Tractable Computation of Optimal Schedules and Routing in Multihop Wireless Networks

Peng Wang Stephan Bohacek
pwangee@udel.edu bohacek@udel.edu
Department of Electrical and Computer Engineering
University of Delaware

***Abstract*— Interference and collisions greatly limit the throughput of mesh networks that used contention-based MAC protocols such as 802.11. Significantly higher throughput is achievable if transmissions are scheduled. However, traditional methods to compute optimal schedules are computationally intractable (unless co-channel interference is neglected). This paper presents a tractable technique to compute optimal schedules and routing in multihop wireless networks. The resulting algorithm consists of three layers of optimization. The inner-most optimization computes an estimate of the capacity. This optimization is a linear or nonlinear optimization with linear constraints. The middle iteration uses the Lagrange multipliers from the inner iteration to modify the space over which the inner optimization is performed. This is a graph theoretic optimization known as the maximum weighted independent set problem. The outer-most optimization uses the Lagrange multipliers from the inner-most optimization to find optimal routes. This optimization requires solving several least cost paths problems and several maximum weighted independent set problems. The impact of optimal scheduling and routing is examined on realistic models of mesh networks where it is found that the capacity provided by optimal scheduling and routing is between a factor of 5 and 15 greater than that provided by 802.11's CSMA/CD with least hop routing.**

I. Introduction

802.11-based mesh networks are being deployed or planning to be deployed in over 300 cities [1]. One motivation for 802.11-based mesh networks is that mesh routers can be densely deployed with relatively low cost. A dense deployment of routers results in the typical mobile user being close to at least one router, allowing high data rate communication between the user and the router. However, a dense distribution of routers also results in significant interference. Due to the poor performance of CSMA/CD in environments with high interference, it is unclear if 802.11 with CSMA/CD will provide sufficient data rates for future mobile applications. An alternative to CSMA/CD is to schedule some fraction of the transmissions.

Achieving high capacity in the face of interference has been an active area of research for at least 25 years [2]. However, nearly 20 years ago it was shown that computing optimal schedules is *potentially* NP-complete [3]. On the other hand, it has never been shown that the cases where the capacity maximization is NP-complete actually arise in wireless networks. Notably, it has been proved that if there is no co-channel interference, then optimal schedules can be computed in polynomial time [4]. Co-channel interference arises when two nodes transmit simultaneously and, due to the interference, impede the ability of the receivers to correctly decode the messages. Under the assumption that co-channel interference does not arise, tremendous progress has been made (e.g., [4]–[14]). Unfortunately, with the dense deployment of mesh routers, co-channel interference is expected to be significant. Moreover, schedules generated under the assumption that co-channel interference does not arise, tend to perform quite poorly when there is co-channel interference (e.g., [15], [16]).

This paper presents tractable techniques for computing optimal schedules even when co-channel interference exists. On networks small enough, the optimality has been confirmed through exhaustive search. Furthermore, the ability to compute the schedules of a 500 node network densely covering downtown Chicago is demonstrated. There are two key theoretical results that underpin this approach.

- Letting L be the number of links in the network, a brute-force approach requires optimization over a space with 2^L elements. However, the optimal solution requires no more than L elements. If these L special elements were somehow known in advance, then the optimization could be performed over a space with L elements and the result would be identical to the one found by optimizing over the space of all elements.
- From a solution of the optimization problem over an arbitrary set of L elements, either
  - a new set of elements can be found that will improve the solution, which, in turn, leads to a better set of elements, and so on,
  - or, if no set of better elements exists, then the current set of elements is optimal.

Tractable computation of optimal schedules is required for tractable computation of optimal routing. However, in the worst case, the number of routes between a source-destination pair is exponential in the number of nodes. Thus, as an alternative to optimizing over all paths, an iterative scheme is developed that adds paths to a set of considered paths, where the paths added depends on the Lagrange multipliers from the computation of the optimal schedule.

