Characterizing Achievable Rates in Multi-hop Wireless Networks: The Joint Routing and Scheduling Problem

Murali Kodialam Thyaga Nandagopal
Bell Laboratories, Lucent Technologies
101 Crawfords Corner Road
Holmdel, NJ 07733, USA
{muralik, thyaga}@bell-labs.com

ABSTRACT

This paper considers the problem of determining the achievable rates in multi-hop wireless networks. We consider the problem of jointly routing the flows and scheduling transmissions to achieve a given rate vector. We develop tight necessary and sufficient conditions for the achievability of the rate vector. We develop efficient and easy to implement Fully Polynomial Time Approximation Schemes for solving the routing problem. The scheduling problem is a solved as a graph edge-coloring problem. We show that this approach guarantees that the solution obtained is within 67% of the optimal solution in the worst case and, in practice, is typically within about 80 % of the optimal solution. The approach that we use is quite flexible and is a promising method to handle more sophisticated interference conditions, multiple channels, multiple antennas, and routing with diversity requirements.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless Communication; G.2.2 [**Graph Theory**]: Graph Algorithms

General Terms

Algorithms, Design, Theory

Keywords

Wireless Networks, Scheduling, Routing, Graph Coloring, Optimization

1. INTRODUCTION

In this paper, we consider the problem of characterizing the rates that are achievable in a class of multi-hop wireless networks. The primary model that we consider was proposed in Baker, Wieselthier and Ephremides [1]. The main constraint that is imposed in this model is that each node can communicate with at most one other node at any given time. Hajek and Sasaki [2] showed that the link scheduling algorithm in this model can be solved in polynomial time but the algorithm that they develop is not implementable in practice. Further, the main problem that they consider in that paper is a link scheduling problem. The joint routing and scheduling problem is not the primary focus of that paper. Many variants of this problem have been addressed in literature. However, the problem of efficiently determining if end-to-end flows are achievable does not seem to have been addressed in the literature. This is the primary focus of this paper. Our main objective is to develop efficient algorithms for the joint routing/scheduling problem. A secondary objective is to develop a flexible approach that can be used in the case where there is more complex interaction between the transmission in the different links. For example, IEEE 802.11 networks use a RTS-CTS mechanism to schedule the transmission and this imposes a more complex interaction between link transmission. In the last section of this paper, we indicate how the approach developed in this paper can be extended to more complex routing/scheduling objectives.

The seminal paper of Gupta and Kumar [3] shows that the throughput per node in a multi-hop wireless networks with n nodes scales as $O(\frac{1}{\sqrt{n}})$ bit-meters/second. This throughput bound holds under fairly general conditions. This paper however, discusses questions of the following nature: In a given multi-hop network with specified node configurations and wireless link speeds

- What is the maximum data rate that can be jointly routed and scheduled between a given source node and a destination node?
- Is a given rate vector between multiple source-destination pairs jointly achievable ?

The first problem is analogous to the maximum flow problem [4] in regular data networks except that the constraints imposed by link scheduling make this problem more difficult. The second problem is analogous to the multi-commodity flow problem [4]. This problem is non-trivial due to the fact that this problem involves jointly solving a routing and scheduling problem. In the case of communication in the presence of interference, the scheduling problem itself is NP-hard. The routing problem has to be solved over the achievable scheduling space. Even if we assume that there is no interference resulting from communications between different nodes, there is still the constraint that each node can

MobiCom'03, September 14–19, 2003, San Diego, California, USA.

transmit or receive from at most one node at any given instant.

In this paper, we consider a system free of secondary interference, where the only constraint on the nodes is that it can transmit to or receive from at most one node at a time. We address the problem of determining if a given set of source-destination rates are achievable or not and, if achievable, we derive efficient, simple to implement algorithms for routing of flows and scheduling at the nodes to achieve this rate. The algorithms are guaranteed to be within 67% of the optimal solution in the worst case and in practice perform a lot better. The algorithmic approach can be extended to MIMO (Multiple Input Multiple Output) systems, joint routing and scheduling in systems with secondary interference, such as IEEE 802.11, joint routing and scheduling in networks with routing diversity requirements etc. We outline our approach to such extensions later in the paper. We now summarize some of the related work in Section 2.

2. RELATED WORK

The work that is most closely related to this paper is by Hajek and Sasaki [2]. The problem that they consider is the link flow achievability problem in the case of a spread spectrum system. The authors showed that the problem can be solved in polynomial time. They formulated the problem as a linear optimization problem on the fractional matching polyhedron and used the ellipsoid algorithm [5] to solve the problem. The running time of the algorithm is roughly $O(mn^4)$ (where n is the number of nodes in the network and m the number of links) and uses a separation oracle as a subroutine in the ellipsoid algorithm. The paper made an important contribution in showing that the scheduling problem can be solved in polynomial time. However, as the authors point out in their paper, the algorithm developed is not practical. Several authors including Post, Kershenbaum and Sarachick [6], Wieselthier, Barnhart, and Ephremides [7] have considered heuristic procedures for this problem.

The authors in [2] also consider the problem of achieving a given set of source-destination rates. This problem, in theory, can be formulated as a multi-commodity flow problem on the fractional matching polyhedron. It can be solved in polynomial time but it is not clear what the complexity of the solution procedure would be. The approach would not be practical to solve even small sized problems. Further, their approach cannot be extended to interference limited systems.

Jain et. al. [8] have recently proposed a model similar to the one proposed in this paper and the one in [2], using a linear programming approach to characterize networks with interference. They use a conflict graph to model constraints on simultaneous transmissions. They focus on the routing component alone, and assume that the existence of an ideal scheduling mechanism. Moreover, they do not propose any polynomial time approximation algorithms to solve the routing problem. In this paper, we propose both: approximation algorithms that solve both the end-to-end flow routing problem and the link scheduling problem near optimally.

We solve the link flow achievability problem by characterizing the necessary and sufficient conditions for the achievability of a link flow vector. This approach naturally extends to the problem where we have to determine the maximum data rate that can be sent between two nodes in the network. In the case of determining the maximum rate, the problem has a routing component and a scheduling component. The necessary conditions from the link scheduling problem, gives raise to constraints on the routing problem that are needed for schedulability. This is a natural generalization of the maximum flow problem in capacitated networks. We also consider the problem of determining if a given rate vector is achievable. We formulate the problems as linear programming problems with exponential number of variables. We then develop simple to implement primal-dual algorithms for solving the problem.

The approach that we take to solve the routing component of the problem is to formulate it as a linear programming problem with an exponential number of variables. We then use a primal-dual approach to develop a Fully Polynomial Time Approximation Scheme (FPTAS). The idea in FPTAS is to obtain an ϵ optimal solution to the problem. An ϵ optimal solution to the (maximizing) problems that we consider in this paper is a solution to the problem that has a value at least $(1-\epsilon)$ times the optimal solution. An FPTAS is a family of algorithms that finds an ϵ-optimal solution in time that is a polynomial function of the problem parameters and $\frac{1}{\epsilon}$. The problem parameters in our case are the number of nodes in the network n, the number of links in the graph m, and the number of source-destination pairs. The idea of solving linear programming problems approximately with FPTAS originated with the work of Shahrokhi and Matula [9]. There were a series of papers improving and extending these results. This paper follows the algorithm and analysis in Garg and Könemann [10] and Karakostas [11]. For the ease of notation, throughout this paper we use $\mathcal{O}(f)$ to represent $f \log^{O(1)} m$, i.e., we hide polylog terms in $\mathcal{O}$.

3. MODEL AND ASSUMPTIONS

We consider a multi-hop wireless network with n nodes. The nodes communicate with each other via wireless links. Each node in the network can communicate directly with a subset of the other nodes in a network. If node u can transmit *directly* to node v, we represent this fact by a directed edge (link), $u \rightarrow v$, from node u to node v. We assume that there are m links in the network. We represent the nodes in the network and possible communication with a directed graph $G = (V, E)$ where V represents the set of nodes in the network and E the set of directed edges (links) in the network. We do not assume that links are bi-directional.

