Improved Throughput Bounds for Interference-aware Routing in Wireless Networks

Chiranjeeb Buragohain and Subhash Suri
Department of Computer Science
University of California
Santa Barbara, CA 93106, USA
{chiran,suri}@cs.ucsb.edu

Csaba D. Tóth
Department of Mathematics
MIT
Cambridge, MA 02139, USA
toth@math.mit.edu

Yunhong Zhou
HP Labs
1501 Page Mill Road
Palo Alto, CA 94304, USA
yunhong.zhou@hp.com

***Abstract*— We propose new algorithms and improved bounds for interference-aware routing in wireless networks. First, we prove that n *arbitrarily matched* source-destination pairs with average distance d, for any $1 \leq d \leq \sqrt{n}$, in an $O(n)$ size grid network achieve throughput capacity $\Omega(n/d)$. By a simple packing argument, this is also an upper bound in the worst-case. We show that, interestingly, the $\Omega(n/d)$ throughput can be achieved with *single-path* routing, and present a simple distributed algorithm to compute these routes. For arbitrary networks, we propose a new *node-based* linear programming formulation (LP-NODE) that leads to an improved worst-case throughput bound. Specifically, we show that the throughput delivered by LP-NODE is at least $1/3$ of the optimal, improving the previous best of $1/8$. In addition, we show that for certain special topologies, such as tree-structured networks, LP-NODE yields optimal throughput.**

The multipath routes produced by our linear program can be replaced by single-path routes using randomized rounding, at a loss of $O(\log n)$ factor in the throughput. Achieving a constant factor throughput approximation using single-path routes in arbitrary networks seems difficult, and we prove that several natural candidates for single-path routing fail to achieve a constant-factor throughput and, in fact, do arbitrarily poorly.

Finally, we report on the experimental evaluation of our algorithms. In simulations, LP-NODE achieves a throughput typically within 10% of the optimal, and significantly higher (almost twice) than the previous edge-based LP formulations.

I. INTRODUCTION

Interference is a fundamental limiting factor in wireless networks. Due to interaction among transmissions of neighboring nodes and need for multi-hop routing in large networks, it is a non-trivial problem to estimate how much throughput a network can deliver. In an important piece of work, Gupta and Kumar [4] showed that in a random model, where n identical nodes are distributed uniformly in a unit square and each node is communicating with a random destination, the capacity of the network as measured in bit-meters/sec is $O(\sqrt{n})$. This result articulates the packing constraint of the n paths: on average each path is $\Theta(\sqrt{n})$ hops long, and thus in the space of size $O(n)$, only $O(\sqrt{n})$ paths can be accommodated.

The Gupta-Kumar result is quite elegant, but its relevance to practical networks is dubious because it assumes *random* source-destination (s–t) pairs. As Li et al. [12] point out, such an assumption may hold in small networks, but it is unlikely that communication patterns would be distributed uniformly at random as the network scales. It is more relevant to ask: what is the maximum possible throughput for a given network layout and a given set of s–t pairs? Jain et al. [5], Alicherry et al. [1], and Kumar et al. [10] have investigated the capacity of wireless networks for arbitrary source-destination pairs, and arbitrary networks. They all model the problem as a linear program (LP), and provide a computational scheme for estimating the throughput. This is indeed an important direction and, as one of our main results, we present a novel *node-based* LP formulation which, combined with a *node ordering* technique, yields a 1/3 approximation of the optimal throughput— this improves the previous best lower bound of $1/8$. But we first begin with a more fundamental question.

Is there a generalization of the Gupta-Kumar result for arbitrary networks and arbitrary sets of s–t pairs? In other words, can one estimate the network capacity in broad terms, without resorting to computational techniques? And how widely does the capacity vary for different choices of s–t pairs in the network?

Of course, it is easy to observe that without some additional parameters this question is not particularly meaningful. Because we measure throughput in the number of bits transmitted (and not bit-meters as Gupta and Kumar), the capacity can vary widely depending on how far apart the sources and destinations are. If each source is adjacent to its destination, then we can achieve a throughput of $\Theta(n)$; if source-destination pairs are $\Theta(n)$ distance apart (as in a path graph), then the throughput drops to $O(1)$. Thus, a natural and important parameter is the *distance* between the source and destination nodes.

However, even if two input instances have roughly equal average distance between s–t pairs, their throughputs can vary widely. If a constant size cut separates all source-destination pairs, then the throughput is only $O(1)$. This may occur, for instance, if a single node with a few neighbors is the source of all network traffic. Thus, under arbitrarily structured networks, there seems little hope of a general theorem that characterizes the maximum throughput tightly. We show, however, that there is an intermediate ground of *structured network* and *given* s–t pairs, where such a characterization is possible. The special structure we consider is a *grid network* where the s–t pairs form a *matching*. The following subsection details the main contributions of our paper.

A. Our Contributions

1) Suppose we have n *arbitrarily matching* of s–t pairs in an $\Theta(\sqrt{n}) \times \Theta(\sqrt{n})$ size grid network. We show that if the average (hop) distance among the s–t pairs is d, then it is always possible to achieve a total throughput of $\Omega(n/d)$. There are instances where this bound is tight. The $\Omega(n/d)$ throughput in a grid network can be achieved by a simple routing scheme that routes each flow along a *single* path. Both the routing and the scheduling algorithms are simple, deterministic, and distributed. Thus, for the grid topology networks, one can achieve (asymptotic) worst-case optimal throughput *without resorting to* computationally expensive LP type methods.
2) Our second result concerns an approximation bound for the throughput in a general network: *arbitrary network* layout and *arbitrary s–t pairs*. In contrast to previous work [5], [1], [10], we introduce two novel ideas: improved interference constraints at the *node* level, and an *ordering* on all the nodes. As a result of these two ideas, we achieve an approximation ratio of 3 for the optimal throughput, improving all previous bounds.
3) An interesting corollary is that LP-NODE, our node-based LP formulation, yields *provably optimal* throughput if the network topology has a special structure, such as a tree. Tree-like networks may be quite natural in some wireless mesh networks, especially at the peripheries.
4) We show through experimentation that LP-NODE delivers excellent performance. In most cases, it achieves twice the throughput of the edge-based LP, and within 10% of the optimal.
5) All LP based techniques split flows across multiple paths, and an obvious question is to bound the integrality gap between the optimal multi-path and single-path routes. Several natural heuristics based on the classical shortest path schemes are possible that try to route flows along single-paths, while taking into account the interference between paths. Simulations studies [10] suggest that, for random inputs, such routing schemes can give acceptable results. However, little is known about the worst-case performance of these heuristics. In this paper, we establish upper bounds on the performance of three natural routing schemes, and show that all of them can have arbitrarily bad throughput. On the other hand, in the special case of grid networks, we show that an $O(1)$ approximation with single-path routing is possible.

