots and subsequently used for transmitting T packets, then the throughput region does not change as long as T is finite. Thus, by using $T = \Theta(2^N)$, the maximum throughput region can be obtained while devoting $O(1)$ fraction of total time in computing the schedules. This infrequent schedule computation is however likely to substantially increase packet delays and packet loss when nodes have finite buffers. Schedules can be computed frequently if the time for computing each schedule is reasonable. Thus, subsequent research attempted to maximize the throughput region subject to constraints on the computation time of each schedule.

Tassiulas [27] provided randomized scheduling schemes that attain the maximum achievable throughput region while requiring $\Theta(N)$ time to compute each schedule for arbitrary interference models. In each slot, this policy randomly selects

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

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. Modiano *et al.* [5] have shown that gossip based algorithms can be used to implement the above policy for arbitrary interference models in networks where nodes do not have unique identities and know only limited information about the global topology such as path lengths, number of nodes in the network etc. Dimakis *et al.* [4] have shown that a greedy maximal weight scheduling, which requires $\Theta(N)$ time to compute each schedule, attains the maximum throughput region in several different networks. All the above policies are in the INFORMATION (N) class.

Chaporkar *et al.* [2] proved that a simple greedy scheduling scheme, maximal independent set selection, which can be computed in $\Theta(\Delta_G \log N)$ time [10], attains guaranteed fraction of the maximum throughput region for arbitrary interference models. The guarantees depend on the interference model, e.g., 1/2 for primary interference [3], [17], [29], 1/8 for geometric graphs under secondary interference model [2], etc., and can not be made arbitrarily close to 1 [2]. Sarkar *et al.* [23] proved that for the primary interference model and tree graphs, a queue length dependent maximal matching attains 2/3 of the throughput region while using $\Theta\left(\Delta_G(\log \Delta_G) \log N\right)$ time for computing each schedule. Lin *et al.* [16] 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 $O(\Delta_G)$ time for computing each schedule. All these policies are in the INFORMATION (1) class.

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.* [9], Kuhn *et al.* [13], Nieberg *et al.* [20] have devised graph partitioning techniques for approximating maximum weighted independent sets in geometric graphs within a factor of $1-\epsilon$ using policies in INFORMATION (N) class which have computation times of $\Theta\left(N + \Delta_G^{\Theta(1/\epsilon^2)}\right)$. The computation time depends on N as the policies consider several different partitions of the graph, computes the maximum weighted independent set for each partition, and selects the independent set that has the maximum weight among the above. 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. The partitioning technique used in [13] however requires $\Delta_G^{\Theta(1/\epsilon^2)}$ time for computing a maximum size independent set which does not depend on N, 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. Brzezinski *et al.* [1] and Sharma *et al.* [24] have recently used graph partitioning schemes for spectrum allocation and maximum weight independent set selection in wireless networks.

For geometric graphs, our framework yields a policy in the INFORMATION $\left(O(\Delta_G/\epsilon^2)\right)$ 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. Our design is based on the following result which may become useful for approximating maximum weighted independent sets in an expected sense in several different contexts, and therefore constitutes a contribution of the paper in its own right. We show that for geometric graphs, given any $\epsilon > 0$ and any allocation of non-negative weights, the expected weight of the maximum weighted independent set in a randomly selected partition approximates the overall maximum weighted independent set within a factor of $1-\epsilon$ for appropriate random selection strategies, and the maximum weighted independent set in any such partition can be computed in $O(\Delta_G^2/\epsilon^2)$ time (Appendix B). Thus, if the goal is to approximate the maximum weighted independent set in an expected sense, which incidentally suffices for approximating the maximum throughput region, the computation time need not depend on N given Δ_G, ϵ. For trees under the primary interference model, we show that the schedules that approximate the throughput regions arbitrarily closely need not approximate, even in the expected sense, the maximum weighted schedule within any guaranteed factor. Performance guarantees in this case has been attained by combining similar simple partitioning schemes with properties of trees and matchings.

Finally, recently, Jung and Shah [11], [12] obtained policies that attain order optimal expected delays in a class of graphs that includes geometric graphs with bounded node density. Using results from [11], [12], we show that many of the policies we proposed, attain the same result in a similar class of networks.

III. SYSTEM MODEL

We consider scheduling at the MAC layer in a wireless network. We assume that time is slotted and the clocks on network nodes are kept synchronized, possibly by a separate algorithm, so that there is a common notion of time among the nodes. The length of each time slot is the time required to send a packet. 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. Each node in the network has a unique ID which allows a recipient node to know the sender of a received packet. A link exists from a node u to another node v if and only if both u and v can receive each others' signals. We assume that the graph modeling the network does not change with time. 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, and $(u, v) \in E$. Note that multiple sessions may traverse a link. We consider a network with $\hat{N}$ sessions. Finally, we assume that the nodes have synchronized pseudo-random number generators so that all nodes can generate the same (random) number at a given time slot.

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

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

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 no two links in the subset interfere with each other. Let $\mathcal{X}$ be the collection of independent sets of links. The *interference graph* $I^{\mathcal{N}} = (V_I^{\mathcal{N}}, E_I^{\mathcal{N}})$ of a network $\mathcal{N}$ is an undirected graph in which the vertex set $V_I^{\mathcal{N}}$ corresponds to the set of links and there is an edge between two vertices i and j if either i interferes with j or j interferes with i. The *distance* between links l_1 and l_2 is the distance between the corresponding nodes in the interference graph of the network, and a k*-hop neighborhood* of a link l is the set of links whose distance from l is at most k.

We now describe the data packet[1] 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 subset of sessions can transmit packets in a slot if no two sessions 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. A *scheduling policy* is an algorithm that decides in each slot the subset of sessions that would transmit packets in the slot.

Let $\hat{D}_i(t)$ be the number of packets that session i transmits in slot t, $i = 1, \ldots, \ldots, \hat{N}$. $\hat{D}_i(t) \in \{0,1\}$ 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 $\hat{Q}_i(t+1) = \hat{Q}_i(t) + \hat{A}_i(t) - \hat{D}_i(t)$.

Let $DL_i(t)$ be the *delay,* or the number of slots that elapsed between the arrival and transmission of the tth arriving packet in the queue of session j. Thus, the *expected delay* for session i is $\lim_{T \to \infty} \sum_{t=1}^{T} DL_i(t)/T$. The expected delays for the sessions depend on the scheduling policy.

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

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

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 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. Also, $Q_l(t+1) = Q_l(t) + A_l(t) - D_l(t)$, and (1) holds if and only if $\limsup_{T \to \infty} \sum_{t=0}^{T-1} Q_i(t)/T$ 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 $\vec{\lambda} \in \Lambda$, $(1-\epsilon)\vec{\lambda} \in \Lambda^\pi$.

We assume that a link l knows the instantaneous virtual queue length of any other link l' only when l' communicates it to l. Also, depending on the scheduling policy, l may or may not be able to determine whether l' is scheduled in a slot if it only knows the queue length of l' in the slot, and in the latter case l knows the scheduling decision for l' only when l' communicates it to l'. A scheduling policy is said to be in INFORMATION(k) class if each link l can decide[2] whether to schedule itself once it knows the queue lengths and the scheduling decisions of a subset of the links in its k-hop neighborhood; the subset depends on l's k-hop neighborhood and the policy, and may be different for queue lengths and scheduling decisions. Finally, each link may know limited information about the entire topology; the amount of this information will depend on the specific policy and does not determine the INFORMATION class the policy is in. For a few representative policies, we will specify the information each link knows about the topology.