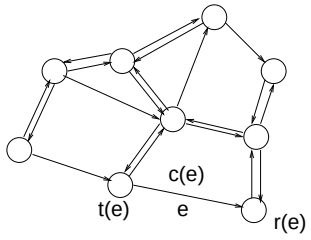

Figure 1: Network Example

Given a link $e \in E$, we use $t(e)$ to represent the transmission end of the link and $r(e)$ to be the receiving end of the link e. We say that a link e is *active* when there is a transmission from $t(e)$ to $r(e)$. We assume that link e can transmit data at $c(e)$ bits/second. Therefore we implicitly assume stationary channel conditions and no power control at the nodes. An example of a network along with the link parameters are shown in Figure 1.

We assume that system operates in a synchronous time-slotted mode. In any given time slot, a node can be in communication with only one another node. The length of each time slot is τ seconds. Therefore, if a link is active for one time slot, then $\tau.c(e)$ bits will be transmitted from $t(e)$ to $r(e)$ in one time slot. In the computation of the schedule for nodes, we assume that the schedule is periodic and has T time slots in each period. We label the time slots $1, 2, \ldots, T$. If the system is asynchronous, then the results in this paper, will provide upper bounds on the performance of the system.

Given a node $v \in V$, we use $N_{in}(v)$ to denote the set of links that terminate at node v. In other words,

$$N_{in}(v) = \{e \in E : r(e) = v\}.$$

Similarly, for a given node $v \in V$, we use $N_{out}(v)$ to denote the set of edges that emanate from v, i.e,

$$N_{out}(v) = \{e \in E : t(e) = v\}.$$

We use $N(v)$ to denote $N_{in}(v) \cup N_{out}(v)$. Note that $N(v)$ is the set of links incident on v. Let Δ_G denote the maximum degree in the network, i.e., the maximum number of neighbors of a node.

$$\Delta_G = \max_{v \in V} |N(v)|.$$

The network model that we consider here occurs with networks with multiple channels available for transmission. Examples are CDMA based multi-hop networks where node pairs can choose the code sequence they want to transmit, networks with beam-forming antennas that enable a highly directed transmission to/reception from another node. We assume that a coordination mechanism that helps hosts decide the transmitting slots and channels exists. Clearly, such systems are already under development, and hence our model is practical, though limited in scope. Our goal, however, is to lay a framework for extensions to other kinds of systems with interference.

In the rest of this paper, we assume that all link speeds, flows and rates are rational numbers. Before we consider the maximum data rate that can be transmitted from a given source node to destination node, we consider the simpler problem of the achievability of a given set of link flows.

4. ACHIEVING LINK FLOWS

We now consider the problem of determining if a set of link flows are achievable. Instead of attempting to solve this problem directly as in [2], we outline simple necessary and sufficient conditions for the achievability of link flows. The objective is to derive a set of simple conditions that can be used to formuate and solve the end-to-end flow requirement problems. Solving the link scheduling problem also serves to illustrate the effect of scheduling on the routing problem. Some of the results in this section are also implicit in [2].

Assume that we are given a m-vector $\mathbf{f}$ where $f(e)$ is the desired flow on link $e \in E$, where flow is defined as the number of bits to be sent in one second. We refer to $\mathbf{f}$ as the link flow vector. The objective now is to determine necessary and sufficient conditions for this link flow vector to be achievable. Note that the flow is specified as a link flow and not as an end-to-end flow. This connection between end-to-end flows and individual link flows will be made in the next section. In order to achieve this link flow we first define a $0-1$ scheduling variable

$$y_e^t = \begin{cases} 1 & \text{If link } e \text{ is active in time slot } t \\ 0 & \text{Otherwise} \end{cases}$$

Note that y_e^t is set to one if there is a transmission on link e in time period t. Since no node can be transmitting or receiving from multiple nodes in a given time slot, we have the inequality

$$\sum_{e \in N(v)} y_e^t \leq 1 \qquad \forall v \in V \quad \forall t \leq T. \tag{1}$$

Note that the fraction of time link e is active is given by

$$\frac{\sum_{t \leq T} y_e^t}{T}$$

Therefore, the mean flow on link e is given by

$$f(e) = \frac{c(e) \sum_{t \leq T} y_e^t}{T}$$

Summing Equation (1) over all $t \leq T$, we get

$$\sum_{t \leq T} \sum_{e \in N(v)} y_e^t \leq T \qquad \forall v \in V$$

Interchanging the order of summation, and dividing by T, we get

$$\sum_{e \in N(v)} \frac{\sum_{t \leq T} y_e^t}{T} \leq 1 \qquad \forall v \in V$$

Therefore

$$\sum_{e \in N(v)} \frac{f(e)}{c(e)} \leq 1 \qquad \forall v \in V$$

We state this condition formally.

LEMMA 1. *If a given link flow vector* $\mathbf{f}$ *does not satisfy any of the following inequalities*

$$\sum_{e \in N(v)} \frac{f(e)}{c(e)} \leq 1 \qquad \forall v \in V$$

then this flow vector is not schedulable.

Since we averaged the scheduling variables over time, the above condition is only a necessary condition. This is illustrated by the following example. The value of $c(e)$ for each link is 1 unit and the value of $f(e)$ for each link is 0.5 units as shown in the Figure 2. Note that this flow vector satisfies

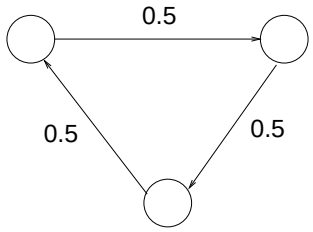

Figure 2: Three Node Example

the utilization bound at all the nodes in the network. However, this flow vector is not achievable. This is so because in any given time slot at most one of the three links can be

active. This gives a throughput of at most $\frac{1}{3}$ on each link in the network. This gap between the necessary condition for achievability and actual achievability of the flow vector is the fact that averaging the scheduling constraints over time relaxes the $0-1$ constraints on the y_e^t variables. We now give a sufficient condition for schedulability. In order to derive the sufficiency conditions, we need a few more definitions.

DEFINITION 1. *A multi-graph is a graph where there may be multiple edges between the same pair of nodes. An alternate representation of a multi-graph is to have an integral weight $w(e)$ on link[1] e in the network G, where $w(e)$ represents the number of edges between $t(e)$ and $r(e)$ in the multi-graph.*

We now define a multi-graph on the network G to help us obtain an achievable schedule for a given link flow vector $\mathbf{f}$. With τ seconds/slot, we can send at most $\tau.c(e)$ bits/slot on link e. Hence, in order to achieve a link flow of $f(e)$ bits/second, link e needs to be scheduled for $f(e)/\tau.c(e)$ slots/second. Therefore, choose a slot time τ such that

$$w(e) = \frac{f(e)}{c(e)} \frac{1}{\tau} \text{ slots/second} \tag{2}$$

is integral. Such a τ exists since all values of $f(e)$ and $c(e)$ were assumed to be rational. There exist many such τ that satisfy the above equation. We choose the largest such τ. The $w(e)$ is the slot rate required to satisfy a flow of f(e) bits/second on a link e with capacity $c(e)$.

DEFINITION 2. *Given the network $G = (V, E)$, the flow vector $\mathbf{f}$ and link speed vector $\mathbf{c}$, let the link weights $w(e)$ be defined as $w(e) = \frac{f(e)}{\tau.c(e)}, \forall e \in E(G)$. The* scheduling multi-graph *$G_S(\mathbf{f}, \tau)$, corresponding to G, has the same node set V, with a link $e \in E(G)$ represented by $w(e)$ edges in $G_S(\mathbf{f}, \tau)$ between the same endpoints.*

It is clear that $w(e)$ represents the link weight in the network graph G, and the edge multiplicity in the scheduling multi-graph $G_S(\mathbf{f}, \tau)$. The maximum degree of a node in the scheduling multi-graph is denoted by Δ. We have

$$\Delta = \max_{v \in V} \sum_{e \in N(v)} w(e)$$

The edges, degree and other parameters of the scheduling multi-graph are dimensionless. Thus, a link between nodes u and v with link weight of $w(e) = 3$ slots per second is represented by 3 edges between nodes u and v in the scheduling multi-graph.