The performance impact of optimal scheduling and routing is investigated on a realistic model of a mesh network in downtown Chicago that was developed with the UDel Models urban network simulator [17]. When compared to 802.11

with CSMA/CD, optimal scheduling typically increases the capacity by a factor of 4 to 12, depending on the density of the wired gateways. When compared to least hops routing, the impact of optimal routing ranges from 20% to 65%. Thus, as compared to 802.11 with CSMA/CD and least hop routing, the combined impact of optimal scheduling and routing ranges from a factor of 5 to 15.

The remainder of the paper proceeds as follows. In the next section, the system model, notation, and problem definition are given. Optimal scheduling is discussed in Section III while optimal routing is described in Section IV. Numerical experiments on scheduling are discussed in Section III-D and numerical experiments on optimal scheduling and routing are given in Section V. Concluding remarks are given in Section VI. Proofs can be found in the Appendix.

II. System Model and Problem Formulation

A router-to-router connection is denoted by ϕ, with Φ denoting the set of all such connections. A connection may make use of several flows with each flow using a different path; the kth flow for connection ϕ is denoted (ϕ, k). The data rate of flow (ϕ, k) is denoted $f_{\phi,k}$, and the path followed by flow (ϕ, k) is denoted $P(\phi, k)$. The set of paths used by connection ϕ is denoted with $\mathbf{P}(\phi)$, i.e., $\mathbf{P}(\phi) = \{P(\phi, k) : k = 1, ..., |\mathbf{P}(\phi)|\}$, where $|\mathbf{P}(\phi)|$ is the number of paths used by connection ϕ. The set of all considered paths is $\mathcal{P}$. Using this notation, the total data rate over connection ϕ is $\sum_{k=1}^{|\mathbf{P}(\phi)|} f_{\phi,k}$, and the total data rate sent over link x is $\sum_{\{(\phi,k)|x\in P(\phi,k)\}} f_{\phi,k}$, where $\{(\phi, k) | x \in P(\phi, k)\}$ is the set of flows that cross link x. All links are directional.

We define an *assignment* to be a vector $\mathbf{v} = \begin{bmatrix} v_1 & \cdots & v_L \end{bmatrix}$, where there are L links in the network and where $v_x \in \{0, 1\}$ with $v_x = 1$ implying that link x is transmitting during assignment v. It is possible to extend this approach to accommodate links with multiple bit-rates and multiple transmit powers. The *set of considered assignments* is denoted by $\mathcal{V}$, while the *set of all assignments* is denoted $\overline{\mathcal{V}}$. In the simple case where $v_x \in \{0, 1\}$, $\overline{\mathcal{V}}$ has 2^L assignments. The size of $\overline{\mathcal{V}}$ is the main challenging facing optimal scheduling. Thus, typically, $\mathcal{V} \subsetneq \overline{\mathcal{V}}$.

The data rate across link x during assignment v is denoted by $R(v, x)$. In general $R(v, x)$ is a complicated function. However, here a simple binary relationship is used to define $R(v, x)$. Specifically,

$$R(v,x) = \begin{cases} R_x \text{ if } v_y = 0 \text{ for all } y \in \chi(x) \\ 0 \text{ otherwise} \end{cases}, \quad (1)$$

where $\chi(x)$ is a set of links that conflict with x, i.e., $y \in \chi(x)$ if simultaneous transmissions over x and y are not possible. R_x is the nominal data rate over link x. Note that this definition of $R(v, x)$ neglects the possibility that the aggregate interference from transmissions from several links not in $\chi(x)$ can result in a transmission failure over link x. However, as discussed in [18], such problems can easily be addressed. All computations in this paper use this technique, and hence the computed capacities account for multiple interferers.

The set of conflicting links, $\chi(x)$, depends on the communication model. Arguably, the $SINR$ protocol communication model is the most relevant and is the model that is used in this paper. Let $SINR(x, y)$ be the SINR at the receiver of link x when link y is also transmitting. Then, the SINR model specifies that $y \in \chi(x)$ if $SNIR(x, y) < T(x)$ or $SINR(y, x) < T(y)$, where $T(x)$ and $T(y)$ are thresholds that depend on the modulation schemes.