We now relate our assumptions to those in related papers. The assumption that the graph G does not change with time has been motivated by the fact that queue-length evolution is much faster than topological changes. This assumption is consistent with several papers in this genre (e.g., [5], [16], [17], [25], [27], [28]). Note that if the topology changes in the same time scale as queue lengths, the throughput region must be defined for the case where the graph G itself is random and sampled freshly in every slot; approximating the throughput region of such graphs is an interesting topic for future research. The assumption that each node has a unique identity may be too restrictive in some cases (such as sensor networks), but in networks where packets must be directed to specific destinations (as in an ad hoc network), such unique identities are necessary. The assumption that the time is slotted and the clocks on network nodes are kept synchronized is justified when clock drifts are negligible at the time scale of control packet transmission; similar assumptions have been made in several papers in this genre (e.g., [5], [16], [17], [25], [27], [28]). Clock synchronization, however, is a challenging problem and an area of active research; addressing the relevant issues is beyond the scope of this paper. When the above assumptions hold, using one time set up schemes or periodic set up schemes (in time scales of topological changes), each node can obtain necessary information about the topology, node identities, and can ensure that the random number generators have the same seed. This justifies the assumption that the pseudo-random number generators of all

[1]Henceforth, unless otherwise stated, a packet will refer to a data packet.

[2]In an actual implementation, one of the end nodes of a link will determine whether the link is scheduled, and for an INFORMATION(k) policy it can arrive at this decision once it knows the queue lengths and the scheduling decisions of a subset of the k-hop neighborhood of the link.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

nodes are synchronized. Also, in the time scale of queue length evolutions, only the queue lengths and the scheduling decisions need to be communicated among the links. This motivates our notion of INFORMATION(k) policies[3]. Finally, interference relations between different links need not always be pairwise in practice, e.g., transmission in a link may be successful only when the signal to interference ratio exceeds a threshold, which may for example allow pairs of neighboring links to transmit simultaneously but not three neighboring links, etc. Nevertheless, pairwise interference relations capture several important transmission scenarios, and the well-investigated protocol interference model [7] is a special case of pairwise interference relations.

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. Tree based topologies have been proposed and investigated for several resource allocation problems in multihop wireless networks, e.g., [1], [15], [22], [23]. Under the primary interference model, two links interfere if and only if they have a common node. 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 where each node has a single transceiver and a unique channel (frequency or code) in its neighborhood, e.g., Bluetooth networks, cognitive radio networks, and has been considered in several related papers [5], [16], [17], [25], [27].

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 each schedule.

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 (Fig. 1(a)).

[3]Note that distributed or local information based policies can be defined in several ways. The strongest definition is that which characterizes a policy as distributed only when the policy can be implemented without any entity having any information about the global topology [5]. To the best of our knowledge, no policy that attains guaranteed fractions of the throughput region fulfills this condition. A somewhat weaker definition requires that the policy can be implemented in networks where nodes do not have unique identities. The policies proposed in [5], [16] are distributed under this notion. The weakest notion is that which requires the nodes (or links) to base their decision on information received from their neighbors. By using broadcasts, any policy can be made distributed under this notion, and designing such a policy is trivial. The notion of INFORMATION(k) that we put forth is intermediate between the above extremes and differs from all of the above notions in that it (a) distinguishes between the attributes (e.g., queue lengths) that change fast and those (e.g., topology) that change relatively slowly and (b) parameterizes the set of nodes a node can communicate with while determining the transmission decisions.

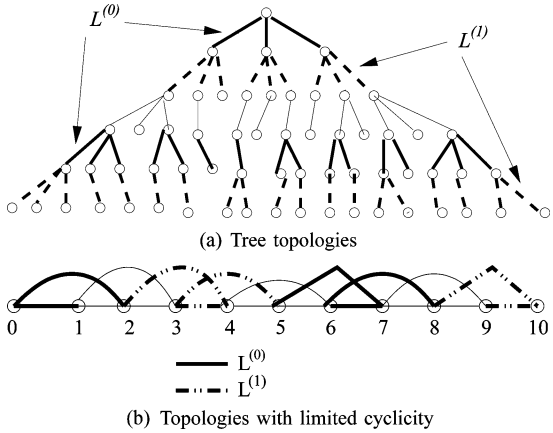


Fig. 1. The figures demonstrate the edge sets $L^{(0)}, L^{(1)}$ under the primary interference model for (a) a tree and (b) topology with limited cyclicity. In (a), $k = 3$. In (b), $k = 2$, $H = 3$, and the numbers identify the nodes, e.g., 1 is node 1. The spanning tree T we consider consists of links $(i, i+1)$ for $i = 0, \ldots, 9$, and the level of node u in T is u.

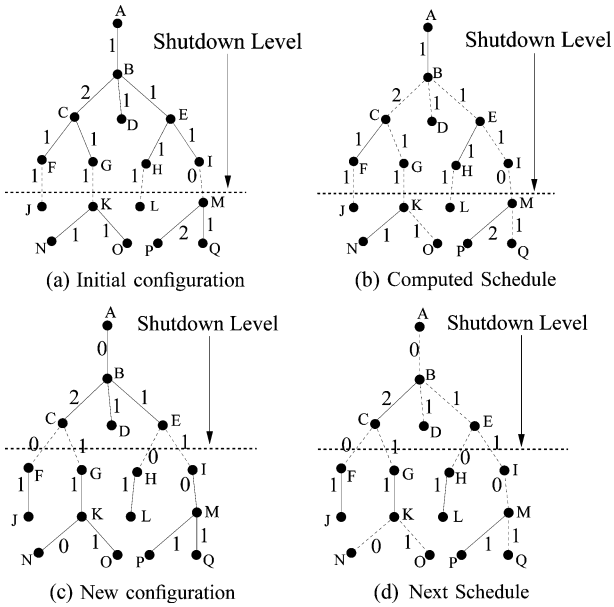
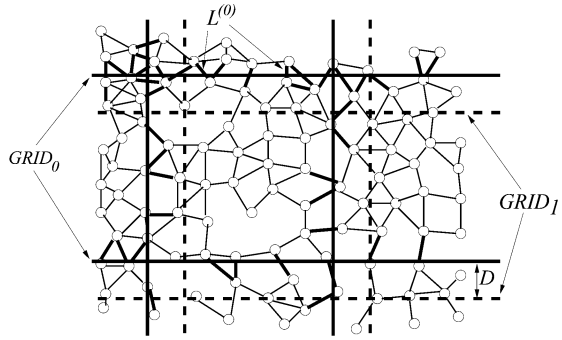


Fig. 2. The figures illustrate the operation of TPM(6) in an example tree. Fig. 2(a)shows the initial configuration in time slot t. The number on each link denotes the number of packets waiting on that link, and between any two sibling links, the one towards the left is the older sibling. Let the random number $i(t)$ selected by the links be 3. The level of nodes F, G, H, I is 3. Thus, $\mathcal{L}^{(3)}$ consists of the links shown in dashed lines; these links do not contend. Thus, no parent or older sibling of links (A, B), (K, N) and (M, P) contend. Thus, these links schedule themselves first. Thus, links (B, C), (B, D), (B, E), (K, O), (M, Q) do not schedule themselves. Thus, (C, F) and (E, H) schedule themselves. The links scheduled in t are shown in solid lines in Fig. 2(b). Let no exogenous packet arrive in slot t. Fig. 2(c) shows the new number of packets waiting on each link at the beginning of slot $t+1$. Let the random number $i(t+1)$ selected by the links in $t+1$ be 2. $\mathcal{L}^{(2)}$ consists of the links shown in dashed lines in Fig. 2(c); these links do not contend in $t+1$. The links scheduled in $t+1$ are shown in solid lines in Fig. 2(d). Note that link (A, B) contends in this slot, but does not schedule itself since it does not have a packet to transmit.

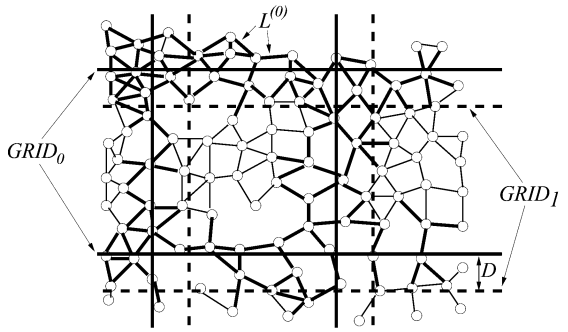
A formal description of TPM (k) is shown in Fig. 4.

TPM (k) belongs in the INFORMATION (1) class irrespective of the value k since each link needs to know the scheduling decisions of only its parent and older siblings which are within its

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.