II. PRELIMINARIES AND RELATED WORK

We assume a standard graph model of wireless networks. The network connectivity is described by an undirected graph $G = (V, E)$, where V denotes the set of ad-hoc wireless nodes in a terrain, and E denotes the set of node-pairs that are neighbors. The communication radius of every radio node $i \in \{1, 2, \dots, n\}$ is R; throughout the paper, we assume that the communication occurs on a single radio channel, although the extension to multiple channels is straightforward. Each communicating node causes interferences at all other nodes within distance ϱ from it, where $\varrho \geq R$, is called the *interference radius* of the node. Note that we assume that all radios have an identical communication radius R, and an identical interference radius ϱ. In order to simplify the discussion, we assume that $\varrho = R$, but all our arguments can be easily extended to the general case of $\varrho > R$.

A problem instance is a network $G = (V, E)$, and a set of k source-destination pairs (s_j, t_j), $j = 1, 2, \dots, k$, where s_j and t_j are nodes of V. We assume that each source s_j wants to transmit to its target t_j at a normalized rate of 1. For simplicity, we also assume that the channel capacity is also 1; again, these can easily be generalized to different rates. Our problem is to maximize the network *throughput*, which is the total amount of traffic that can be scheduled among all the s–t pairs subject to the capacity and interference constraints.

A. Models of Interference

The wireless network uses a broadcast medium, which means that when one node transmits, it causes interference at the neighboring nodes, preventing them from receiving (correct) signals from other nodes. The details of which nodes cause interference at which other nodes depend on the specifics of the MAC protocol being used. In this paper, we adopt the interference model corresponding to the IEEE 802.11-like MAC protocols, which is currently the most widely used MAC protocol in wireless networks. Under this protocol, transmitters are required to send RTS control messages and receivers to send CTS and ACK messages. Two edges *interfere* if either endpoint (node) of one edge is within the interference radius ϱ of a node of the other edge. In other words, the edges ij and kl interfere if $\max\{\mathrm{dist}(i, k), \mathrm{dist}(i, l), \mathrm{dist}(j, k), \mathrm{dist}(j, l)\} \leq \varrho$. It is clear that whenever a set of edges pairwise interfere with each other, then only one of those edges can be active at any point of time.

There are several other models of interference in the literature. The *protocol model* introduced by Gupta and Kumar [4] assumes that the transmission from node i is received correctly at j if no other node k is transmitting within interference range ϱ of j. This model corresponds to MAC protocols that *do not require an ACK from the receiver*. The throughput of a network can be higher under the protocol model because it assumes a weaker interference condition than the 802.11-like protocols. The *transmitter model* introduced in Kumar et al. [10] assumes that two transmitting nodes are in conflict unless they are separated by *twice the interference range* (2ϱ). The interference condition assumed here is unnecessarily stronger than 802.11 MACs and leads to a lower estimate of throughput of the network. While we have chosen to work with the 802.11 model of interference, our methodology is quite general, and can be applied to these other models as well.

B. Organization

The remainder of the paper is organized as follows. In Section II-C, we briefly review related previous work. In

Section III, we consider the network capacity in grid-like networks for arbitrary matched s–t pairs. In Section IV, we describe our node-based linear programming scheme for arbitrary networks and arbitrary s–t pairs, and prove that it yields 1/3 of the optimal throughput. In Section V, we discuss the throughput problem under the single-path constraint, and present constructions that exhibit poor performance by natural greedy schemes in the worst-case. In Section VI, we discuss our experimental results. We conclude with some future directions in Section VII.

C. Related Work

Gupta and Kumar [4] provide (near) tight bounds on the throughput capacity of a *random* network, where the nodes are placed randomly in a square and sources and destinations are randomly paired. They show that the expected throughput available to each node in the network is $\Theta(1/\sqrt{n})$. Their result essentially suggests that interference leads to a *geometric packing* constraint in the medium. In a follow up paper, Li et al. [12] carried out simulations and experiments to measure the impact of interference in some realistic networks. They made the case that uniformly random s–t pairs might be very unlikely in practice. They argue that if s–t pairs are not too far from each other then the throughput improves; in fact, they observe that the throughput is bounded by $O(n/d)$ if the average s–t separation is d. They cannot tell, however, if this throughput bound can always be achieved. Kyasanur et al. [11] have recently extended the work of Gupta and Kumar [4] and studied the dependence of total throughput on the number of radio channels and interfaces at each node.

While the results of Gupta-Kumar and Li et al. focused on random or grid-like networks, they did not address a very practical question: given a particular instance of a network and a set of s–t pairs, how much throughput is achievable? Jain et al. [5] formalized this problem, proved that it is NP-hard, and gave upper and lower bounds to estimate the optimal throughput. Their methods, however, do not translate to polynomial time approximation algorithms with any provable guarantees. Kodialam et al. [7] studied a variant of the throughput maximization problem for arbitrary networks, but they do not consider the effect of interference in detail. The interference constraints were very simply modeled by requiring the nodes not to transmit and receive at the same time. The actual patterns of interference in a realistic wireless network are more complex. Recently Padhye et al. [13] have taken significant steps to measure interference between real radio links.