DEFINITION 3. *The chromatic index, L, of a multi-graph is defined as the minimum number of colors needed to color the edges of the multi-graph such that two edges having the same color are not incident on the same node.*

It is easy to see that $L \geq \Delta$. Note that any proper edge coloring of the multi-graph $G_S(\mathbf{f}, \tau)$ can be mapped into an achievable transmission schedule on the network graph G, and vice-versa. To achieve this mapping, let us consider a scheduling frame with slot of length τ seconds each. Each edge e in the scheduling multi-graph $G_S(\mathbf{f}, \tau)$ *corresponds* to one time slot that has to be allocated to the link that represents e, in the network graph G. If a set of edges, W, in $G_S(\mathbf{f}, \tau)$ have color i, then correspondingly, in time slot i, make all the links in G, represented by the edges in W, active. Repeat this process for colors $i = 1, 2, \ldots L$. Note that there will be no conflicts in the schedule in G, due to the proper edge-coloring in $G_S(\mathbf{f}, \tau)$. Moreover, with L colors, we have colored all edges of the multi-graph, which implies that in a schedule of length L slots, each link e in the network graph G is active for $w(e)$ time slots, thereby sending $f(e)$ bits. Thus, we have mapped a proper edge-coloring of the scheduling multi-graph onto a valid transmission schedule in G. This is the reason we are interested in the chromatic index of the graph.

The chromatic index, L, is dependent on the values of the flow $\mathbf{f}$ and the slot length τ. Using the mapping described above, the chromatic index tells us that to achieve a flow of $\mathbf{f}$ bits/second, a slot rate of at least L slots/second are needed, where each slot is of length τ seconds.

The following upper bound on the chromatic index of a multi-graph is due to Shannon [12].

THEOREM 2. *The chromatic index of a multi-graph with maximum degree Δ is at most $\frac{3}{2}\Delta$.*

Determining the chromatic index of a graph is NP-hard. However, in the case of the scheduling multi-graph

- There are fast algorithms to determine a 1.1– approximation to the chromatic index [13].
- The greedy algorithm gets 2-approximate solution to the problem in time $O(\frac{m}{\tau})$.
- An algorithm to construct the $\frac{3}{2}\Delta$ solution can be computed in time $O(\frac{nm}{\tau})$. A constructive proof was provided by Shannon [12].

It is easy to implement the greedy edge coloring algorithm in a distributed manner. Though the algorithm guarantees only a 2-approximation in the worst case, it usually does much better in practice.

Let us consider an example to understand the conditions needed for the achievability of flows. Let the desired link flow be $f(e) = 2$ bits per second, and capacity $c(e) = 4$ bits per second for all links in a given network graph G. Thus, we choose $\tau = 0.5$ seconds per slot, so that $w(e) = 1$ slot/second, $\forall e \in E$. We form the scheduling multi-graph, and let the chromatic index of $G_S(\mathbf{f}, \tau)$ be $L = 3$. This implies that if we have at least L slots per second, then we can achieve the desired flow of 2 bits/second. However, we have only 2 time-slots per second $(1/\tau)$. The flow $\mathbf{f}$ cannot, therefore, not achieved. This can be formally stated in terms of the following theorem.

LEMMA 3. *Let L represent the chromatic index of the scheduling multi-graph $G_S(\mathbf{f}, \tau)$, where τ is the length of a time slot. Then, flow $\mathbf{f}$ is achievable iff the slot rate is at least L slots per second, i.e., iff $L\tau \leq 1$.*

Proof:
Consider the scheduling multi-graph $G_S(\mathbf{f}, \tau)$. Since the time slots are of length τ seconds, we have $1/\tau$ slots per second. If the chromatic index of G_S is L, it implies that in order to meet the link flow demand $\mathbf{f}$, we need a slot rate of at least L slots/second. Thus, $L \leq 1/\tau$ for the flow $\mathbf{f}$ to be achievable.

[1]We use the term *link* exclusively to denote an edge of the network graph $G = (V, E)$.

Since a proper edge coloring of the scheduling multi-graph cannot exist using less than L colors, the corresponding mapping ensures that a valid schedule that satisfies the demand with a slot rate of less than L slots per second cannot be found. Therefore, if $L > 1/\tau$, then $\mathbf{f}$ is not achievable. □

COROLLARY 4. *Let L represent the chromatic index of the scheduling multi-graph $G_S(\mathbf{f}, \tau)$. Given that a slot rate of L slots/second is needed to achieve the flow $\mathbf{f}$, then, with the available slot rate of $1/\tau$ slots per second,*

$$\frac{f(e)}{L\tau}, \quad \forall e \in E$$

is an achievable flow in G.

Proof:
If we have L slots/second, then we are guaranteed to achieve a flow of $\mathbf{f}$ bits/second. However, if there are only $1/\tau$ slots/second, then, using these slots, we can only send $\frac{\mathbf{f}}{L}\frac{1}{\tau}$ bits per second. □

We would like to point out that once the chromatic index L and the corresponding coloring is known for $G_S(\mathbf{f}, \tau)$, we do not have to recompute the chromatic index to find the schedule for a flow of $\mathbf{f}' = \mathbf{f}/L\tau$. By changing the length of the time slot to $\tau' = 1/L$, it is easy to see that $G_S(\mathbf{f}', \tau') = G_S(\mathbf{f}, \tau)$. Thus, $L' = L$ and $L'\tau' = 1$. Therefore, the coloring and hence, the schedules are still the same as before, except that the time slot length has changed.

We now state sufficient conditions for the achievability of a link flow vector $\mathbf{f}$.

THEOREM 5. *An m-link flow vector is achievable if*

$$\sum_{e \in N(v)} \frac{f(e)}{c(e)} \leq \frac{2}{3}, \qquad \forall v \in V$$

Proof:
From the conditions in statement of the theorem, and the definition of $w(e)$, note that

$$\sum_{e \in N(v)} w(e) \leq \frac{2}{3}\frac{1}{\tau}$$

Therefore, $\Delta \leq \frac{2}{3}\frac{1}{\tau}$ and, using Theorem 2,

$$L \leq \frac{3}{2}\Delta \leq \frac{1}{\tau}$$

This implies that $L\tau \leq 1$ and by Lemma 3, $f(e)$ is achievable. □

Therefore when a link flow vector satisfies the sufficiency conditions, it is schedulable. Note that there is a gap between the necessary and sufficiency conditions. Clearly, if there exits a $v \in V$ such that

$$\sum_{e \in N(v)} \frac{f(e)}{c(e)} \geq 1$$

then it not schedulable. If a link flow vector satisfies

$$\sum_{e \in N(v)} \frac{f(e)}{c(e)} \leq \frac{2}{3}, \qquad \forall v \in V$$

then it is schedulable. If

$$\frac{2}{3} < \sum_{e \in N(v)} \frac{f(e)}{c(e)} \leq 1, \qquad \forall v \in V$$

then it is not clear whether it is schedulable or not. We attempt to close this gap (in practice) using the following strategy: As long as the link flow vector $\mathbf{f}$ satisfies the necessary conditions, we construct the scheduling multi-graph and determine its chromatic index. Let τ denote the length of the time-slot and L the chromatic index of the resulting scheduling multi-graph. If $L\tau \leq 1$ then the given link flow vector $\mathbf{f}$ is achievable. If $\mathbf{f}$ satisfies the sufficiency conditions, then clearly it satisfies $L\tau \leq 1$ and is therefore achievable. Using this strategy will result in determining the achievability of link flow vectors that fall in the gap between the necessary and sufficiency conditions. We state the algorithm formally below:

- If the vector $\mathbf{f}$ does not satisfy the necessary conditions then output f is not achievable.
- Determine τ and $w(e)$ as shown in equation (2).
- Construct the scheduling graph and determine it chromatic index L.
- If $L\tau \leq 1$, then f is achievable.