A schedule is a convex combination of assignments. Specifically, a schedule is a set $\{\alpha_v : v \in \mathcal{V}\}$ where $\sum_{v\in\mathcal{V}} \alpha_v \leq 1$ and $\alpha_v \geq 0$. With this notation, the total data rate that the schedule $\boldsymbol{\alpha}$ provides over link x is $\sum_{v\in\mathcal{V}} \alpha_v R_x v_x$. Finally, the capacity optimization problem is

$$\max_{\boldsymbol{\alpha},\mathbf{f}} G(\mathbf{f}) \quad (2a)$$

subject to:

$$\sum_{\{(\phi,k)|x\in P(\phi,k)\}} f_{\phi,k} \leq \sum_{v\in\mathcal{V}} \alpha_v R(v,x) \text{ for each link } x \quad (2b)$$

$$\sum_{v\in\mathcal{V}} \alpha_v \leq 1 \quad (2c)$$

$$0 \leq \alpha_v \text{ for each } v \in \mathcal{V}, \quad (2d)$$

where $\mathbf{f}$ is the vector of flow rates. The function G is referred to as the *capacity metric*. Several different capacity metrics are possible. In some cases, the capacity metric is the same as the network utility and $G(\mathbf{f}) = \sum_{\phi\in\Phi} U_\phi(\sum_k f_{\phi,k})$, where U_ϕ is the utility function for connection ϕ. Popular utility functions include $U_\phi(f) = w_\phi \log(f)$ [19]–[21] and $U_\phi(f) = w_\phi f^{1-\gamma}/(1-\gamma)$ [22], where w_ϕ are administrative weights. Another widely used capacity metric is $G(\mathbf{f}) = \min_{\phi\in\Phi} w_\phi \sum_k f_{\phi,k}$ [23]. Note that when a utility-based capacity metric is used, (2a) is a nonlinear optimization with linear constraints. We assume that U_ϕ is continuously differentiable, is concave, and increasing. In this case, (2a) is a convex optimization. the solvability of such a problem is detailed in [24]. If $G(\mathbf{f}) = \min_{\phi\in\Phi} w_\phi \sum_k f_{\phi,k}$, then (2a) can be written as a linear programming problem, which is extensively studied in [25].

In theory, Problem (2a) is solvable. However, there are two computational challenges. First, if $\mathcal{V}$ is the set of all assignments, then the vector $\boldsymbol{\alpha}$ has 2^L elements. Second, if all paths are considered, then the number of elements in the vector $\mathbf{f}$ might be exponential in the number of nodes in the network. Thus, the size of the space over which the optimization is performed must be reduced. This idea of considering a reduced space was considered in [23] and [26], however, the space was constructed arbitrarily. In this paper the space is constructed so that the capacity found by optimizing over the reduced space is the same capacity found by optimizing over the entire space. The next section focuses on the scheduling problem while Section IV focuses on the routing problem.

III. Optimal Scheduling

A. Introduction

The objective of this section is to compute optimal schedules by optimizing over a set of considered assignment $\mathcal{V} \subsetneq \overline{\mathcal{V}}$. The key questions are 1) is it possible to reduce the size of $\mathcal{V}$ without impacting the solution, and 2) if so, how can the set of considered assignments be constructed so that the value of (2a) with the reduced sized $\mathcal{V}$ is the same or near to the value when $\mathcal{V} = \overline{\mathcal{V}}$? The answer to the first question is provided next and the following subsections focus on the second question.

Proposition 1: Proposition 2: There exists $\mathcal{V}$ with L assignments such that the solution to (2a) is the same as the solution to (2a) when $\mathcal{V} = \overline{\mathcal{V}}$

This result, which follows from Caratheodory's Theorem (e.g., Theorem B.6 in [24]), implies that the optimal schedule can be found by considering a set, $\mathcal{V}$, that is relatively small.