The interference effects in wireless networks can be reduced by utilizing multiple radio channels and interfaces. Raniwala et al. [15] have designed and implemented a multichannel wireless network. Draves et al. [3] have proposed routing metrics to efficiently route in such networks. On the theoretical side, the problem of maximizing throughput in a network using multiple radio channels and interfaces have been studied by Alicherry et al. [1] and Kodialam et al. [8].

Kumar et al. [10], [9] were the first to give a constant factor approximation algorithm for the throughput maximization problem in a network with a single radio channel. In particular, they give a 5-approximation algorithm under the *transmitter model*. As we mentioned earlier, the transmitter model is unduly restrictive compared to the 802.11-like models, and their algorithm does not give any explicit approximation bound for the 802.11 model. Alicherry et al. [1] considered the problem of routing in the presence of interference with multiple radio channels and interfaces. As part of that work, they give an approximation algorithm for the throughput maximization problem with a constant factor guarantee under the 802.11-like model using interference constraints between edges. Their approximation factor is 1/8 for the case of $\varrho = R$, and it becomes progressively worse as ϱ becomes larger compared to R. By contrast, our approximation factor is 1/3, and does not depend on the ratio ϱ/R.

III. Maximum Throughput for Grid Topologies

Before we discuss our linear programming approach for computing interference-aware routes in arbitrary wireless networks with arbitrary s–t pairs, it is worth asking to what extent one can estimate the throughput using *structural* facts, in the style of Gupta and Kumar [4]. In other words, are there simple characterizations of the network and the s–t distributions that allow us to derive good estimates of the achievable throughput *without* resorting to computationally expensive methods such as linear programming. We do not know of any result of this type for completely general setting (nor is one likely to exist), but we show below that for special network topologies, such as grids, and if each node participates in a bounded number of s–t pairs, one can obtain a bound on achievable throughput based on average separation among source-destination pairs. Furthermore, we also show that a simple and distributed routing scheme can achieve the optimal throughput using *single* paths.

Consider a grid network of size $\Theta(\sqrt{n})\times\Theta(\sqrt{n})$, which can be thought of as a square lattice in the plane. We assume there are n s–t pairs, arbitrarily chosen by the user (or adversary), and each lattice point appears in $O(1)$ source and destination nodes. We assume that $R = \varrho = 1$, each edge in the network has capacity 1, and each source wants to communicate with its destination at the rate of 1. These demands are assumed to be persistent, i.e., the flow demands are constant over time and we are interested in the steady state flow. We wish to maximize the total throughput among all s–t pairs. (For the moment, we will not worry about fairness among different pairs, but will discuss that issue briefly in Section VI-C.)

A. Manhattan Routing

We first consider the case when each s–t pair has (lattice) distance d. In the following subsection, we will generalize the result to average distances. A simple packing argument shows that the maximum possible throughput is at most $O(n/d)$; a similar observation was also made in Li et al. [12]. But it is far from obvious that $\Omega(n/d)$ throughput can *always* be realized (for adversarially chosen s–t matchings). By clustering sources on one side, and destinations on the other, it may be possible to create significant bottlenecks in routing.

In fact, one can see that a simple-minded routing scheme can lead to very low throughput. Consider, for instance, the particular choice of s–t pairs shown in Fig. 1. There are 4 source-destination pairs $\{(A, B), (C, D), (F, E), (G, H)\}$. Suppose we route each flow using the shortest paths, staying as close as possible to the straight line joining the s–t pair. These routes are shown using dotted lines in Fig. 1. Observe that all these paths go through a common node N, which becomes the bottleneck, and limits the total throughput to 1. Nevertheless, the following result shows that for any configuration of n source-destination pairs, one can achieve $\Theta(n/d)$ throughput.

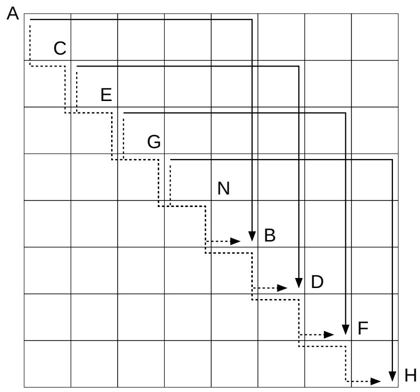

Fig. 1. Illustration of the Manhattan routing. The source-destination pairs are (A,B), (C,D), (E,F) and (G,H).

Theorem 1: Consider n source-destination pairs in an $\Theta(\sqrt{n}) \times \Theta(\sqrt{n})$ size grid, with each grid point appears in $O(1)$ source and destination nodes. Suppose that each s–t pair has (lattice) distance d. Then, one can always achieve the (asymptotic) maximum throughput of $\Theta(n/d)$ using single-path routing.

Proof: Using a packing argument, it is easy to see that $O(n/d)$ is an upper bound on the throughput: the network's total transport capacity (bits $\times$ distance) is $O(n)$, each bit transmitted from a source to its destination travels $\Omega(d)$ distance, and so the maximum throughput is $O(n/d)$.

The main part of the theorem is the lower bound, that $\Omega(n/d)$ throughput can always be realized. We argue that a particular routing strategy, which we call *Manhattan routing*, is able to guarantee $\Omega(n/d)$ throughput. We consider a particular s–t pair, and suppose that the source has coordinates (x_s, y_s) while the destination has coordinates (x_t, y_t). We route along a path that runs from (x_s, y_s) to (x_t, y_s), and then to (x_t, y_t). That is, the path is composed of two segments, forming an L-shape (see Fig. 1).