Experimental testing indicates that the above algorithm performs extremely well in practice. The effectiveness of this approach will be illustrated in the case of determining the maximum achievable rate between a given pair of nodes. Also note that the link rates are not just achievable in the long run but is achieved in time at most 1 second.

4.1 Predefined Time Slots

In this section we consider the case where the length of the time slot τ is predefined. In this case,

$$w(e) = \frac{f(e)}{c(e)}\frac{1}{\tau}$$

may not be integral. Therefore we set

$$w'(e) = \lceil \frac{f(e)}{c(e)}\frac{1}{\tau} \rceil$$

We again form the scheduling multi-graph, now with weights of $w'(e)$ on link $e \in E$.

LEMMA 6. *Let $\mathbf{f}$ be a link flow vector that satisfies the necessary conditions. If L is the chromatic index of the scheduling multi-graph where the link weight $w'(e)$ is given by*

$$w'(e) = \lceil \frac{f(e)}{c(e)}\frac{1}{\tau} \rceil$$

and τ is the length of a time slot, then

$$L\tau \leq \frac{3}{2}\left(1 + \tau.\Delta_G\right)$$

Proof:
Note that

$$\begin{aligned} \sum_{e \in N(v)} w'(e) &= \sum_{e \in N(v)} \lceil \frac{f(e)}{c(e)}\frac{1}{\tau} \rceil \\ &\leq \sum_{e \in N(v)} \left(\frac{f(e)}{c(e)\tau} + 1 \right) \\ &\leq \frac{1}{\tau} + \Delta_G \end{aligned}$$

Recall that Δ_G is the maximum degree in the original graph (not the scheduling multi-graph), representing the maximum number of neighbors of a node in the graph. By Theorem 2, $L \leq \frac{3}{2}\left(\frac{1}{\tau} + \Delta_G\right)$. □

This implies that in the case where the length of the time slots are fixed, if $f(e)$ is the desired link flow vector then

$$\frac{2}{3(1+\tau.\Delta_G)} f(e)$$

is achievable. In practical instances, the product $\tau.\Delta_G$ is usually very small, since slot-lengths are of the order of milliseconds, if not smaller, and Δ_G is typically less than 100. Hence the effects of fixed slot lengths can be ignored.

5. SINGLE PAIR MAXIMUM ACHIEVABLE RATE

A fundamental problem in network flows is the maximum flow problem. The objective for the maximum flow problem is to determine the maximum amount of flow that can be sent from a given source to a given destination in a capacitated network. We now consider the following problem.

- INPUT: A directed graph $G = (V, E)$ with a link speed $c(e)$ for $e \in E$ and two distinguished nodes s and d.
- OUTPUT: The maximum data rate, R_{sd} that can transmitted from s to d.

This quantity R_{sd} represents capacity of the node pair. The strategy that we use to determine R_{sd} for a given pair of nodes s and d is the following:

- We use the results developed in the last section to formulate necessary and sufficient conditions for a rate to be achievable.
- We get an upper bound on R_{sd} by solving a linear optimization problem over the *necessary conditions*.
- We use the scheduling multi-graph to get an achievable solution which is a lower bound on R_{sd}
- We show that this lower bound gets to within 67% of the optimal solution in the worst case guarantee and in practice typically gets to between 80-90% of the optimal solution.

The linear optimization problem that is solved in the second step is the key to determining bounds on R_{sd}. We first give a straightforward formulation for this problem with flow variables. We also give an alternate path-arc formulation that is amenable to the development of primal-dual algorithms for the solution of fully polynomial time approximation schemes. The reason for preferring the FPTAS to solving the linear program directly are the following:

- The FPTAS are very simple to implement and therefore can be potentially implemented at the nodes of the wireless networks.
- It is possible to trade-off the accuracy needed with the speed of solution. From our experiments, we observed that solving the linear programming problem approximately is enough to solve the routing-scheduling problem almost optimally.
- There is no need for a linear programming solver to solve the problem. (This is especially important if we have to implement the algorithm at the individual nodes).

With these objectives in view, we first give the necessary conditions for a given rate vector to be achievable between a given source destination pair.

THEOREM 7. *Given a graph $G = (V, E)$, two distinguished nodes $s, d \in V$ and a rate r between nodes s and d, the rate is achievable if there exist $y(e)$ for $e \in E$ that satisfies*

$$\begin{aligned}
\sum_{e:t(e)=s} y(e) &= r \\
\sum_{e \in N_{in}(v)} y(e) &= \sum_{e \in N_{out}(v)} y(e), \quad \forall v \neq s, d \\
\sum_{e \in N(v)} \frac{y(e)}{c(e)} &\leq \frac{2}{3}, \quad \forall v \in V
\end{aligned}$$

Proof:
The first constraint ensures that a data rate of r units is sent out of the source node. The second set of equalities ensure flow balance at the nodes in the network. The last set of inequalities are the sufficient conditions for a link flow vector to be achievable. □

The necessary condition for schedulability is given by replacing

$$\sum_{e \in N(v)} \frac{y(e)}{c(e)} \leq \frac{2}{3}, \quad \forall v \in V$$

with the inequalities

$$\sum_{e \in N(v)} \frac{y(e)}{c(e)} \leq 1, \quad \forall v \in V$$

The proof of the necessary and the sufficient conditions follow directly from the necessary and sufficient conditions for the achievability of a link flow vector. We now give an alternate path-arc formulation that is used to develop the FPTAS to solve this problem. Let $\mathcal{P}$ denote the set of simple paths between nodes s and d. (A path is defined to be simple if no nodes are repeated on the path). We let P denote a generic path in this path set $\mathcal{P}$. Let $x(P)$ denote the flow on path P.

Note that there can be an exponential number of paths between two given nodes in the network. Therefore, the path-arc formulation has an exponential number of variables. A given rate r is achievable, if there exists path flows $x(P)$ such that

$$\begin{aligned}
\sum_{P \in \mathcal{P}} x(P) &= r \\
\sum_{e \in N(v)} \frac{\sum_{P \in \mathcal{P}: P \ni e} x(P)}{c(e)} &\leq \frac{2}{3}, \quad \forall v \in V
\end{aligned}$$

If the factor $\frac{2}{3}$ is replaced by 1, the conditions are necessary. For a given pair of nodes s and d, if the objective is to determine the maximum achievable flow, R_{sd} from s to d then we do the following:

- Solve the following optimization problem:

$$r^* \;=\; \max \sum_{P\in\mathcal{P}} x(P)$$

$$\sum_{e\in N(v)} \frac{\sum_{P\in\mathcal{P}:P\ni e} x(P)}{c(e)} \le 1, \quad \forall v \in V$$

$$x(P) \ge 0, \quad \forall P \in \mathcal{P}$$

- Let $x^*(P)$ denote the optimal solution and let

$$f(e) = \sum_{P\in\mathcal{P}:P\ni e} x^*(P)$$

- Determine τ and $w(e)$ as shown in equation (2).
- Construct the scheduling multi-graph $G_S(\mathbf{f}, \tau)$ and determine its chromatic index L.
- Output $\frac{r^*}{L\tau}$

THEOREM 8. *Let R_{sd} be the maximum data rate that can be routed and scheduled between nodes s and d. Let r^* denote the optimal solution to the linear programming problem. Then*

$$\frac{2}{3}r^* \le \frac{r^*}{L\tau} \le R_{sd} \le r^*$$

Proof:
Since the conditions in the linear programming problem is only necessary, it implies that $R_{sd} \le r^*$. Note that $\frac{r^*}{L\tau}$ is an achievable (schedulable) flow. Therefore it is a lower bound on the maximum flow. Further, by the proof of Theorem 3, $L\tau \le \frac{3}{2}$ and the result follows. □

Therefore the maximum $s-d$ data rate that we determine is within 67% of the optimal solution in the worst case. We can evaluate the performance of the algorithm since r^* is an upper bound on the optimal solution value. We note that in practice, the algorithm consistently seems to get to about 85% of the upper bound, as shown in Section 8.