(a) Geometric graphs under primary interference



(b) Geometric graphs under secondary interference

Fig. 3. The figures demonstrate two grids, grids $0, 1$, and the edge set $L^{(0)}$ for a geometric graph under (a) Primary and (b) Secondary interference models.

TREE-PARTITION-MATCHING (k)

In slot t,, a single integer $i(t)$ in the range $[0, \ldots, k-1]$ is randomly selected by all links (i.e., all links select the same integer). **Prob**$(i(t) = m) = 1/k$ for all $0 \leq m \leq k-1$. 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.

Fig. 4. Illustrates TPM(6) using an example.

1-hop neighborhood; no link needs to know the queue lengths, or any other function thereof, of any other link.

We now evaluate the time required for computing each schedule 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 for each schedule is $O(k\Delta_G)$.

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

The above theorem is somewhat counter-intuitive as TPM(k) does not use queue lengths of the links in the schedule computation. Thus, clearly, TPM(k) does not necessarily schedule a set of links whose sum of queue lengths is within any constant factor of the maximum possible sum of queue lengths of links in a matching. Thus, the proof cannot rely on the well-known result that a policy that schedules a set of links whose sum of queue lengths is within a factor c of the maximum possible sum of queue lengths of links in a matching, attains a throughput region which is within the factor c of the maximum throughput region [17]. We therefore first outline the idea behind the proof.

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 of each schedule 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 schedule 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, $\sum_{l' \in J_l \cup \{l\}} \lambda_{l'} < 1$ [8], [28]. Let $\delta = \min\left(1 - \max \sum_{l' \in J_l \cup \{l\}} \lambda_{l'}/2|E| \max_{l'} \lambda_{l'}, 1\right)$. Clearly, $\delta > 0$. For a link $l = (u, v)$, χ_l denotes the sum of $\Delta_G \min(\text{level}(u), \text{level}(v))$ and the number of older siblings of l. Note that $\chi_l \leq \chi_{l'} - 1$ if $l \in J_{l'}$. This is because if $l \in J_{l'}$ l is either an older sibling of l' or the parent of l'. in the first case, the end nodes of l and l' have the same levels, and l has fewer older siblings as compared to l'. In the second case, the level of the source (end) node of l is 1 less than that of the source (end) node of l', and l may have at most $\Delta_G - 1$ more older siblings than 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 asymmetricity 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, $\mathbf{E}(Q_l(t)) < \infty$ for each l under the steady state distribution for the above Markov chain. Thus,

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

$\lim_{K\longrightarrow\infty}\sum_{t=0}^{K-1}Q_l(t)/K<\infty$. The result follows:

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

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

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

The result follows. ∎

Thus, TPM ($\lceil 1/\epsilon\rceil$) approximates the maximum throughput region within a factor of $1-\epsilon$ and computes each schedule in $O(\Delta_G/\epsilon)$.

Finally, we describe one specific implementation for TPM(k) for any k. Since G is a tree, it has a root. For any node u other than the root, there exists only one node $p(u)$, denoted as the parent of u, such that (a) there is a link between u and $p(u)$ and (b) the level of $p(u)$ is less than that of u. If there exists a link between u and v and v is not the parent of u, then v is a child of u. For each link, one end node is the parent of the other—the parent node is referred to as the source node. Let the set of links for which u is the source node be C_u and the set of links for which u is an end node be E_u. Note that $E_u=C_u\cup\{(p(u),u)\}$, and the links in C_u are siblings. For example, in Fig. 2(a), $A=p(B)$, $C_B=\{(B,C),\ (B,D),\ (B,E)\}$,, $E_B=\{(A,B),\ (B,C),\ (B,D),\ (B,E)\}$.

We assume that each node u knows its level, its parent and children nodes in the tree and the ordering among the links in C_u. The source node of a link decides whether to schedule the link. Consider a node u in G. In each slot, either $(p(u),u)$, or links in C_u, or all links in E_u contend; u decides which is the case as per the first step of TPM(k). (a) If links in C_u do not contend in t, then u takes no scheduling decision. (b) If $(p(u),u))$ does not contend in t, u schedules the oldest sibling in C_u that has a packet to transmit, and decides that the rest of the links in C_u will not be scheduled in t. (c) If all links in E_u contend in t, u waits for $p(u)$ to inform it about whether $(p(u),u)$ is scheduled in the slot (note that $p(u)$ decides whether to schedule $(p(u),u)$). If $(p(u),u)$ is scheduled in the slot, u decides that none of the links in C_u will be scheduled in the slot; else, u schedules the siblings in C_u as in the case that $(p(u),u))$ does not contend in t. In cases (b) and (c), u informs each of its children about the scheduling decision for the link between it and the child node.

Clearly, TPM(k) is simple to implement. Also, during the computation time of a schedule, each node performs no computation, is involved in at most Δ_G communications (a node transmits 1 bit, or rather 1 packet of minimum possible size, to each of its children, and receives at most 1 bit), and waits for the rest of the time. Clearly, in any scheduling policy that avoids collisions during packet transmissions, in the worst case each node needs to communicate at least once with each of its neighbors. Thus, among the policies that avoid collisions, TPM(k) minimizes the communications and computations for each node.

V. Information (k) Policies for Approximating the Maximum Throughput Region Arbitrarily Closely

We first provide a general framework for approximating the maximum throughput region arbitrarily closely using policies in Information (k) class (Section V-A). Then we use this framework to obtain arbitrary tradeoffs between throughput approximations and schedule computation times in several important classes of networks and interference models (Sections V-B–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 that computes each schedule in $O\left(\Delta_G^2/\epsilon^2\right)$ time (Section V-C). These results can be extended to arbitrary graphs with limited cyclicity (Section V-B) and quasi-geometric graphs (Section V-D). We upper bound the expected delays attained by these policies and prove that the bounds are comparable to the best known guarantees in these networks (Section V-E).

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

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:

Let $S_l=\{l' : l'=l$ or l' interferes with l or l 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 $X_i\subseteq E_i$

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

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,

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

The intuition behind the result is as follows. 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. If $L^{(0)},\ldots,L^{(k-1)}$ are $c-$ approximate, then the expected weight of the maximum weight independent set in $L^{(i(t))}$ 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. The arguments in this proof can be generalized to obtain an expected sense approximation for maximum weighted independent sets in geometric graphs using a computation time that depends only on the approximation factor and the degree of the graph (Lemma 10); we state and prove this general result in Appendix B.

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

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

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

Lemma 2: Let $L^{(0)},\ldots,L^{(k-1)}$ be c-approximate. Then, if $\vec{\lambda}\in\text{Int}(\Lambda)$ and $k\ge c>0$, $(1-c/k)\vec{\lambda}\in\Lambda^{\pi(k)}$.

We 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 the weight of the maximum weight independent set of links in the slot. The throughput guarantee now follows using Lyapunov arguments similar to those in [17], [28]. We prove this lemma towards the end of this subsection.

Once we prove that the collection $L^{(0)},\ldots,L^{(k-1)}$ is c-approximate for some c, irrespective of the value of c, Lemma 2 allows us to approximate the maximum throughput region within a factor of $1-\epsilon$ for any $\epsilon>0$ using $\pi(k)$ for $k=\lceil c/\epsilon\rceil$. Then the network designer simply chooses an appropriate ϵ based on the desired trade-off between performance and computational burden (the smaller the ϵ, the better the approximation of the optimal capacity region, but the higher the computational burden) and the corresponding k guarantees the desired throughput. 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 (lemmas 4, 6, 7). The value of c may however be different for different interference models and network topologies, and the constants in the expressions for the schedule computation times will typically increase with increase in 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 $j\in\{0,\ldots,k-1\}$ [4] 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 each

[4]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.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

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

Let $\vec{I}^X$ denote the indicator vector for set $X \in \mathcal{X}$. Note that $\phi \in \mathcal{X}$. Then, Int(Λ) can be characterized as follows [28]:

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

$$\text{Now, } \mathbf{E}\left(\left(\vec{A}(t)\right)^T \vec{Q}(t) | \vec{Q}(t) = \vec{Q}\right) = \left(\frac{1-c}{k}\right) \vec{\lambda}^T \vec{Q} \leq \left(\frac{1-c}{k}\right)(1-\beta_\phi) \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i. \quad (4)$$

The inequality follows by using $\vec{\lambda} = \sum_{\vec{X} \in \mathcal{X}} \beta_X \vec{I}^X$, $\sum_{i \in X} Q_i \leq \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i$ and $\sum_{X \in \mathcal{X} \setminus \{\phi\}} \beta_X = 1 - \beta_\phi$.