We now argue that this Manhattan routing has the desired throughput guarantee. For any node i, let $\text{column}(i)$ be the set of nodes that have the same x-coordinate as i; similarly, let $\text{row}(i)$ be the set of nodes that share the same y-coordinate as i. The key observation is that a node k appears in the routing path from i to j if and only if $i \in \text{row}(k)$ or $j \in \text{column}(k)$, and both i and j lie at a distance less than d from k. Since each grid point appears in $O(1)$ source-destination pairs, at most $O(d)$ flows are routed through node k. Thus, if we route all the flows using Manhattan routing and allocate $\Theta(1/d)$ capacity to each s–t pair, the resulting flow will be feasible and respects all interference constraints. This gives a feasible throughput of $\Theta(n/d)$, and so our proof is complete. ■

B. Extension to average or median distances

It is somewhat surprising that a simple routing scheme that is oblivious to the global details of source-destination placements can route flows (asymptotically) optimally. We now show that the result actually holds more broadly, for the case when d is either the *average* or the *median* distance among all pairs.

Theorem 2: Consider n source-destination pairs in an $\Theta(\sqrt{n}) \times \Theta(\sqrt{n})$ size grid, with each grid point appears in $O(1)$ source and destination nodes. If the *average* or *median* (lattice) distance between the s–t pairs is d, then one can always achieve a throughput of $\Omega(n/d)$.

Proof: We simply observe that if the n pairs have average distance d, then at least half the pairs must be at distance less than $2d$. We set the rate for all the pairs whose separation is larger than $2d$ to zero, and route the remaining pairs using Manhattan routing. By Theorem 1, the throughput of these routes is $\Omega(n/d)$.

A similar argument shows that a throughput of $\Omega(n/d)$ is also achievable when the *median* s–t distance is d. ■

These bounds characterize the throughput of an instance based on just one key parameter: the separation among the source-destination pairs. Given an instance of the problem, a network manager can now deduce the asymptotic worst-case optimal throughput of the network simply from the distances among the source-destination pairs. From a network manager's perspective, this result is an encouraging one: while the traffic matrix of a network is beyond control, the network topology is something she can control. Thus, our result suggests that in sufficiently regular network topologies, one can consistently achieve high throughput *and* do so through single-path routing.

IV. Throughput in Arbitrary Wireless Networks

In this section, we consider the general problem of estimating the throughput for a given (arbitrary) network with arbitrary s–t pairs (namely, the problem defined in Section II).

A. A Linear Programming Approach

The throughput maximization problem is a joint routing and scheduling problem: route a multicommodity flow between the s-t pairs and schedule the links subject to the interference constraints.

We first consider the routing part and formulate a linear program with flow conservation constraints. We will add additional interference constraints later to guarantee schedulability of the underlying flow. The throughput maximization problem

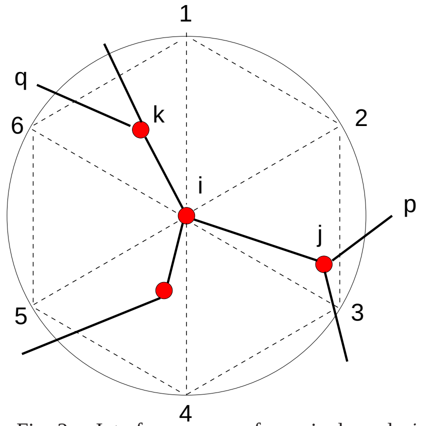

Fig. 2. Interference zone for a single node i

for a single s–t pair with only flow constraints is just the classical max-flow problem:

$$\begin{aligned}\text{Maximize} \quad & \sum_{i \in N(s)} f_{si} \quad \text{subject to} \\ \sum_{j \in N(i)} f_{ij} \quad &= \sum_{j \in N(i)} f_{ji}, \;\; \forall\, i \neq s,t \\ 0 \leq \quad f_{ij} \quad &\leq \quad 1, \;\; \forall\, ij \in E \end{aligned} \tag{1}$$

where f_{ij} denotes the amount of flow in edge ij from node i to node j, for each edge $ij \in E$, and $N(i)$ denotes the set of nodes adjacent to i. The objective function maximizes the total flow out of s subject to the capacity constraint on each edge; the other constraint imposes the flow conservation condition at each intermediate node. In order to simplify the discussion, we have assumed that there is only one source-destination pair (s,t). The extension to multiple pairs is straightforward: in each term, we sum over all flows instead of just one.

Given the routes of a multicommodity flow, it is nontrivial and sometimes infeasible to find a schedule subject to interference constraints. We now describe how to add (restrictive) interference constraints to guarantee schedulability.

Jain et al. [5] modeled schedulability under interference constraints using an *independent set* framework, which attempts to assign time slots to all maximal independent sets of the *conflict graph*. [1] This approach yields optimal throughput, however it takes exponential time since the total number of maximal independent sets is normally exponentially large. Alternative approaches [9], [10], [1] try to enforce *restrictive* interference constraints to guarantee schedulability of the underlying flow. This may sacrifice optimality but hopefully it remains close to the optimum. Alicherry et al [1] added *restrictive* interference constraints on the link level, and Kumar et al. [10] did the same with an additional ordering over all the edges. Our approach is similar to [1], [10], however improves both by adding *less restrictive* interference constraints on the node level and a global ordering on all the nodes. These two ideas allow us to guarantee an approximation ratio of 3.

[1]Nodes of the conflict graph are links in the physical connectivity graph and edges are mutually interfering links.

Schedulability with Node Level Interference Constraints

Let us assume that the flow of data through the network is infinitely divisible like fluids. In a steady state solution, an edge ij carries the flow of $f_{ij} \leq 1$ (recall that each edge has unit capacity). This means that in every *unit time interval*, the edge ij is required to be active for an f_{ij} fraction of time and remain inactive for the rest of the time. We introduce two sets of variables τ_{ij} and τ_i, resp., so that τ_{ij} represents the *total* fraction of the unit time interval that an edge ij is active and similarly τ_i is the fraction of time for the node i.

$$\begin{aligned}\tau_{ij} \quad &= \quad f_{ij} + f_{ji} \leq 1, \\ \tau_i \quad &= \quad \sum_{j \in N(i)} \tau_{ij} \leq 1, \;\; \forall\; i \in V. \end{aligned} \tag{2}$$

Using these variables, we can now introduce the *node interference* constraint which enforces the interference restrictions. Consider the node i shown in Fig. 2, and the set of its neighbors (within interference range) denoted by $N(i)$. It is clear that while any node j in the set $N(i) \cup \{i\}$ is transmitting, all other nodes in its range must be inactive unless there is a single node that is communicating with j. Writing $N(i) \cup \{i\}$ instead of the range of j leads us to the following constraint:

$$\sum_{j \in N(i) \cup \{i\}} \tau_j - \sum_{j,k \in N(i) \cup \{i\},\; jk \in E} \tau_{jk} \leq 1, \quad \forall i \in V, \tag{3}$$

where E denotes the edges of the interference graph. To understand this inequality, let us consider which nodes can be active for how long in any given unit time interval. The first term in LHS, counts the total amount of time (out of the unit time interval) that nodes are active in the neighborhood of i. The second term accounts for the fact that if two nodes j and k in the neighborhood of i are communicating with each other, the time they spend communicating to each other should be counted only once.