B. Considered Assignments

1) Basics: It is well known that Lagrange multiplier theory can be applied to Problem (2a) (e.g., see [24]). Specifically, associated with each link constraint (2b) is a Lagrange multiplier denoted μ_x. Similarly, associated with the constraint (2c) is a Lagrange multiplier denoted λ. Employing the economic interpretation of Lagrange multipliers, μ_x can be interpreted as the price/bit of sending data over links x, or from the network's point of view, μ_x is the revenue that is collected for each bit that crosses link x. Under this interpretation, the revenue generated by assignment v is

$$\sum_{x=1}^{L} R\left(v,x\right)\mu_x.$$

The multiplier λ can be interpreted as the maximum revenue generated by any assignment in V. Specifically,

Proposition 3: Under the conditions given in Section II,

$$\lambda = \max_{v\in\mathcal{V}} \sum_{x=1}^{L} R\left(v,x\right)\mu_x. \tag{3}$$

Typically, there are many assignments in $\mathcal{V}$ that generate revenue λ. The set of such assignments is referred to as the set of active assignments and is denoted $\mathcal{V}^*$, i.e.,

$$\mathcal{V}^*\left(\mu\right) := \left\{ v : \sum_{x=1}^{L} R\left(v,x\right)\mu_x = \max_{v\in V} \sum_{x=1}^{L} R\left(v,x\right)\mu_x \right\}. \tag{4}$$

The reason that $\mathcal{V}^*$ is referred to as the set of active assignments is that the optimal schedule multiplexes between assignments that are in $\mathcal{V}^*$.

Proposition 4: If $v \notin V^*$, then $\alpha_v = 0$.

Since the revenue generated by active assignments is λ, the optimal schedule also generates revenue λ. Specifically, let R_x^* be the optimal data rate across link x, that is

$$R_x^* := \sum_{v\in V} \alpha_v^* R\left(v,x\right), \tag{5}$$

where α^* specifies the optimal schedule. Then, it can easily be shown that

$$\lambda = \sum_{x=1}^{L} R_x^* \mu_x.$$

2) Evaluating candidate assignments: A brute force approach to constructing a good set of assignments is to start with an arbitrary set of assignments, $\mathcal{V}$, select an assignment $v^+ \notin \mathcal{V}$, and evaluate the resulting capacity with the set of assignments $v^+ \cup \mathcal{V}$. However, this approach is computationally complex in that (2a) must be repeatedly solved. Furthermore, it is not clear if the utility of v^+ is only apparent when it is added to $\mathcal{V}$ along with a particular set of other assignments. Alternatively, the question of whether an assignment $v^+ \notin \mathcal{V}$ will increase the capacity when the set of considered assignments is changed from $\mathcal{V}$ to $v^+ \cup \mathcal{V}$ is answered by the following theorem.

Theorem 5: For the set of assignments $\mathcal{V}$, let $\boldsymbol{\mu}$ and λ be the Lagrange multipliers associated with constraints (2b) and (2c) when (2a) is solved with this $\mathcal{V}$. Now consider an assignment $v^+ \notin \mathcal{V}$, The capacity provided by $v^+ \cup \mathcal{V}$ is greater than that provided by $\mathcal{V}$ if and only if

$$\sum_{x=1}^{L} R\left(v^+,x\right)\mu_x - \lambda > 0. \tag{6}$$

Corollary 6: If $\mathcal{V}$ is such that the Lagrange multipliers that result from optimizing over $\mathcal{V}$ are such that no assignment exists that satisfies (6), then the schedule found by optimizing over $\mathcal{V}$ is optimal.

Theorem 5 provides the main tool for constructing a good set of assignments. Several comments are in order.