In case the time slots are of fixed length, then

$$\frac{2}{3(1+\tau.\Delta_G)}r* \le R_{sd} \le r*$$

5.1 Solving the Linear Programming Problem

In this section, we outline a FPTAS for solving the following optimization problem.

$$r^* \;=\; \max \sum_{P\in\mathcal{P}} x(P)$$

$$\sum_{e\in N(v)} \frac{\sum_{P\in\mathcal{P}:P\ni e} x(P)}{c(e)} \le 1, \quad \forall v \in V$$

$$x(P) \ge 0, \quad \forall P \in \mathcal{P}$$

We first write the dual to this optimization problem. In the dual, we associate a weight $w(v) \ge 0$ with node $v \in V$. The dual then is

$$\min \sum_{v\in V} w(v)$$

$$\sum_{e\in P} \frac{w(t(e)) + w(r(e))}{c(e)} \ge 1, \quad \forall P \in \mathcal{P}$$

$$w(v) \ge 0, \quad \forall v \in V$$

If we define the "length" of link as the ratio of the sum of the dual weights of its end nodes to the data rate of the link, then the dual constraint implies that the shortest $s-d$ path has to be greater than one. Given a $s-d$ path P and a node $v \in V$, let $I(v,P)$ denote the set of links in P that are incident on node v.

$$I(v,P) = \{e \in P : v = t(e) \text{ or } v = r(e)\}$$

Let

$$\theta(v,P) = \sum_{e\in I(v,P)} \frac{1}{c(e)}$$

Note that the maximum flow that can occur through node v on path P denoted by $f(v,P) = \theta(v,P)^{-1}$. This is due to the fact that the utilization of the node has to be less than one. Therefore an alternate way to write the dual constraint is

$$\sum_{v\in P} \theta(v,P)w(v) \ge 1, \quad \forall P \in \mathcal{P}$$

The primal-dual algorithm to solve this problem is the following: We initialize the weight of all the nodes to some precomputed value δ. The length of the link e in the network are computed as follows:

$$l(e) = \frac{w(t(e)) + w(r(e))}{c(e)}$$

Now the shortest path is computed from s to d using these link lengths. Let P^* denote the shortest path. We determine

$$m = \min_{v\in P^*} f(v, P^*)$$

This gives the flow that can be sent on the path without violating any of the node capacity bounds. (Note that we use the original capacity of one and not the residual capacity). A flow of m is sent on the path. The weight of the nodes in the optimal path is updated:

$$w(v) \leftarrow w(v)\left(1 + \frac{\epsilon m}{f(v,P^*)}\right), \quad \forall v \in P^*$$

Note that this is the same as

$$w(v) \leftarrow w(v)\left(1 + \epsilon\; m\; \theta(v,P^*)\right), \quad \forall v \in P^*$$

This process of finding the shortest path and updating the node weights is repeated until the sum of the node weights is greater than one. At this point, the flows in the network are scaled and this represents the ϵ-optimal solution to the linear programming problem. The algorithm is formally described below:

MAX_DATA_RATE

$w(v) = \delta \quad \forall v \in V$ and $r^* = 0$.

While $\sum_{v \in V} w(v)$
Set $l(e) = \frac{w(t(e))+w(r(e))}{c(e)}$
on each link $e \in E$ and compute shortest path P^* from s to d.
Let $m = \min_{v \in P^*} f(v, P^*)$
$r^* \leftarrow r^* + m$.
$f(v) \leftarrow f(v) + m\theta(v, P^*), \quad \forall v \in P^*$
$w(v) \leftarrow w(v)\left(1 + \frac{\epsilon c(P^*)}{f(v,P^*)}\right), \quad \forall v \in P^*$
end While
Compute $\rho = \max_{v \in V} f(v)$
Set $r^* \leftarrow \frac{r^*}{\rho}$
Output r^*

The analysis of the algorithm proceeds as in [10]. Since the constraints are on the nodes in the network, the analysis is slightly different.

Let $w_i(v)$ represent the weight of node v at the end of iteration i. Let $D(i) = \sum_{v \in V} w_i(v)$. This is the dual objective function at the end of iteration i. Let $\alpha(i)$ represent the length of the shortest path in iteration i and let P^i represent the shortest path in iteration i. Let L be an upper bound on the number of hops in any shortest path found during the course of the algorithm. In the worst case we can set $L = n$. Note that $\frac{D(i)}{\alpha(i)}$ is a feasible dual solution. Let f_i represent the flow that is sent from the source to the destination *until iteration i*. Then

$$\begin{aligned} D(i) &\leq \sum_e w_{i-1}(e) + \epsilon m \sum_{e \in P^i} w_{i-1}(v)\theta(v, P^i) \\ &= D(i-1) + \epsilon\,(f_i - f_{i-1})\,\alpha(i-1) \\ &\leq D(i-1) + \epsilon\,(f_i - f_{i-1})\,\alpha(i-1) \end{aligned}$$

Therefore

$$D(i) \leq D(0) + \epsilon \sum_{j=1}^{i} (f_j - f_{j-1})\,\alpha(j-1)$$

Let β be the dual optimal solution. We now want to bound $\frac{\beta}{f_t}$.

$$\beta = \min_w \frac{D(w)}{\alpha(w)} \leq \frac{D(w_i - w_0)}{\alpha(l_i - l_0)} \leq \frac{D(l_i) - D(l_0)}{\alpha(w_i) - \delta L}$$

Substituting the bound on $D(i) - D(0)$ we get,

$$\alpha(i) \leq \delta L + \frac{\epsilon}{\beta} \sum_{j=1}^{i} (f_j - f_{j-1})\,\alpha(j-1)$$

For any j note that $\alpha(j)$ is maximum when all these inequalities hold as equalities. This implies that

$$\alpha(i) \leq \delta L e^{\frac{\epsilon f_i}{\beta}} \qquad i = 1, 2, \ldots, t.$$

Since

$$1 \leq \alpha(t) \leq \delta L e^{\frac{\epsilon f_i}{\beta}}$$

Therefore,

$$\frac{\beta}{f_t} \leq \frac{\epsilon}{\log(\delta L)^{-1}}$$

LEMMA 9. *There is a feasible flow of value*

$$\frac{f_t}{\log_{1+\epsilon} \frac{(1+\epsilon)}{\delta}}$$

Proof:
Whenever the node utilization is increased by one unit, the weight of the node increases by at least $1 + \epsilon$. Note that $w_0(v) = \delta, w_{t-1}(v) < 1$ and $w_t(v) < (1 + \epsilon)$. The utilization of node v is at most $\log_{1+\epsilon} \frac{(1+\epsilon)}{\delta}$. Therefore, scaling the final flow f_t by $\log_{1+\epsilon} \frac{(1+\epsilon)}{\delta}$ creates a feasible flow. □

The ratio of the dual to the primal optimal solution γ is $\frac{\beta}{f_t} \log_{(1+\epsilon)} \frac{(1+\epsilon)}{\delta}$. Substituting the bound on $\frac{\beta}{f_t}$,

$$\gamma \leq \frac{\epsilon}{\ln(1+\epsilon)} \frac{\ln \frac{(1+\epsilon)}{\delta}}{\ln(\delta L)^{-1}}$$

Setting the value of $\delta = L^{-\frac{1}{\epsilon}}(1+\epsilon)^{1-\frac{1}{\epsilon}}$, we get

$$\gamma \leq \frac{\epsilon}{(1-\epsilon)\log(1+\epsilon)} \leq \frac{1+\epsilon}{(1-\epsilon)^2} \leq (1+2\epsilon)$$

THEOREM 10. *The running time of the algorithm is*

$$O(n\lceil \frac{1}{\epsilon} \log_{1+\epsilon} L \rceil T_{SP}).$$

where T_{SP} is the running time of one shortest path problem.