Since $V\left(\vec{Q}(t+1)\right) - V\left(\vec{Q}(t)\right) = \left(\vec{Q}(t+1) - \vec{Q}(t)\right)^T \left(\vec{Q}(t+1) + \vec{Q}(t)\right)$, $\vec{Q}(t+1) = \vec{Q}(t) + \vec{A}(t) - \vec{D}(t)$ and $\left(\vec{A}(t) - \vec{D}(t)\right)^T \left(\vec{A}(t) - \vec{D}(t)\right) \leq N\alpha_{\max}^2$, we obtain

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

■

Finally, we present a lemma that we will use in analyzing the expected delay of the policies we develop in this section. Recall that $\hat{\lambda}_j$ is the arrival rate for session j.

Lemma 3: Let the arrival rate vector be $(1-\epsilon')(1-c/k)\vec{\lambda}$ where $0 < \epsilon' < 1$ and $\vec{\lambda} \in \text{Int}(\Lambda)$. Let $L^{(0)}, \ldots, L^{(k-1)}$ be c-approximate. Then under $\pi(k)$ $\lim_{T\to\infty} \sum_{t=1}^{T} \sum_{j=1}^{\hat{N}} DL_j(t)/T \leq N\alpha_{\max}^2 \max_{l \in E} |S_l| / 2(1-c/k)\epsilon' \min_k \hat{\lambda}_k$.

The proof uses techniques for bounding first moments developed in [14], which have subsequently been extensively used in different contexts, e.g., [11], [19].

Proof: Let the arrival rate vector be $(1-\epsilon')(1-c/k)\vec{\lambda}$ where $0 < \epsilon' < 1$ and $\pi(k)$ be used. Since $\vec{\lambda} \in \text{Int}(\Lambda)$, $(1-\epsilon')\vec{\lambda} \in \text{Int}(\Lambda)$. Thus, since $L^{(0)}, \ldots, L^{(k-1)}$ is c-approximate, the Proof of Lemma 2 shows that the Markov chain representing the queue length process is positive recurrent. Thus, $\lim_{T\to\infty} \sum_{t=1}^{T} \sum_{j=1}^{\hat{N}} DL_j(t)/T$, $\lim_{T\to\infty} \sum_{t=1}^{T} \mathbf{E}\left(\sum_{i \in E} Q_i(t)\right)/T$ and $\lim_{T\to\infty} \sum_{t=1}^{T} \mathbf{E}\left(\max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)\right)/T$ exist. Also, using little's law, and the strong law of large numbers for i.i.d. arrivals, $\lim_{T\to\infty} \sum_{t=1}^{T} \sum_{j=1}^{\hat{N}} DL_j(t)/T \leq 1/\min_k \hat{\lambda}_k \lim_{T\to\infty} \sum_{t=1}^{T} \mathbf{E}\left(\sum_{i \in E} Q_i(t)\right)/T$. We now show that $\sum_{i \in E} Q_i(t) \leq \max_{l \in E} |S_l| \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)$ at each t. Then, the lemma follows if we can show that $\lim_{T\to\infty} \sum_{t=1}^{T} \mathbf{E}\left(\max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)\right)/T \leq N\alpha_{\max}^2/2(1-c/k)\epsilon'$.

Consider an $X'(t) \in \mathcal{X}$ which is obtained as follows. Initially, $X'(t) = E$. Now, let link l have the maximum queue length at t among the links in $X'(t)$. Then all links in $S_l \setminus \{l\}$ are removed from $X'(t)$. The process is repeated until $X'(t) \in \mathcal{X}$. Note that $\sum_{i \in \mathcal{C}} Q_i(t) \leq \max_{l \in E} |S_l| \sum_{i \in X'(t)} Q_i(t)$. Now, since $\sum_{i \in X'(t)} Q_i(t) \leq \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)$, $\sum_{i \in E} Q_i(t) \leq \max_{l \in E} |S_l| \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)$.

We now show that $\lim_{T\to\infty} \sum_{t=1}^{T} \mathbf{E}\left(\max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)\right)/T \leq N\alpha_{\max}^2/2(1-c/k)\epsilon'$. Similar to the deduction of (4), we can show that $\mathbf{E}\left(\left(\vec{A}(t)\right)^T \vec{Q}(t) | \vec{Q}(t) = \vec{Q}\right) \leq (1-c/k)(1-\epsilon') \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i$. Let $V(\vec{Q}) = \sum_i Q_i^2$. Thus, from lemma 1 and as in the proof of lemma 2, we can show that

$$\begin{aligned}
&\mathbf{E}\left(V\left(\vec{Q}(t+1)\right) - V\left(\vec{Q}(t)\right) | \vec{Q}(t) = \vec{Q}\right) \\
&\leq -2\left(\frac{1-c}{k}\right)\epsilon' \max_{X \in \mathcal{X}} \sum_{i \in X} Q_i + N\alpha_{\max}^2 \ \forall\, \vec{Q}. \\
\text{Thus, } &\mathbf{E}\left(V\left(\vec{Q}(t+1)\right) - V\left(\vec{Q}(t)\right)\right) \\
&\leq -2\left(\frac{1-c}{k}\right)\epsilon' \mathbf{E}\left(\max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)\right) + N\alpha_{\max}^2 \ \forall\, t. \\
\text{Hence, } &\mathbf{E}\left(V\left(\vec{Q}(T+1)\right) - V\left(\vec{Q}(1)\right)\right) \\
&\leq -2\left(\frac{1-c}{k}\right)\epsilon' \sum_{t=1}^{T} \mathbf{E}\left(\max_{X \in \mathcal{X}} \sum_{i \in X} Q_i(t)\right) + NT\alpha_{\max}^2.
\end{aligned}$$

The last inequality follows using a telescopic sum. The result follows since $V(\vec{Q}(t)) \geq 0$ for all t and $\lim_{T\to\infty} \mathbf{E}\left(V\left(\vec{Q}(1)\right)\right)/T = 0$ if the initial queue lengths are bounded. ■

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

B. Graphs With Limited Cyclicity

Using the above general framework, we generalize the tradeoff between throughput and the time required to compute each schedule 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.

The sets $L^{(0)},\ldots,L^{(k-1)}$ for the scheduling policy, referred to as H-LIMITED-CYCLICITY-PARTITION-MATCHING (k) and abbreviated as H-LCPM (k), are as follows. 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 (Fig. 1(b)). Intuitively, for H-LIMITED-CYCLICITY-PARTITION-MATCHING (k), when $i(t)=j$, levels jH, $jH+kH$, $jH+2$ hops, $\ldots$ 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 time H-LCPM(k) needs to compute each schedule. 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 $T^{(j)}$. Consider all nodes that are in a given component of $G^{(j)}$, but are in different components of $T^{(j)}$. These nodes have a common ancestor, say v, in T. The subtree of T with v as the root and the above nodes as leaves has diameter at most $H-1$. Thus, the number of leaves of this tree is at most Δ_G^H. Hence, 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 of each schedule is $O(\Delta_G^{3(k+1)H})$. If G is a bipartite graph, the overall computation time of each schedule 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 4: $L^{(0)},\ldots,L^{(k-1)}$ is 6- 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 $\sum_{j=0}^{k-1}\sum_{l\in X_j}W_l\le 6\sum_{l\in X^*}W_l$. For any link l, $W_l\le\sum_{i\in X^*\cap S_l}W_i$. Let $\eta_l^{(j)}=|X_j\cap S_l|$. Thus, $\sum_{l\in X_j}W_l\le\sum_{l\in X_j}\sum_{i\in X^*\cap S_l}W_i=\sum_{l\in X^*}\eta_l^{(j)}W_l$.