By construction, if the nodes satisfy condition (3), then the flow is definitely free of interference. But condition (3) is actually more restrictive than necessary. For instance, consider the nodes j and k in Fig. 2, which are separated by a distance larger than the radio range. Constraint (3) implies that the edges jp and kq can not be active at the same time, while in reality they can. Eliminating such unnecessary constraints is key to our improved analysis, and so we next introduce the idea of node ordering.

Node ordering

Consider a total order on the nodes. (We will prescribe a specific order shortly.) Observe that the interference relation is symmetric. If nodes i and j interfere with each other, then constraint (3) imposes the interference condition twice: once when we consider the neighborhood of i and once for j. Therefore, if i precedes j in the ordering, then it is enough to only consider the constraint introduced by i on j. Specifically, let $N_L(i)$ denote the set of interfering nodes *preceding* node i in the ordering, then the following relaxed constraint still ensures an interference-free schedule.

$$\sum_{j \in N_L(i)\cup\{i\}} \tau_j - \sum_{j,k \in N_L(i)\cup\{i\},\ jk \in E} \tau_{jk} \leq 1, \quad \forall i \in V. \tag{4}$$

In order to define $N_L(i)$, any arbitrary ordering over the nodes will work. To get a good approximation factor, we specify the following *lexicographical* order on the nodes: i *precedes* j if and only if, denoting the coordinates of the points by $i = (x_i, y_i)$ and $j = (x_j, y_j)$, we have either $x_i < x_j$ or $x_i = x_j$ and $y_i < y_j$.

LP-NODE

We are now ready to describe the complete linear program, which we call LP-NODE.

$$\begin{aligned}
\text{Maximize} \quad & \sum_{i \in N(s)} f_{si} \ \text{ subject to} \\
& \sum_{j \in N(i)} f_{ij} = \sum_{j \in N(i)} f_{ji}, \ \forall\, i \neq s, t \\
& 0 \leq f_{ij} \leq 1, \ \forall\, ij \in E \\
& \tau_{ij} = f_{ij} + f_{ji} \leq 1, \\
& \tau_i = \sum_{j \in N(i)} \tau_{ij} \leq 1, \ \forall\, i \in V, \\
& \sum_{j \in N_L(i)\cup\{i\}} \tau_j - \sum_{j,k \in N_L(i)\cup\{i\},\ jk \in E} \tau_{jk} \leq 1, \ \forall\, i \in V.
\end{aligned} \tag{5}$$

By construction, the solution to LP-NODE leads to a feasible flow. Let the total flow produced by LP-NODE be f_{NODE}. We prove below that f_{NODE} is schedulable, and it gives a factor-3 approximation to f_{OPT}.

Theorem 3: The flow produced by the solution of LP-NODE is schedulable and satisfies $f_{\mathrm{NODE}} \leq f_{\mathrm{OPT}} \leq 3 f_{\mathrm{NODE}}$.

Proof: The schedulability of f_{NODE} is guaranteed by constraint (5). Details are omitted due to space limit. Similar proofs can be found in [10] and [1]. The inequality $f_{\mathrm{NODE}} \leq f_{\mathrm{OPT}}$ follows directly from the definition of f_{OPT} and the feasibility of f_{NODE}.

For the second part of the inequality, we will show that $f_{\mathrm{OPT}}/3 \leq f_{\mathrm{NODE}}$. For the optimal solution OPT, let $f_{ij}(\mathrm{OPT})$ denote the flow value over ij for each edge $ij \in E$. We define another solution OPT' with $f_{ij}(\mathrm{OPT}') = f_{ij}(\mathrm{OPT})/3$ for all $ij \in E$. It is obvious that $f_{\mathrm{OPT}'} = f_{\mathrm{OPT}}/3$ and OPT' has a interference-free schedule. If OPT' satisfies the constraints of LP-NODE, then $f_{\mathrm{OPT}'} \leq f_{\mathrm{NODE}}$, and the second part of the inequality follows.

It is easy to verify that OPT' satisfies all the flow constraints of LP-NODE. The only non-trivial constraint is eqn. (5), which is equivalent to

$$\sum_{j \in N_L(i)\cup\{i\}} \tau_j(\mathrm{OPT}) - \sum_{j,k \in N_L(i)\cup\{i\},\ jk \in E} \tau_{jk}(\mathrm{OPT}) \leq 3, \ \forall\, i \in V. \tag{6}$$

Consider a node $i \in V$ as shown in Fig. 2 and the airspace P which is the left half of the unit disk centered at i. All links counted in eqn. (6) have an endpoint in P. At any point in time, we claim that there are at most three active links intersecting with P. This immediately implies eqn. (6). To prove this claim, it is sufficient to show that it is impossible to put four points in P such that all pairwise distances are strictly greater than 1. If four points are placed on the half-disk centered at i, then there exist two points p, q such that $\angle piq \leq \pi/3$. Since $|ip| \leq 1$ and $|iq| \leq 1$, $|pq| \leq 1$. In summary, there are at most three active links in P at any time, and the proof is complete. ■

The above proof can easily be extended to the case that the interference range ϱ is larger than radio range $R = 1$. Consider any $\varrho > 1$. Constraint (5) will now include all nodes which are within interference range of i. We can see from Fig. 2, that within a semicircle of radius ϱ, we can still pack at most 3 nodes which do not interfere with each other and hence the approximation bound given above, holds for *any* $\varrho > R$. By contrast, the approximation ratio given by Alicherry et al. [1] grows monotonically with increasing ϱ; it is 8 when $\varrho = 2R$, 12 when $\varrho = 2.5R$, and so on.