Proof:
At iteration i, the weight of the at least one node is increased by a factor of $1+\epsilon$. Since $w_t(e) < (1+\epsilon)$ the number of iterations in which e is the minimum capacity edge on the path set chosen is at most $\lceil \frac{1}{\epsilon} \log_{1+\epsilon} L \rceil$. Since there are n nodes, the total number of iterations is bounded by $n\lceil \frac{1}{\epsilon} \log_{1+\epsilon} L \rceil$. Each iteration involves solving one shortest path problem and the result follows. □

From experimenting, even setting ϵ as high as 0.3-0.4 seems to give good results for the maximum data rate problem. If we fix the value of ϵ, then the running time of the algorithm is roughly the same complexity as solving n shortest path problems.

6. ACHIEVABLE RATES FOR MULTIPLE SOURCE DESTINATION PAIRS

Now consider the problem of characterizing the achievable rates in the case of multiple source destination pairs. We assume that the traffic demand for different source-destination pairs is given in the form of a rate vector r. We assume that the rate vector has $K < n(n-1)$ components. Each source-destination pair between which there is a request will be termed a commodity. We use k to index the commodities. Let $s(k)$ represent the source node for commodity k and $d(k)$ the destination node for commodity k. Let $r(k)$ represent the flow that has to be routed from $s(k)$ to $d(k)$. The problem that we have to solve is the following:

- INPUT: A directed graph $G = (V, E)$ with a link speed $c(e)$ for $e \in E$ and K node pairs $(s(k), d(k))$ and associated with each node pair k is a desired rate $r(k)$.
- OUTPUT: A set of routes and associated schedule that achieves the given rates or declare the problem not achievable.

As in the case of the single source-destination flow problem, it is easy to show the following result.

THEOREM 11. *Given a graph $G = (V, E)$, with link speed $c(e)$ associated with link $e \in E$, K source destination pairs $(s(k), d(k))$ for $k = 1, 2, \ldots K$ with a desired flow rate $r(k)$ between $s(k)$ and $d(k)$. The rate vector r is achievable if there exists $y_k(e)$ such that*

$$\begin{aligned} \sum_{e:t(e)=s(k)} y_k(e) &= r(k), \quad \forall k \\ \sum_{e \in N_{in}(v)} y_k(e) &= \sum_{e \in N_{out}(v)} y_k(e), \quad \forall v \neq s, d, \quad \forall k \\ \sum_{e \in N(v)} \frac{\sum_k y_k(e)}{c(e)} &\leq \frac{2}{3}, \quad \forall v \in V \end{aligned}$$

Proof:
The proof follows from the arguments made for the single source-destination case. □

As in the previous case, if we replace

$$\sum_{e \in N(v)} \frac{\sum_k y_k(e)}{c(e)} \leq \frac{2}{3}, \quad \forall v \in V$$

by

$$\sum_{e \in N(v)} \frac{\sum_k y_k(e)}{c(e)} \leq 1, \quad \forall v \in V$$

the conditions are necessary. An alternate formulation of the above conditions can be given in an arc-path formulation. Let $\mathcal{P}_k$ represent the set of paths for the source-destination pair k. Consider a path $P \in \mathcal{P}_k$. Let $x(P)$ be the amount of flow sent on that path. This path leads from $s(k)$ to $d(k)$. ¿From the demand requirements, note that

$$\sum_{P \in \mathcal{P}_k} x(P) = r(k), \quad \forall k.$$

The total amount of flow on link e, represented by $f(e)$ is given by

$$f(e) = \sum_k \sum_{P \in \mathcal{P}_k : e \in P} x(P)$$

Then the necessary conditions for a rate vector r to be achievable is given by

$$\begin{aligned} \sum_{P \in \mathcal{P}_k} x(P) &= r(k), \quad \forall k \\ \sum_{e \in N(v)} \frac{\sum_k \sum_{P \in \mathcal{P}_k : P \ni e} x(P)}{c(e)} &\leq 1, \quad \forall v \in V \end{aligned}$$

Given a rate vector r, the strategy then is to solve for the x variables that satisfies the necessary conditions. If such a vector does not exist, then the given rate vector is not achievable. If it satisfies the necessary condition, then we determine the length of the time slot τ, form the scheduling multi-graph and determine its chromatic index L. Using the same techniques used above, it is not difficult to show that

$$\frac{r(k)}{L\tau}$$

is achievable. As in the previous cases, we show that in practice the performance of the algorithm is extremely good in practice.

6.1 Solving the Linear Programming Problem

In order to solve the linear programming problem, we first write the achievability problem as a concurrent flow problem and then use a primal-dual algorithm to solve this problem.

$$\max \lambda$$

$$\begin{aligned} \sum_{e \in N(v)} \frac{\sum_k \sum_{P \in \mathcal{P}_k : P \ni e} x(P)}{c(e)} &\leq 1, \quad \forall v \in V \\ \sum_{P \in \mathcal{P}_k} x(P) &= \lambda\, r(k), \quad \forall k \\ x(P) &\geq 0, \quad \forall P \in \mathcal{P}_k, \quad \forall k \end{aligned}$$

In the concurrent flow problem, the objective is to determine the maximum scaling factor λ^* that if all the desired traffic rates are scaled up by this factor, then it will still fit in the network. Therefore, if the objective function λ^* is less than than one then the vector is not achievable. If $\lambda^* \geq 1$, then we have to schedule the flow to determine if the flow is schedulable.

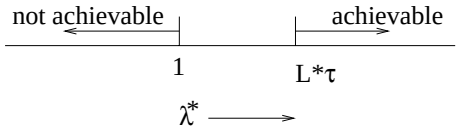

Figure 3: Schedulability of Flows

The largest flow vector that still satisfies the necessary constraints is $\lambda^* \mathbf{r}$ which is given by the optimal link flow vector $\mathbf{x}^*$, found from the solution to the LP above. Applying Lemma 3 to this flow, we get a schedule for a flow vector $\frac{\lambda^* \mathbf{r}}{L^* \tau}$, where L^* is the chromatic index of the scheduling multi-graph $G_S(\mathbf{x}^*, \tau)$. For this schedulable flow to be at least $\mathbf{r}$, we need $\lambda^* \geq L^* \tau$. This is shown in Figure 3. We have the following theorem.

THEOREM 12. *If $\lambda^* < 1$, then $\mathbf{r}$ is not schedulable. If $\lambda^* \geq L^* \tau$, then there exists a schedule for the flow $\frac{\lambda^* \mathbf{r}}{L^* \tau}$, and hence for $\mathbf{r}$. If $1 \leq \lambda^* \leq \frac{\lambda^* \mathbf{r}}{L^* \tau}$, then it is not known whether there exists a valid schedule for $\mathbf{r}$.*

The dual to this problem assigns a weight $w(v)$ to each node v in the network.

The dual formulation is as follows.

$$\min \sum_{v \in V} w(v)$$

$$\begin{aligned} \sum_{e \in P} \frac{w(t(e)) + w(r(e))}{c(e)} &\geq z(k), \quad \forall P \in \mathcal{P}_k, \ \forall k \\ \sum_{k=1}^{K} r(k) z(k) &\geq 1 \\ w(v) &\geq 0, \quad \forall e \in E \end{aligned}$$

DETERMINE_FEASIBILITY

$w(v) = \delta \quad \forall v \in V$ and $c = 0$

While $\sum_{v \in V} w(v) < 1$
For $k = 1, 2, \ldots, K$
  $r = d(k)$
  While $r > 0$
    Set $l(e) = \frac{w(t(e)) + w(r(e))}{c(e)}$
    Compute shortest path length from $s(k)$ to $t(k)$
    $z = \min_{P \in \mathcal{P}_k} w(P)$
    Let P^* be the optimal path.
    Let $u = \min_{v \in P^*} f(v, P^*)$.
    $\delta = \min\{r, u\}; \quad r \leftarrow r - \delta$
    $f(e) \leftarrow f(e) + \delta, \quad \forall e \in P^*$
    $w(v) \leftarrow w(v)\left(1 + \theta(v, P^*)\delta\right), \quad \forall v \in P^*$
  end While
end For
$c \leftarrow c + 1$
end While
Compute $\rho = \max_{v \in V} \sum_{e \in N(v)} \frac{f(e)}{c(e)}$
Output $\lambda^* = \frac{c}{\rho}$

The primal dual algorithm to solve the concurrent flow problem starts by assigning a precomputed weight of δ to all nodes v. The algorithm proceeds in phases. In each phase, for each commodity k, we route $r(k)$ units of flow from $s(k)$ to $d(k)$. A phase ends when commodity K is routed. The $r(k)$ units of flow from $s(k)$ to $d(k)$ for commodity k is sent via multiple iterations. In each iteration, the shortest path P^* from $s(k)$ to $d(k)$ is determined. Let $f(P^*)$ represent the maximum flow that can be sent on this path. We can send a flow of at most $f(P^*)$ units this iteration. Since $r(k)$ units of flow have to be sent for commodity k in each phase, the actual amount of flow sent is the lesser of $f(P^*)$ and the remaining amount of flow to make up $r(k)$ in this phase. Once the flow is sent, the weights of the nodes that carry the flow is increased. The algorithm is shown in Table 6.1. Therefore, the algorithm then alternates between sending flow along shortest path pairs and adjusting the length of the links along which flow has been sent until the optimal solution is reached.