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

Hence, we need to show that $\left(\sum_{j=0}^{k-1}\eta_l^{(j)}\right)\le 6$ for each $l\in X^*$.

Consider $l=(u,v)\in E$, and let u be the parent of v in T. There exists a unique j_l such that level of u in T is in $((j_l-1)H,j_lH]\bmod kH$. Note that l is not adjacent to any link in $L^{(q)}$ where $q<(j_l-1)\bmod k$ or $q>(j_l+1)\bmod k$, i.e., $\eta_l^{(q)}=0$ for the above q. Since X_j s are matchings, at most 2 links in X_j is adjacent to l for any j, i.e., $\eta_l^{(j)}\le 2$ for any j. Thus, $\left(\sum_{j=0}^{k-1}\eta_l^{(j)}\right)\le 6$ for each $l\in X^*$. ■

Theorem 2: If $\vec{\lambda}\in\text{Int}(\Lambda)$ and $\epsilon\in(0,1)$, then $(1-\epsilon)\vec{\lambda}\in\Lambda^{\text{H-LCPM}(\lceil 6/\epsilon\rceil)}$.

Using $k=\lceil 6/\epsilon\rceil$, $c=6$, Theorem 2 follows from Lemmas 4 and 2. Now, H-LCPM $(\lceil 6/\epsilon\rceil)$ is in INFORMATION $\left(O(H\Delta_G^H/\epsilon)\right)$ class and requires $\Delta_G^{O(H/\epsilon)}$ time to compute each schedule. Thus, H-LCPM will be useful for small values of H.

Finally, note that under the primary interference model, $\max_l|S_l|\le 2\Delta_G+1$. Now, consider any $\epsilon\in(0,1)$, $\epsilon'\in(0,1)$, $\vec{\lambda}\in\text{Int}(\Lambda)$. Using $k=\lceil 6/\epsilon\rceil$, $c=6$, it follows from lemmas 3 and 4 that when the arrival rate vector is $(1-\epsilon)(1-\epsilon')\vec{\lambda}$ and H-LCPM $(\lceil 6/\epsilon\rceil)$ is used, the sum of the expected delays of the sessions $\lim_{T\to\infty}\sum_{t=1}^{T}\sum_{j=1}^{\hat{N}}DL_j(t)/T$ is at most $N\alpha_{\max}^2(2\Delta_G+1)/2(1-\epsilon)\epsilon'\min_k\hat{\lambda}_k$. In other words, for an arrival rate vector in the throughput region of H-LCPM $(\lceil 6/\epsilon\rceil)$, $\text{Int}((1-\epsilon)\Lambda)$, the above sum is upper bounded by a quantity that depends on the arrival process (through $\alpha_{\max}$, $\min_k\hat{\lambda}_k$, and the parameter ϵ' that determines the distance of the rate vector from the boundary of the throughput region of H-LCPM $(\lceil 6/\epsilon\rceil)$), network (through N,Δ_G) and the policy parameter (through ϵ).

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. Geometric graphs have been extensively investigated in several different contexts in wireless networks (e.g., [2], [24]). We consider both the primary interference model (Section V-C-I) and the secondary interference model (Section V-C-II).

1) Geometric Graphs With Primary Interference Model: We consider a geometric graph G with primary interference model. The sets $L^{(0)},\ldots,L^{(k-1)}$ for the policy GEOMETRIC-GRAPH-PARTITION-MATCHING (k), which we abbreviate as GGPM (k), are as follows. 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 (Fig. 3(a)). Note that the links in a component of $G^{(j)}$ do not interfere with those in other components.

We first evaluate the time for computing each schedule for GGPM (k). The overall computation time for each schedule is the worst case computation time in a component. Let ν be the maximum number of nodes in any component of $G^{(j)}=(V,E\setminus L^{(j)})$ for any j. We show that ν is $O(\Delta_G k^2)$. Thus,

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

the time for computing each schedule is the time for computing 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 $\left(O(\Delta_G k^2)\right)$ class.

Lemma 5: 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 6: $L^{(0)}, \ldots, L^{(k-1)}$ is 12-approximate.

Proof: The proof is similar to that for Lemma 4. 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 $l \in X^*$.

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 $(1-\epsilon)\vec{\lambda} \in \Lambda^{\text{GGPM}(\lceil 12/\epsilon \rceil)}$.

Using $k = \lceil 12/\epsilon \rceil$, $c = 12$, Theorem 3 follows from lemmas 2 and 6. GGPM $(\lceil 12/\epsilon \rceil)$ is in INFORMATION $\left(O(\Delta_G/\epsilon^2)\right)$ class and computes each schedule in $O(\Delta_G^3/\epsilon^6)$ 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 $(1-\epsilon)$.

Finally, we upper bound the expected delays of the sessions. Now, consider any $\epsilon \in (0,1)$, $\epsilon' \in (0,1)$, $\vec{\lambda} \in \text{Int}(\Lambda)$. Using $k = \lceil 12/\epsilon \rceil$, $c = 12$, and since under the primary interference model, $\max_l |S_l| \leq 2\Delta_G + 1$, it follows from lemmas 3 and 6 that when the arrival rate vector is $(1-\epsilon)(1-\epsilon')\vec{\lambda}$ and GGPM $(\lceil 12/\epsilon \rceil)$ is used, the sum of the expected delays of the sessions $\lim_{T\to\infty} \sum_{t=1}^{T} \sum_{j=1}^{\hat{N}} DL_j(t)/T$ is at most $N\alpha_{\max}^2(2\Delta_G+1)/2(1-\epsilon)\epsilon' \min_k \hat{\lambda}_k$.

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 $L^{(0)}, \ldots, L^{(k-1)}$ for the policy GEOMETRIC-GRAPH-PARTITION-INDEPENDENT-SET (k) which we abbreviate as GGPIS (k). Just as in Section V-C-I, 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 (Fig. 3(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 each schedule for GGPIS (k). From lemma 5, 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 5 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, each schedule can be computed is $(\Delta_G^2 k^2)^{O(k^2)}$ time. Again, like GGPM (k), GGPIS (k) is in INFORMATION $\left(O(\Delta_G k^2)\right)$ class.

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

- (*Observation* 1) Let $\psi_l = \{j : l \in L^{(j)}\}$. Then, $|\psi_l| \leq 6$ for any $l \in E$. 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.
- (*Observation* 2) For any l, $l \notin L^{(j)}$ but l interferes with a link in $L^{(j)}$ for at most 8 js in $\{0, \ldots, k-1\}$. This happens only if one of the end nodes of l is within $2D$ units of a horizontal or vertical line of grid j. This can happen at most 4 times for vertical lines and 4 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 7: $L^{(0)}, \ldots, L^{(k-1)}$ is 119-approximate.

Proof: The proof is similar to that for Lemma 6. Like in Lemma 6, 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 14 js in $\{0, \ldots, K-1\}$. Hence, $\left(\sum_{j=0}^{k-1} \eta_l^{(j)}\right) \leq 8 \times 14 = 112$ for each $l \in X^*$. ■

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

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 $k = \lceil 112/\epsilon \rceil$, $c = 112$, Theorem 4 follows from Lemmas 2 and 7. GGPIS $(\lceil 112/\epsilon \rceil)$ is in INFORMATION $\left(O(\Delta_G/\epsilon^2)\right)$ class and computes each schedule in $(\Delta_G/\epsilon)^{O(1/\epsilon^2)}$ time. Now, consider any $\epsilon \in (0,1)$, $\epsilon' \in (0,1)$, $\vec{\lambda} \in \text{Int}(\Lambda)$. Note that under the secondary interference model, $\max_l |S_l| \leq 2\Delta_G^2 + 1$. Using $k = \lceil 112/\epsilon \rceil$, $c = 112$, it follows from lemmas 3 and 7 that when the arrival rate vector is $(1-\epsilon)(1-\epsilon')\vec{\lambda}$ and GGPIS $(\lceil 112/\epsilon \rceil)$ is used, the sum of the expected delays of the sessions $\lim_{T\to\infty} \sum_{t=1}^{T} \sum_{j=1}^{\hat{N}} DL_j(t)/T$ is at most $N\alpha_{\max}^2(2\Delta_G^2+1)/2(1-\epsilon)\epsilon' \min_k \hat{\lambda}_k$.