B. Optimal Throughput for Tree-Structured Networks

If the underlying network is a tree, then we show that a variation of our LP-NODE can solve the throughput maximization problem optimally. Given a tree, pick an arbitrary node and root the tree at that node. Then perform a *breadth first search* (BFS) of the tree, starting at the root, and list the nodes in the order they are visited during the search. We now make a *key observation*: for any node i, the set $N_L(i)$ contains exactly one preceding node in this ordering, which is the parent of i. That is, constraint (4) states that at most one edge should be active simultaneously for all edges adjacent to either i or j. This is also a *necessary* condition, since all edges adjacent to either i or j are pairwise interfering with each other. Thus the constraint imposed by condition (5) is optimal, and so the solution of LP-NODE is optimal as long as the ordering of the nodes is given by the BFS order. We summarize this result as follows.

Theorem 4: If the network connectivity graph is a tree, then a solution of LP-NODE with a BFS ordering solves the throughput maximization problem optimally.

V. SINGLE-PATH ROUTING

The solution of any LP formulation described above leads to a multicommodity flow which uses multiple paths to route each commodity from source to destination. In many wireless network protocols, however, data are generally routed along a single-path. In this section we address the question whether it is feasible to achieve near optimal throughput using single-paths only. A straightforward single-path routing approach could be to greedily route flows using an interference-dependent cost metric [10]. We first show that such greedy approaches can have very poor performance, and then outline approaches that can potentially lead to better throughput.

A. Worst-Case Performance of Single-Path Routing Heuristics

We show that three natural heuristics for maximizing throughput under the single-path constraints do quite poorly:

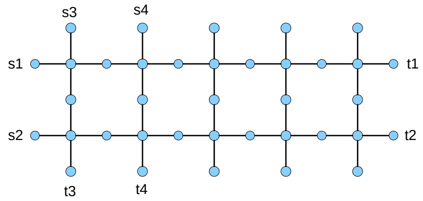

Fig. 3. Lower bound construction for MAXFLOW and ADJUSTFLOW-GLOBAL algorithms.

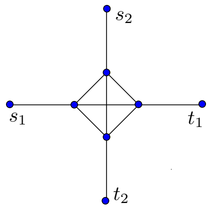

Fig. 4. Two crossing paths.

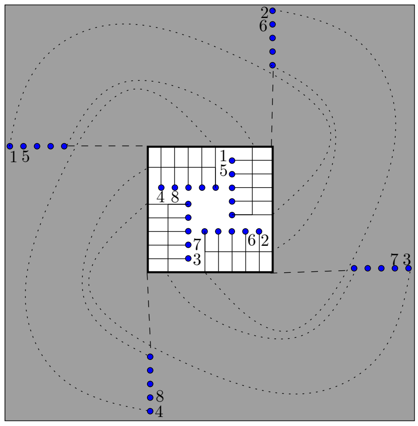

Fig. 5. Lower bound construction for ADJUSTFLOWLOCAL.

when routing k source-destination pairs, their throughput may be $\Omega(k)$ times smaller than the optimum.

We establish our bounds for greedy heuristics that route s–t pairs sequentially. They differ in their policies of how to route the next flow, but they all have the same objective: to maximize the throughput (either system-wide or for a particular flow). In the i-th step, each algorithm chooses a *path* from s_i to t_i along which to route the i-th flow, and the *amount of flow* f_i to send along this route. The three natural greedy heuristics we study are the following:

1) MAXFLOW: This algorithm routes the next flow along a path that maximizes its flow amount. Once a flow has been set up, the algorithm cannot change its route nor its flow amount.
2) ADJUSTFLOWGLOBAL: This algorithm can *reduce* the flow for any previously routed flow, but it cannot change the route for any existing flow. The algorithm adjusts previous flows, if necessary, and routes the next one, to maximize the *current total* throughput.
3) ADJUSTFLOWLOCAL: Like the previous algorithm, this algorithm cannot change the route for any existing flow, but it can *reduce* the flow along any previously routed flow. In particular, when multiple flows share a link or interfere, the bandwidth is shared equally among the flows (and so previously routed flow amounts are never reduced to zero). But, unlike the previous algorithm, this algorithm adjusts the previous flows to maximize the flow of the *next* s–t pair.

All these algorithms are *interference-aware* and *fair-sharing*. In terms of computational power, all three algorithms described above have unlimited power in choosing the routes; but once chosen, they are not to be altered. However, the algorithm can still reduce the previously scheduled flows if doing so helps it maximize the throughput for the future flows. Thus, these algorithms are fairly powerful versions of the simple minimum-hop routing. Nevertheless, as we show below, in the worst-case, their throughput can be $1/k$ of the optimal, where k is the number of s–t pairs in the input.

In order to understand the behavior of MAXFLOW, consider the example network in Fig. 3. This greedy scheme schedules the first two pairs (s_1, t_1) and (s_2, t_2), which then completely block the remaining $k-2$ pairs. (Because the algorithm does not alter the previously set up flows, there is no interference-free capacity available along any of the paths s_i to t_i, for $i \geq 3$.) The optimal solution, on the other hand, routes unit flows along all s_i–t_i pairs, for $i = 3, 4, \ldots, k$. Thus, this routing strategy has performance ratio $\Omega(k)$.