By organizing the computation by source, one can send flows to multiple destinations at the same time as in [11] and the running time of the algorithm has only a logarithmic dependency on the number of source destination pair.

THEOREM 13. *The DETERMINE_FEASIBILITY algorithm computes a $(1-\epsilon)^{-3}$ optimal solution to the rate achievability problem in time $\mathcal{O}(\epsilon^{-2}m^2)$.*

7. EXTENSIONS AND VARIATIONS

The algorithmic scheme is fairly general and can be extended in many directions. We consider two of them here.

7.1 Link Transmissions with Interference

The first extension is to the case where in addition to the fact that a node cannot transmit or receive in the same time slot, there is interference between different link transmissions. The model is flexible enough to handle different kinds of interference between link transmissions and we illustrate one such case here. Our primary focus is to determine the upper bound on the data transmission rate. In the case of interfering channels, the coloring problems that have to be solved are different from the link coloring problems that we considered in the last few sections. We will comment on the coloring problems later in this section. The interference model that we consider is similar to the RTS-CTS scheme that is used in IEEE802.11 standard. We want the neighborhood of the transmitter and the receiver to be silenced during transmission. This is due to the fact that interference occurs at the receiver and the transmitter (because of the feedback from the receiver).

Consider a node u in the network. If this node is receiving or transmitting in a given time slot, then we want to ensure that the radio neighborhood of this node is silenced. In other words, if u is a receiver/transmitter in a given time slot then we would like nodes w in the neighborhood of u to not be transmitting to/receiving from u or any other node in this time slot. In order to make this notion more formal, we first define two node sets corresponding to given node $u \in V$. Let

$$V_{in}(u) = \{v \in V : t(e) = v \text{ for some } e \in N_{in}(u)\}$$

and

$$V_{out}(u) = \{v \in V : r(e) = v \text{ for some } e \in N_{out}(u)\}.$$

Therefore $V_{in}(u)$ represents the set of nodes that can transmit directly with u and $V_{out}(u)$ represents the set of nodes that u can directly transmit to. Therefore, $V(u) = V_{in}(u) \cup V_{out}(u)$ denotes all neighbors of u with whom it can communicate. When node u is transmitting/receiving in a given time slot, we want none of the nodes in $V(u)$ to be transmitting/receiving in that time slot. We cannot directly model this as a constraint in a linear programming formulation. We therefore use an alternate characterization to use in the linear programming formulation. We consider a pair of nodes u and w. In order to model the interference, we can write the following constraint.

$$\sum_{e \in N(u): V(w) \ni \{r(e), t(e)\}} \frac{y(e)}{c(e)} + \sum_{e \in N_{out}(w)} \frac{y(e)}{c(e)} \leq 1$$

Figure 4 illustrates the constraint.

We can write one such constraint for each pair of nodes in the network. These constraints will be enforced in addition to the constraints $\sum_{e \in N(v)} \frac{y(e)}{c(e)} \leq 1$ for all $v \in V$. In general, the kind of constraints that we have to enforce have

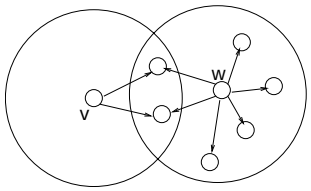

Figure 4: Constraint Set for Channels with Interference

the following structure then we can use the primal-dual algorithm developed in the last section to solve this problem: Let $S_1, S_2, \ldots, S_p$ represent p sets of links in the network. The constraint for the linear programming problem for necessary conditions has the following structure.

$$\sum_{e \in S_j} \frac{y(e)}{c(e)} \leq 1, \quad 1 \leq j \leq p$$

Each of these p sets has a dual weight associated with it. In each step of the primal-dual algorithm, the weight of link e is computed as the sum of the weights of the sets that the link belongs to divided by the capacity of the link. Once the shortest path and the amount of flow that has to be routed are determined, the weight of all the sets that contain a link in the shortest path are updated. Note that in determining the shortest path, the computation of the link weights is proportional to the number of sets that a link belongs to. Therefore the overall computation time will be longer than the case where there is no interference.

A detailed analysis and characterization of this and other interference models with respect to the joint routing and scheduling problem is provided in [15]. We do not give the details of the linear programming algorithm here since it is quite similar to the linear programming problem for the system with no interference. Once the linear programming problem is solved, then the flows have to be scaled and the coloring problem has to be solved. The coloring problem is not a simple link coloring problem. The coloring has to respect the interference constraint. See Ramanathan [14] for a thorough analysis of the coloring problems. However, there is no analog of Shannon's result and therefore, at this point we do not know the sufficient conditions even for scheduling a link flow vector. As stated earlier, the primal-dual scheme also carries over to the case where both the source and destination neighborhoods have to be silenced during transmissions as in IEEE 802.11. In this case the coloring problem is a distance-2 coloring and there is some literature for this problem [16].

7.2 Routing Resilient Flows

In the case where, in order to increase the reliability of the system, flows have to be routed from the source to the destination along two or more node disjoint paths. We can easily modify both our algorithms to take this into account. The main difference in the implementation, will be that instead of computing shortest paths we have to compute the shortest pair of disjoint paths between a given source node and destination node. This can be done via the algorithm of Suurballe and Tarjan [17] by doing two shortest path computations. The asymptotic running time of the algorithm is the same as in the case where there are no resiliency requirements.

8. SIMULATION RESULTS

In this section, we give some preliminary results on the performance of the routing-scheduling algorithms. The routing problem is solved using the primal-dual scheme with $\epsilon = 0.2 - 0.3$. The algorithm executes within a couple of seconds for all the problems considered. We tried both the $\frac{3}{2}$-approximation algorithm as well as 2-approximate greedy algorithm for the coloring problem. The performance of the greedy algorithm in practice was comparable to the $\frac{3}{2}$ approximation algorithm but greedy was much faster. The results show the performance of greedy. In all cases we plot the upper bound that is given by solving the linear programming problem with the necessary conditions as well as the achievable solution given by the coloring algorithm. In all cases we assumed that each time slot is 0.01 time units.

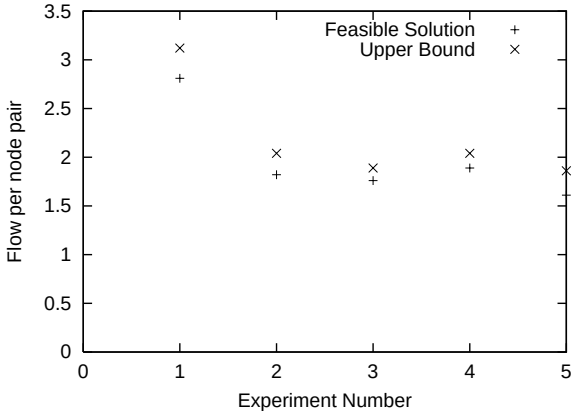

Figure 5: 10-node example

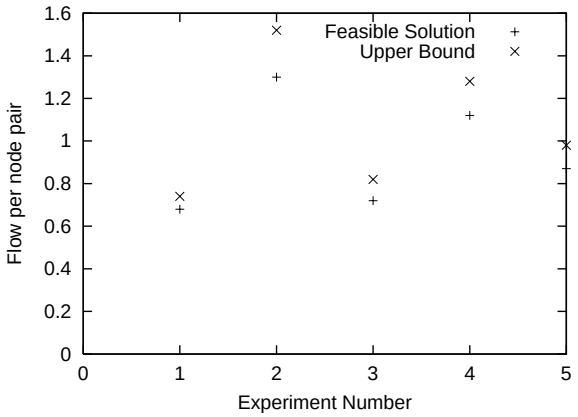

Figure 6: 20-node example

- Points were randomly distributed in a 10×10 square. All nodes within a distance of 5 units from a given node were assumed to have direct communication with the node. The link speed was normalized to 100 units. Each node has unit rate to send to every node in the network. The number of nodes were varied from 10-30 and the achievable throughput was plotted. (Figures 5,6,7).