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 GGPIS $(\lceil 112/\epsilon \rceil)$ but computes each schedule in only $O(\Delta_G^2/\epsilon^2)$ time. 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 [27] 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 reduces the time required to compute each schedule from $\Theta(N)$ attained using only the improvement based selection schemes in [27] 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. Independent sets can be determined in each component in $O\left(\Delta_G \log(\Delta_G k)\right)$ time using existing randomized algorithms [18], [21]; such algorithms select the maximum weighted independent set in each component with a positive probability. 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 time required by GGPGI (k) to compute each schedule 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 $\left(O(\Delta_G k^2)\right)$ class.

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

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 [28]

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

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 $\vec{\lambda'} \in \text{Int}(\hat{\Lambda}^{(j)})$ [5], [27].

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

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

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

$k(1/(k-6))(1-6/k)\lambda_l = \lambda_l$. Thus, the arrival rate vector for these secondary virtual queues is $\vec{\lambda}^{(j)} \in \text{Int}(\Lambda^{(j)})$. 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 $\left(O(\Delta_G^2/\epsilon^2)\right)$ class, approximates the maximum throughput region within a factor of $1-\epsilon$ and computes each schedule in $O(\Delta_G^2/\epsilon^2)$ time. Note that GGPM (k) can be similarly modified to attain the same throughput guarantee using $k = \lceil 4/\epsilon \rceil$ and computing each schedule in $O(\Delta_G^2/\epsilon^2)$ time. 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 $1-\epsilon$ while computing each schedule in $O\left(f(r)\Delta_G^2/\epsilon\right)$ where $f(r)$ increases with increase in r.

We now sketch one possible implementation of GGPGI (k), with the goal of elucidating the information each node maintains about the topology and analyzing the control message exchange complexity in the time scale of packet transmission. Owing to space limitations, we omit several details. We assume that each node knows the grids for which it is within distance D of a vertical or horizontal line. Note that a node can determine this if it knows its location, or it can be informed of this when the network is initialized. Any end node of a link can now determine the set of virtual queues associated with the link, and the slots in which each such virtual queue contends, which in turn determines which slots of the frame the link contends. Note that a link always contends in the same slots of every frame as this does not depend on its queue length. During network initialization, a forest $F^{(j)}$ spanning the links that contend in the jth slot of each frame is established for each j (depth first search or breadth first search or their variants can be used to determine such forests). Again tree traversal policies can be used to inform each node of its parent and children in each such forest. The resulting control message exchange occurs in the time scale of topology evolution, and not in the time scale of packet transmission.

Now, consider the decisions and the control message exchanges in the time scale of packet transmission (i.e., the control messages that are exchanged for determining each schedule). For each j, each node stores the total weight of the links scheduled in its component in the jth slot of the previous frame, and which, if any, of its incident links were scheduled in the jth slot of the previous frame (we explain how a node determines these quantities). Consider the jth slot of each frame. An existing randomized policy can be used for determining an independent set among the links that contend in $O\left(\Delta_G \log(\Delta_G k)\right)$ time [18], [21]; each node exchanges $O\left(\Delta_G \log(\Delta_G k)\right)$ messages during this procedure. Such randomized policies requires each node to only know which of its incident links are contending in a slot, and at the end of the procedure each node knows which, if any, of its incident links are selected in the independent set. Each node computes the sum of the weights of its incident links that have been selected in the independent set. The root of each tree in the forest $F^{(j)}$ initiates a message where it inserts the number it computed, and as the message propagates through the tree, each node adds the sum it computed with the number in the message. The message is returned to the root after it finishes traversing the entire tree. When the message returns to the root, it contains twice the total weight of the newly selected independent set in the component spanned by the tree. The root broadcasts the message again in the tree, which informs each node of the weight of the links in the newly selected independent set in the component. Using this weight, each node can now determine whether the newly selected independent set should be scheduled, or the schedule used in the jth slot of the previous frame should be used, and accordingly updates the weight it stores and the identities of the incident links scheduled. Each node thereby knows whether any of its incident links belong to the scheduled independent set, and participates in the transmission accordingly.

Clearly, each node exchanges $O\left(\Delta_G \log(\Delta_G k)\right)$ control messages for computing each schedule. The computation time and the information class of this implementation are as discussed for the policy. The above implementation is clearly a naive one, and can be optimized in several different ways, e.g., using gossip algorithms as in [5], which constitutes interesting directions for future research. The policies proposed in the previous subsections can be implemented similarly.

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, depending on propagation conditions, receiver sensitivity, antenna orientations, etc., 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. Quasi-geometric graphs generalize the notion of geometric graphs, and become geometric when $\iota = 1$ and can approximate arbitrary graphs, as long as the nodes are embedded in a plane, for small ι and large D (as in this case, (b) applies for most edges and thus the existence of the edges do not depend on the distance between the nodes). But, as we discuss next, the schedule computation times for the proposed policies becomes large as ι becomes small.

Under primary interference model, as before, two links interfere if and only if they are adjacent. Under secondary interference model, 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, GGPGI (k) computes each schedule in $O(\Delta_G^2 k^2/\iota^2)$ time. Also, GGPGI (k) is in INFORMATION $\left(O(\Delta_G k^2/\iota^2)\right)$ class. Thus, GGPGI $(\lceil 6/\epsilon \rceil)$ approximates the maximum throughput region within a factor of $1-\epsilon$ while computing each schedule in $O\left(\Delta_G^2/(\iota^2\epsilon^2)\right)$ time and is in INFORMATION $\left(O\left(\Delta_G/\left(\iota^2\epsilon^2\right)\right)\right)$ class. Similarly, under the primary interference model, a throughput region of $1-\epsilon$ of the maximum throughput region can be attained

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

using a policy in INFORMATION $\left(O(\Delta_G/\left(\iota^2\epsilon^2\right))\right)$ class which computes each schedule in $O(\Delta_G^2/(\epsilon^2\iota^2))$ time.

E. Delay Guarantees

Characterizing the tradeoffs between schedule computation time and other performance attributes such as packet loss in networks where nodes have finite buffers, and delay constitute interesting directions for future research. In fact, characterization of policies that minimize the expected packet loss in networks where nodes have finite buffers, and delay remain open as well. Recently, Jung and Shah [11], [12] obtained policies that attain order optimal expected delays in a class of graphs that includes geometric graphs with bounded node density[5]. We now show that the policies we propose attain the same result as well. The delay guarantees provided after Theorems 2,3,3 show that for constant $\epsilon, \epsilon', \min_k \hat{\lambda}_k, \alpha_{\max}, \Delta_G$, the sum of the expected delays of the sessions $\lim_{T\to\infty} \sum_{t=1}^{T} \sum_{j=1}^{\hat{N}} DL_j(t)/T$ for H-LCPM, GGPM and GGPIS is $O(N)$. Thus, since $\hat{N} \leq N$, the expected delay per session is $O(1)$ for these policies. Jung and Shah [11], [12] provided an example which showed that if the network satisfies certain characteristics there exists arrival rate vectors such that the sum of the expected delays of the sessions is $\Omega(N)$ (Theorem 5 [12]). Networks with primary and secondary interference and bounded degree Δ_G satisfy the properties needed to construct the above example. Thus, the sum of the expected delays in networks that satisfy the above characteristic is $\Omega(N)$. Thus, H-LCPM, GGPM and GGPIS attain order optimal expected delays in their respective topologies provided the network degrees are bounded. The degrees are for example bounded in geometric graphs under primary and secondary interference constraints when the node density is bounded.