The same example also shows the poor performance of the second routing strategy ADJUSTFLOWGLOBAL. It routes the first two pairs (s_1, t_1) and (s_2, t_2) along the conflict-free horizontal paths. Now consider any subsequent pair s_i–t_i, $i \geq 3$. Because the only s_i–t_i path interferes with both s_1–t_1 and s_2–t_2, any flow allocation that does not set f_i to zero, has total throughput strictly smaller than 2. Thus, the algorithm again rejects all subsequent pairs (s_i, t_i), for $i = 3, 4, \ldots k$. An optimal solution establishes $k-2$ interference-free routes, and so this algorithm also achieves only $O(1/k)$ fraction of the optimal.

For ADJUSTFLOWLOCAL, we present a geometric construction. Consider a simple path in a network carrying a single unit of traffic. When two such paths intersect, they can interfere with each other and reduce their individual throughput. If two paths cross as depicted in Fig. 4, then they can jointly carry the same amount of traffic as each one would carry independently. In our general construction (see Fig 5), we have $4k$ pairs (s_i, t_i), $i = 1, 2, \ldots, 4k$. Curves on the figure represent a sequence of nodes that do not interfere with nodes off the path, the gray area at the perimeter denotes a dense wireless cloud. Crossings inside the middle square are realized as in Fig. 4. An optimal solution can achieve $\Omega(k)$ total throughput, since any pair can be connected by interference-free paths through the wireless cloud. ADJUSTFLOWLOCAL may route the first four pairs along the *shortest paths* without interference. But the first four paths form a rectangle that separates the remaining (s_i, t_i), pairs for all $i > 4$. The flow F_5 has to cross at least one of the first four paths, therefore the total flow volume cannot increase, even if 5 maximizes its own flow on the expense of previous flows. So ADJUSTFLOWLOCAL can just as well choose the shortest paths (s_5, t_5). Inductively one can see that ADJUSTFLOWLOCAL optimizes the volume for each pair (s_i, t_i) if it chooses the shortest path. This means that $k/4$ collides on each side of the rectangle, and the total flow volume is only $\Theta(1)$.

B. LP Rounding Techniques and Single-Path Routing

The preceding discussion highlights some of the difficulties in designing single-path routing schemes. A more sophisticated approach is to utilize algorithms designed for the *Unsplittable Flow Problem* (UFP). Given an undirected graph $G(V, E)$ with edge capacities c_e, and k source-destination pairs (s_i, t_i) with demands q_i, the UFP asks for the maximum multicommodity flow where each commodity is *routed along a single-path*. Without the interference constraint, our throughput problem in wireless networks is exactly equivalent to the UFP. For arbitrary network topologies, Raghavan and Thompson [14] pioneered a *randomized rounding* scheme that constructs a single-path flow from the multipath flow solution achieved by the linear program solution. Thus, we can apply this idea to the solution produced by our LP-NODE. However, this solution is not entirely satisfactory because it loses a factor of $O(\log n)$ in the final throughput.

Although a constant factor approximation algorithm for arbitrary graphs remains elusive, Kleinberg [6] describes an offline $O(1)$-approximation and an online $O(\log n)$-approximation algorithm for the unsplittable flow problem on grid-like graphs. This work was built upon earlier results of Raghavan-Thompson [14] and Awerbuch et al. [2]. Relying on Kleinberg's work, we can show that the maximal single-path throughput problem on grid-like graphs can be reduced to the unsplittable flow problem; which immediately leads to a constant factor approximation algorithm for the single-path routing problem. We point out that this result differs from the setting of Section III, because the source-destination pairs are not required to form a matching. In other words, this result holds for arbitrary s–t pairs, furthermore the number of s–t pairs can be larger than the network size. This result is fairly technical and, due to its complexity and length, we omit the details, and simply state our main result.

Theorem 5: Given a square grid network and an arbitrary set of source-destination pairs, there exists a polynomial-time $O(1)$-approximation algorithm to maximize end-to-end throughput with single-path routing.

VI. EXPERIMENTAL RESULTS

In this section, we report on the experimental evaluation of our algorithms, and discuss the results of our simulations. We ran experiments on both the regular as well as random networks. The random networks consist of n nodes spread over a square $\sqrt{n} \times \sqrt{n}$ area with radio range 3.0. Any two nodes which are within radio range can communicate. This radio range was chosen so that the network is connected with probability close to one. We assume a bidirectional MAC protocol (like 802.11) and that all radio ranges as well as all interference ranges are the same. We assume that each link can support 1 unit of throughput.

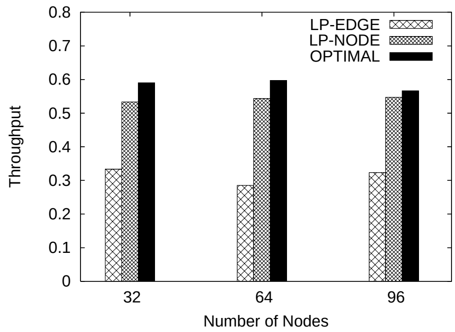

Fig. 6. Performance of the LP-NODE, LP-EDGE and OPTIMAL algorithms with 32, 64 and 96-node networks.

In our evaluation, we used three algorithms:

- LP-NODE: This is our main linear program described in Section IV. This algorithm has provable worst-case approximation ratio of 3.
- LP-EDGE: This is the best previously known linear programming based scheme, as described in Alicherry et al. [1]. This algorithm has an approximation ratio of 8, under the condition that $\varrho = R$.
- OPTIMAL: Since the throughput maximization problem is NP-Complete, there is no polynomial time scheme to compute the maximum throughput. We therefore use the independent set enumeration method described by Jain et al. [5] by adding maximal independent sets into interference constraints until adding more independent sets does not improve the throughput any more. At this point we declare convergence and use the final throughput as optimal.

A. Throughput Scaling With Network Size

In this experiment, we tested how the performance of LP-NODE scales with the network size. We used a random network layout where the nodes were distributed uniformly at random in a square. The source and destination are located at diagonally opposite corners. We then increased the number of nodes in the network from 32 to 64 to 96. In each case, we also computed the optimal throughput f_{OPT} by running the OPTIMAL algorithm.