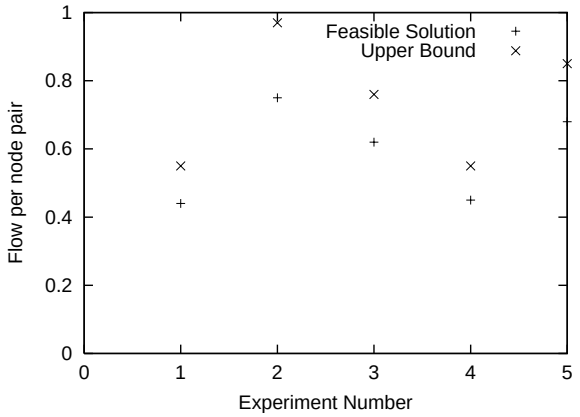

Figure 7: 30-node example

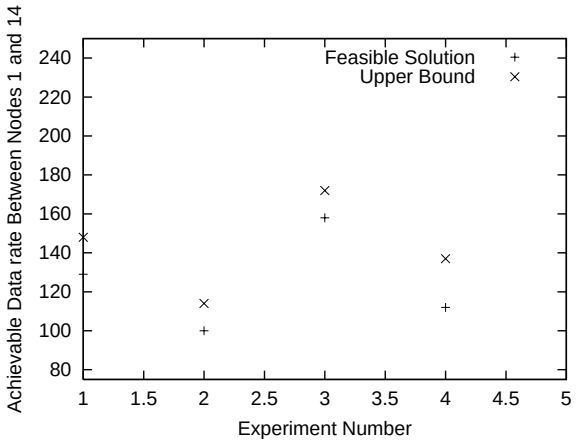

Figure 9: Nodes 1-14 Max Data Rate

- We considered a network with 15 nodes and 56 links (Figure 8) and we assumed that link capacities are uniformly distributed in the range [100 : 200]. We plotted the data rate that can be achieved between nodes 1 and 13 for 5 different experiments. (Figure 9).

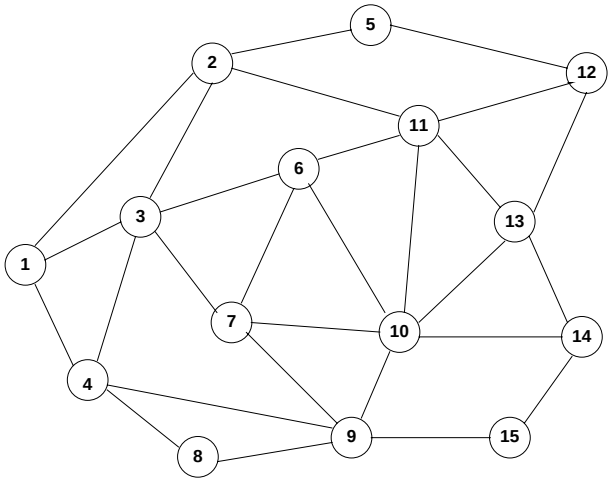

Figure 8: 15 Node Example

In all the examples that we studied, the achievable solution that we found was about 15 % from the LP upper bound. The approach that we use seems very promising and practical.

9. CONCLUSIONS

We studied algorithms for routing flows and scheduling transmissions in multi-hop wireless networks. We considered networks with only primary interference, where the only constraint on nodes is that each node can be communicating to at most one node at any time. The approach that we use is to develop tight necessary conditions and solve this as a linear programming problem. We developed FPTAS which are very simple to implement. The scheduling problem is solved as a coloring problem. We provide efficient coloring algorithms that guarantee 67 % of the optimal solution. In almost all of the cases that we tested, our approach is within 15-20 % of the optimal solution. This framework is a promising way to investigate other interference models and networks where nodes have multiple radios.

10. REFERENCES

[1] Baker, D.J., Wieselthier, J.E., and Ephremides, A., "A Distributed Algorithm for Scheduling the Activation of Links in a Self-Organizing Mobile Radio Networks", *IEEE Int. Conference Communications*, 1982, pp. 2F6.1-2F6.5.

[2] Hajek, B., and Sasaki, G., "Link Scheduling in Polynomial Time", *IEEE Transactions on Information Theory*, 34(5), pp. 910-917, 1988.

[3] Gupta, P., and Kumar, P.R., "The Capacity of Wireless Networks", *IEEE Transactions on Information Theory*, 46(2), pp. 388-404, 2000.

[4] Ahuja, R.K., Magnanti, T.L., Orlin, J.B., "*Network Flows: Theory, Algorithms, and Applications*", Prentice Hall, 1993.

[5] Grotschel, M., Lovasz, L., and Schrijver, A., "The Ellipsoid Method and its Consequences in Combinatorial Optimization", *Combinatorica*, 1(2), pp. 169-197, 1981.

[6] Post, M.J, Kershenbaum, A.S. and Sarachik, P.E., " Scheduling Multi-hop CDMA Networks in the Presence of Secondary Conflicts", *Algorithmica*, 1989, pp. 365-393.

[7] Wieselthier, J.E., Barnhart, C.M., and Ephremides, A., "A Neural Network Approach to Routing Without Interference in Multi-hop Networks", *IEEE Transactions on Communications*, 1994.

[8] Jain, K., Padhye, J., Padmanabhan, V., and Qiu, L., "Impact on Interference on Multi-hop Wireless Network Performance", *ACM Mobicom'03*, September 2003.

[9] Shahrokhi, F., and Matula, D., "The Maximum Concurrent Flow Problem", *Journal of the ACM*, 37, pp. 318-334, 1990.

[10] Garg, N., and Könemann, J., "Faster and Simpler Algorithms for Multi-commodity Flow and other Fractional Packing Problems", *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, pp.300-309, 1998.

[11] Karakostas, G., "Faster Approximation Schemes for Fractional Multi-commodity Flow Problems", *13th ACM/SIAM Symposium on Discrete Algorithms*, pp. 166-173, 2002.

[12] Shannon, C.E., "A Theorem on Coloring the Lines of a Network", *J. of Math. Physics*, 28, pp. 148-151, 1949.

[13] Nishizeki, T., and Kashiwagi, K., "On the 1.1 Edge-Coloring of Multi-graphs", *SIAM Journal of Discrete Math.*, 3(3), pp. 391-410, 1990.

[14] Ramanathan, S., "A Unified Framework and Algorithm for (T/F/C)DMA Channel Assignment in Wireless Networks", *IEEE Int. Conference on Communications*, 1991, pp. 7d.2.1-7d.2.8

[15] Kodialam, M., and Nandagopal, T., "The Effect of Interference on the Capacity of Multi-hop Wireless Networks", *Bell Labs Technical Report*, Lucent Technologies, July 2003.

[16] Barrett, C., Istrate, G., Anil Kumar, V.S., Marathe, M., and Thite, S., "Approximation Algorithms for Distance-2 Edge Coloring", Unpublished Document, 2002.

[17] Suurballe, J.W., and Tarjan, R.E., "A Quick Method for Finding Shortest Pairs of Disjoint Paths", *Networks*, 14, pp. 325-336, 1984.