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 schedule 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. Thus, the queue-length and departure vectors, $\vec{Q}(t), \vec{D}(t)$, are M -dimensional vectors 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 $\pi^{\mathrm{MH}}(k)$, differs from $\pi(k)$, in only the assignment of link weights. For $\pi^{\mathrm{MH}}(k)$ 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 $\pi^{\mathrm{MH}}(k)$ and $\pi(k)$ are otherwise the same.

Lemma 8: Let $L^{(0)}, \ldots, L^{(k-1)}$ 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 8 in appendix.

We now consider the throughput guarantees of π^{MH} for different classes of networks considered in Sections V-B to V-D. The choice of $L^{(0)}, \ldots, L^{(k-1)}$ for different classes of networks remain the same as in Sections V-B to V-D. Using $k = \lceil 4/\epsilon \rceil$, $c = 4$, Theorem V-B follows from lemmas 2 and 4 for H-LCPM $^{\mathrm{MH}}(k)$. Using $k = \lceil 12/\epsilon \rceil$, $c = 12$, Theorem 2 follows from lemmas 2 and 6 for GGPM $^{\mathrm{MH}}(k)$. Using $k = \lceil 80/\epsilon \rceil$, $c = 80$, Theorem 3 follows from lemmas 2 and 7 for GGPIS $^{\mathrm{MH}}(k)$.

Clearly, the schedule 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 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 [26], the proofs can be generalized to accommodate Markov modulated arrival processes. Also, under the weaker notion of rate stability which only ensures

[5]Node density is the number of nodes per unit area. If the number of nodes in any circle of a given radius is bounded, node density is bounded.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

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 schedule computation times for all stationary ergodic arrival processes that satisfy the strong law of large numbers. Rate stability however does not ensure that the expected queue lengths are finite which is required in many applications and which is the notion of stability we consider in this paper. Obtaining provable throughput guarantees for non-Markovian arrival processes under the notion of stability that requires that expected queue lengths be finite remains largely open. In a companion paper, we obtain a policy in INFORMATION (1) class, that approximates the maximum throughput region for non-Markovian arrival processes under the above notion of stability within a factor of 2/3 in tree topologies under primary interference model and computes each schedule in $O\left(\Delta_G(\log\Delta_G)\log N\right)$ time. The results in these papers compliment each other.

APPENDIX

Proof for Lemma 8: We first state and prove lemma 9 for $\pi^{\mathrm{MH}}(k)$ which will be useful in proving lemma 8.

Lemma 9: Let $L^{(0)},\ldots,L^{(k-1)}$ be c-approximate. Then,

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

Proof: Let $B(t)=\arg\max_{\substack{X\in\mathcal{X}\\ X\subseteq L^{(i(t))}}}\sum_{l\in X}G_l(t)$. Again, $\sum_i G_i(t)D_i(t)\geq\left(\max_{X\in\mathcal{X}}\sum_{i\in X}G_i(t)-\sum_{i\in B(t)}G_i(t)\right)$

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

The result follows. ∎

We now prove lemma 8. This proof follows from lemma 9 using techniques similar to those used by Tassiulas *et al.* in [28].

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 $\vec{\lambda}\in\mathrm{Int}(\Lambda)$. Clearly, under $\pi^{\mathrm{MH}}(k)$, $\vec{Q}(t)$ 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, $\lim_{K\to\infty}\sum_{t=0}^{K-1}Q_i(t)/K<\infty$. 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, $\mathrm{Int}(\Lambda)$ can be characterized as follows [28]:

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

$$\begin{aligned}
\text{Now,}\quad&\mathbf{E}\left(\left(\vec{A}(t)\right)^T\frac{\vec{Q}(t)}{\vec{Q}(t)}=\vec{Q}\right)\\
&=\left(\frac{1-c}{k}\right)\vec{\lambda}^T\vec{Q}\\
&\leq\left(\frac{1-c}{k}\right)(1-\beta_\phi)\max_{X\in\mathcal{X}}\sum_{i\in X}G_i\ (\text{from (6)}).
\end{aligned}\tag{7}$$

$$\begin{aligned}
&\mathbf{E}\left(V\left(\vec{Q}(t+1)\right)-V\left(\vec{Q}(t)\right)|\vec{Q}(t)=\vec{Q}\right)\\
&\leq 2\mathbf{E}\left(\vec{A}^T(t)\vec{Q(t)}|\vec{Q}(t)=\vec{Q}\right)\\
&\quad-2\mathbf{E}\left(\sum_i G_i(t)D_i(t)|\vec{Q}(t)=\vec{Q}\right)+M\hat{\alpha}_{\max}^2\\
&\leq-2\left(\frac{1-c}{k}\right)\beta_\phi\max_{X\in\mathcal{X}}\sum_{i\in X}G_i+M\hat{\alpha}_{\max}^2\\
&\quad(\text{from Lemma 9 and (7)})\\
&<-1\text{ for all sufficiently large }\|\vec{Q}\|\\
&\quad(\text{since }\beta_\phi>0,\ 0<c<k).
\end{aligned}$$

∎

Arbitrary Tradeoffs Between Computation Times and Expected Sense Approximation for Maximum Weighted Independent Sets in Geometric Graphs: We now show that the approximation techniques we use can also be used for approximating maximum weighted independent sets in an expected sense arbitrarily closely in geometric graphs using a computation time which depends only on the degree of the graph and the desired approximation factor.

Consider a graph $G=(V,E)$. Let a set of nodes be independent if there does not exist links between any two nodes in the set. Note that this is the usual notion of independence used in graphs, and is similar to the notion of independence we used for links. Let $V^{(0)}$, $V^{(1)},\ldots,V^{(k)}$ be subsets of V, and $\hat{G}^{(j)}$ be the graph obtained by removing $V^{(k)}$ from V and the links incident to nodes in $V^{(k)}$ from E.

Definition 3: Let $\hat{\mathcal{X}}$ be the collection of independent sets of G. A collection of subsets $V_1,\ldots,V_q$ of V is said to be c *-vertex-approximate* if for (a) any given $|V|$ -dimensional vector of non-negative real numbers $\vec{W}=(W_1,\ldots,W_{|V|})$ and (b) any collection of subsets of V, $X_1,\ldots X_q$ such that $X_i\in\hat{\mathcal{X}}$ and $X_i\subseteq V_i$

$$\sum_{i=1}^{q}\sum_{v\in X_i}W_v\leq c\max_{X\in\hat{\mathcal{X}}}\sum_{v\in X}W_v.$$

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

In the proof of lemma 1, we have not used any specific properties of queue lengths and departure vectors, except that (a) independent sets refer to sets of links rather than vertices (b) the queue lengths are non-negative, and (c) the departure vector $\vec{D}(t)$ is such that it constitutes a maximum weighted independent set in $G^{(j)}$ where j is selected uniformly among $[0,\ldots,k-1]$ and the weight of a link is its queue length. Thus, we have actually proved a more general result which states that the expected weight of the maximum weight independent set in $\hat{G}^{(j)}$ is greater than or equal to $(1-c/k)$ times the weight of the maximum weight independent set in G, if j is selected uniformly in $[0,\ldots,k-1]$. We state this result next. Note that in this sentence, and henceforth, the term independent set will refer to the definition introduced in this subsection.

Lemma 10: Let $V^{(0)},\ldots,V^{(k-1)}$ be c-vertex-approximate, and w_v be the weight of vertex $v \in V$ such that $w_v \geq 0$. Let j be selected uniformly among $[0,\ldots,k-1]$, and $\hat{X}^{(j)}$ be a maximum weight independent set in $\hat{G}^{(j)}$. Then, $\mathbf{E}\left(\sum_{i\in\hat{X}^{(j)}} w_i\right) \geq (1-c/k)\max_{X\in\hat{\mathcal{X}}}\sum_{i\in X} w_i$.

Consider a geometric graph as defined in the first paragraph of Section V-C. Consider the grids as described in the first paragraph of Section V-C-I. Let $V^{(j)}$ consist of all nodes that are within distance D of a vertical or a horizontal line of the jth grid. We next state and prove the following lemma.

Lemma 11: $V^{(0)},\ldots,V^{(k-1)}$ are 48-vertex-approximate.