In Fig. 6 we plot the throughput of the OPTIMAL, LP-NODE and LP-EDGE algorithms. Our LP-NODE algorithm shows excellent performance and yields close to 90% of the optimal throughput. By contrast, LP-EDGE performs much worse and achieves only 50%-60% of the OPTIMAL. In fact, even with a single source-destination pair, LP-EDGE at times failed to achieve 1/3 of the optimal throughput, which one could have achieved by routing along a single-path [12]! With a single s–t pair, the maximum possible throughput using multipath routing is $5/6$; by contrast, the maximum throughput using

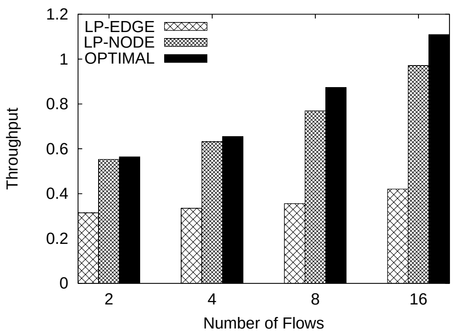

Fig. 7. The total throughput with different numbers of s–t pairs and a 64-node network.

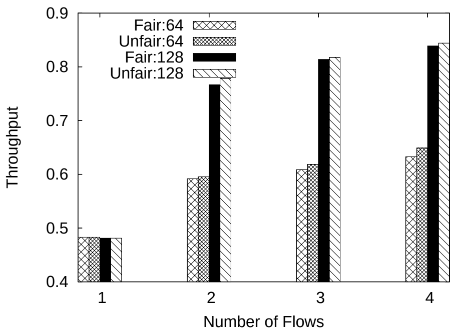

Fig. 8. Effect of fairness on total flow for 64 and 128-node networks. Note that the fairness constraint lowers total throughput by only a small amount.

a single-path is $1/3$. In these cases, the constant factors in the approximation algorithms become crucially important, and LP-NODE does well.

B. Throughput Scaling with Source-Destination Pairs

In this experiment, we fixed the network and increased the number of s–t pairs in the network to evaluate the throughput that the various routing schemes achieve. We used a random network topology with 64 nodes and up to 16 source-destination pairs organized in a crosshatch pattern. In Fig. 7 we plot the total throughput using LP-EDGE, LP-NODE and OPTIMAL algorithms for different number of source-destination pairs. As expected, we see that the throughput increases with the number of flows, but the dependence is not linear because interference from one set of paths reduces throughput for other pairs. Again, LP-NODE shows excellent performance, reaching near-optimal throughput in most cases, while LP-EDGE achieves less than half the throughput of LP-NODE.

C. Impact of Fairness on Flows

When multiple flows compete for bandwidth, the optimal flow is not necessarily fair. In practice, though, fairness is an important criterion in any network protocol. To investigate the effect of fairness we again used uniformly random nodes in a square with four source-destination pairs which intersect at the center of the square. For multiple flows we enforced the simplest fairness condition that each flow gets an equal amount of the total flow. We computed the total throughput using the LP-NODE algorithm and the results are shown in Fig. 8. As expected we see that enforcing fairness reduces the total throughput, but surprisingly, *the effect is very mild*. In fact for the larger 128 node networks, the throughput for fair and unfair flows is almost identical. This is due to the fact that in larger networks the nodes have a lot of freedom in routing the flows and hence overall interference around any single node is low. Thus every flow can carry equal amounts of traffic without congestion.

VII. DISCUSSION

We have studied the throughput maximization problem in multi-hop wireless networks explicitly taking into account the radio interference. We show that in regular grid networks, a simple distributed *single-path* routing algorithm is able to achieve (asymptotic) worst-case optimal throughput with a dense distribution of source-destination matchings. For arbitrary network layouts and arbitrary s–t pairs, we proposed a novel node-based linear programming formulation that achieves an approximation ratio of 3. We also argued that, for general networks, the prospects for efficient single-path routing using simple heuristic algorithms are more bleak. But if the network has regular structure, such as a grid, then a constant factor approximation is possible.

REFERENCES

[1] M. Alicherry, R. Bhatia, and L. Li. Joint channel assignment and routing for throughput optimization in multiradio wireless mesh networks. In *Proc. ACM MobiCom*, 2005.

[2] B. Awerbuch, Y. Azar, and S. A. Plotkin. Throughput-competitive online routing. In Proc. *IEEE FOCS*, pp. 32–40, 1993.

[3] R. P. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh network. In *Proc. ACM MobiCom*, 2004.

[4] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory* **46** (2000), 388–404.
Erratum: P. R. Kumar. *IEEE Trans. Inform. Theory* **49** (2003), 3117.

[5] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proc. MobiCom*, 2003.

[6] J. Kleinberg. *Approximation Algorithms for Disjoint Paths Problems*. Ph.D Thesis, Dept. of EECS, MIT, 1996.

[7] M. Kodialam and T. Nandagopal. Characterizing the achievable rates in multihop wireless networks: the joint routing and scheduling problem. In *Proc. ACM MobiCom*, 2003.

[8] M. Kodialam and T. Nandagopal. Characterizing the capacity region in multi-radio multi-channel wireless mesh networks. In *Proc. ACM MobiCom*, 2005.

[9] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. End-to-end packet-scheduling in wireless ad-hoc networks. In *Proc. ACM-SIAM SODA*, pp. 1021–1030, 2004.

[10] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. Algorithmic aspects of capacity in wireless networks. In *Proc. ACM SIGMETRICS*, 2005.

[11] P. Kyasanur and N. H. Vaidya. Capacity of multi-channel wireless networks: Impact of number of channels and interfaces. In *Proc. ACM MobiCom*, 2005.

[12] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *Proc. ACM MobiCom*, 2001.

[13] J. Padhye, S. Agarwal, V. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *Proc. Internet Measurement Conference*, 2005.

[14] P. Raghavan and C. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica* **7** (1987), 365–374.

[15] A. Raniwala and T.-C. Chiueh. Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network. In *Proc. IEEE INFOCOM*, 2005.