Proof: We use the notation of Lemma 4 *mutatis mutandis*. The result follows if we show that $\left(\sum_{j=0}^{k-1}\eta_v^{(j)}\right) \leq 48$ for each $v \in X^*$. This holds since there can at most be 6 independent nodes in a given node's neighborhood, and A Node's Neighborhood May Contain A Node in $V^{(j)}$ for at most 8 different grids (for 4 vertical and 4 horizontal grid lines). ■

Now consider the following independent set selection policy. Select an integer uniformly in the range $[0,\ldots,k-1]$. If j is the selected integer, then determine the maximum weighted independent set in $\hat{G}^{(j)}$. By lemmas 10 and 11, the expected weight of this set is at least $(1-48/k)$ times that of the maximum weight of an independent set in G provided each vertex has a non-negative weight. Note that a maximum weight independent set in $\hat{G}^{(j)}$ is the union of the maximum weight independent sets among the nodes in each square of the jth grid, and the maximum weight independent set among the nodes in the squares of any grid can be computed in parallel. Each square of the jth grid has $O(\Delta_G k^2)$ nodes for each j. Thus, the time required to compute each of the above maximum independent sets is $2^{O(\Delta_G k^2)}$, and since these sets can be computed in parallel, the overall computation time is $2^{O(\Delta_G k^2)}$ as well. Thus, by selecting $k=\lceil 48/\epsilon\rceil$, the maximum weighted independent set can be approximated within a factor of $1-\epsilon$ in an expected sense using $2^{O(\Delta_G/\epsilon^2)}$ computation time.

ACKNOWLEDGMENT

The authors would like to thank Dr. S. Guha, University of Pennsylvania, and Dr. K. Munagala, Duke University, for numerous discussions on graph partitioning techniques and algorithms for approximating maximum weight independent sets.

REFERENCES

[1] A. Brzezinski, G. Zussman, and E. Modiano, "Distributed throughput maximization in wireless mesh networks—a partitioning approach," presented at the ACM MOBICOM, Los Angeles, CA, Sep. 2006.

[2] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput guarantees through maximal scheduling in multihop wireless networks," presented at the 43rd Annu. Allerton Conf. Commun., Control Comput., Allerton, IL, Sep. 28–30, 2005.

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

[4] A. Dimakis and J. Walrand, "Sufficient conditions for stability of longest queue first scheduling: Second order properties using fluid limits," *Adv. Appl. Prob.*, vol. 38, no. 2, pp. 505–521, Jun. 2006.

[5] D. Shah, E. Modiano, and G. Zussman, "Maximizing throughput in wireless networks via gossiping," presented at the ACM SIGMETRICS/IFIP Performance, Jun. 2006.

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

[7] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 388–404, Apr. 2000.

[8] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *IEEE Trans. Inform. Theory*, vol. IT-34, no. 6, pp. 910–917, Sep. 1988.

[9] H. Hunt, M. Marathe, V. Radhakrishnan, S. Ravi, D. Rosenkrantz, and R. Stearns, "Nc-approximation schemes for NP- and PSPACE-HARD problems in geometric graphs," *J. Algorithms*, vol. 26, no. 2, Feb. 1998.

[10] A. Israeli and A. Itai, "A fast and simple randomized parallel algorithm for maximal matching," *Inform. Processing Lett.*, vol. 22, no. 2, pp. 77–80, Feb. 1986.

[11] K. Jung and D. Shah, "Low delay scheduling in wireless networks," in *Proc. ISIT*, 2007, pp. 1396–1400.

[12] K. Jung and D. Shah, "Low Delay Scheduling in Wireless Networks," Tech. Rep., 2007 [Online]. Available: http://web.mit.edu/devavrat/www/delay.pdf

[13] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer, "Local approximation schemes for ad hoc and sensor networks," presented at the DIALM-POMC Cologne, Germany, Sep. 2, 2005.

[14] P. R. Kumar and S. P. Meyn, "Stability of queueing networks and scheduling policies," *IEEE Trans. Automat. Control*, vol. 40, no. 2, pp. 251–260, Feb. 1995.

[15] N. Li and J. C. Hou, "Topology control in heterogeneous wireless networks: Problems and solutions," in *Proc. IEEE Infocom*, Hong Kong, China, Mar. 2004, vol. 1, p. 234.

[16] X. Lin and S. Rasool, "Constant-time distributed scheduling policies for *ad hoc* wireless networks," in *Proc. IEEE CDC-ECC'05*, San Diego, CA, Dec. 2006, pp. 1258–1263.

[17] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks," presented at the INFOCOM, Miami, FL, Mar. 2005.

[18] M. Luby, "A simple parallel algorithm for the maximal independent set problem," *SIAM J. Comput.*, vol. 15, no. 4, pp. 1036–1055, 1986.

[19] M. Neely, E. Modiano, and C. Li, "Fairness and optimal stochastic control for heterogeneous networks," presented at the INFOCOM, Miami, FL, Mar. 2005.

[20] 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*. Heidelberg, Germany: Springer, 2004, pp. 214–221.

[21] D. Peleg, *Distributed Computing: A Locality-Sensitive Approach*. Philadelphia, PA: Society of Industrial and Applied Mathematics, 2000.

[22] T. Salonidis and L. Tassiulas, "Distributed dynamic scheduling for end-to-end rate guarantees in wireless ad hoc networks," in *Proc. 6th ACM Int. Symp. Mobile ad hoc Networking Comput.*, Urbana-Champaign, IL, 2005, pp. 1545–156.

[23] S. Sarkar and K. Kar, "Achieving 2/3 throughput approximation with sequential maximal scheduling under primary interference constraints," presented at the 44th Annu. Allerton Conf. Commun., Control Comput., Allerton, IL, Sep. 27–29, 2006.

[24] G. Sharma, R. R. Mazumdar, and N. B. Shroff, "On the complexity of scheduling in wireless networks," presented at the ACM MOBICOM, Los Angeles, CA, Sep. 2006.

[25] R. Srikant, S. Sanghavi, and L. Bui, "Distributed link scheduling with constant overhead," presented at the ACM SIGMETRICS, 2007.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.

[26] L. Tassiulas, "Scheduling and performance limits of networks with constantly changing topology," *IEEE Trans. Inform. Theory*, vol. 43, no. 3, pp. 1067–1073, May 1997.
[27] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," in *Proc. INFOCOM*, 1998, pp. 533–539.
[28] L. Tassiulas and A. Ephremidis, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Automat. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.
[29] X. Wu and R. Srikant, "Regulated maximal matching: A distributed scheduling algorithm for multihop wireless networks with node-exclusive spectrum sharing," in *Proc. IEEE CDC-ECC'05*, Seville, Spain, Dec. 2005, pp. 5342–5347.

Saswati Sarkar (S'98–M'00) received the M.Eng. degree in electrical communication engineering from the Indian Institute of Science, Bangalore, in 1996 and the Ph.D. degree in electrical and computer engineering from the University of Maryland, College Park, in 2000.

She is currently an Associate Professor in the Department of Electrical and Systems Engineerin, University of Pennsylvania, Philadelphia. Her research interests are in resource allocation and performance analysis in communication networks.

Dr. Sarkar received the Motorola Gold Medal for the best masters student in the division of electrical sciences at the Indian Institute of Science and the National Science Foundation (NSF) Faculty Early Career Development Award in 2003. She was an Associate Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2001 to 2006.

Saikat Ray received the B.Tech. degree in electronics and communications engineering from the Indian Institute of Technology, Guwahati, in 2000, and the M.S. and Ph.D. degrees in electrical engineering from Boston University, Boston, MA, in 2002 and 2005, respectively.

He spent the summers of 2001 and 2003 as an Intern with Fujitsu Network Communications, Acton, MA, and Microsoft Research, Cambridge, U.K., respectively. He was a Post-Doctoral Researcher in the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, from September 2005 to December 2006. Since January 2007 he has been an Assistant Professor at the University of Bridgeport, Bridgeport, CT.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on December 11, 2008 at 10:30 from IEEE Xplore. Restrictions apply.