- Employing an economic interpretation of the Lagrange multipliers, Theorem 5 implies that an assignment v^+ will increase the utility if it generates more revenue per second than any other assignment in the set $\mathcal{V}$.
- In the proof of Theorem 5, it is seen that a linear approximation of the improvement of the capacity is given by $\sum_{x=1}^{L} R\left(v^+,x\right)\mu_x - \lambda$. Therefore, if several assignments are found that satisfy (6), then it is reasonable to only add the assignment that maximizes $\sum_{x=1}^{L} R\left(v^+,x\right)\mu_x$. Numerical experiments have confirmed this behavior.
- Corollary 6 provides a means to determine whether the current schedule is optimal. For example, it is possible to evaluate (6) quite quickly. Hence, for moderate values of L, it is possible to evaluate (6) for all assignments. In this way, an exhaustive search can be performed in order to verify whether a schedule is optimal. However, with today's computational abilities, this approach cannot be applied for networks with more than 23 links.

3) Algorithm to Maximize the Capacity: Based on Theorem 5, Algorithm 1 can be used to iteratively add assignments to $\mathcal{V}$ such that the added assignment satisfies (6). The initial set of assignments can be found using a greedy approach given by Algorithm 2. In our computational experiments, Algorithm 1 converges after $O\left(L\right)$ iterations. Therefore, solving the optimization (2a) over $\mathcal{V}$ can be accomplished with relatively low

Algorithm 1 Computing an Optimal Schedule

1: Select an initial set of assignments $V(0)$, set $k = 0$.
2: Solve (2a) for $\mathcal{V} = \mathcal{V}(k)$ and compute $\mu(k)$ and $\lambda(k)$, the Lagrange multipliers associated with constraints (2b) and (2c), respectively.
3: Search for an assignment $v^* \notin \mathcal{V}(k)$ such that

$$\sum_{x=1}^{L} \mu_x(k) R(v^*, x) > \lambda(k). \quad (7)$$

4: **if** such an assignment is found **then**
set $\mathcal{V}(k+1) = V(k) \cup v^*$, set $k = k+1$, and go to Step 2.
5: **else**
if no such assignment *exists*, then stop, the optimal schedule has been found.
6: **end if**

Algorithm 2 Selecting an Initial Set of Assignments

1: Set $\mathcal{V} = 0$.
2: Start an assignment w with $w_x = 0$ for all x.
3: Select a link x such that $v_x = 0$ for all $v \in \mathcal{V}$. Set $w_x = 1$.
4: Randomly select a link y such that $v_y = 0$ for all $v \in \mathcal{V}$ and $y \notin \bigcup_{\{x|w_x=1\}} \chi(x)$.
5: **if** such a y exists **then**
set $w_y = 1$ and go to Step 4.
6: **else**
set $\mathcal{V} = \mathcal{V} \cup w$.
7: **end if**
8: **if** for all x there exits a $v \in \mathcal{V}$ such that $v_x = 1$ **then**
stop. $\mathcal{V}$ is the set of initial assignments.
9: **else**
go to Step 2
10: **end if**

computational complexity. On the other hand, it is unclear how to find assignments that satisfy (6). This topic is investigated in the next section.

C. Finding New Assignments

Algorithm 1 changes the challenge of solving (2a) over a large set of assignments to the challenge of finding assignments that satisfy (6). According to the discussion after Theorem 5, the best assignment to add to $\mathcal{V}$ is the one that solves

$$\max_{v} \sum_{x=1}^{L} R(v, x) \mu_x. \quad (8)$$

As will be shown next, solving this maximization is equivalent to finding the maximum weighted independent set of the weighted conflict graph.

The utility of the conflict graph for finding schedules has been demonstrated in several previous works (e.g., [3], [23]). The conflict graph is constructed as follows. Each link in the network induces a vertex in the conflict graph. Thus, a link x in the network is associated with a vertex in the conflict graph; this vertex is denoted with x, where whether x refers to a link in the network or a vertex in the conflict graph is clear from the context. There is an edge between vertices x and y if $y \in \chi(x)$, where, as discussed in Section II, x and links in $\chi(x)$ cannot simultaneously transmit. The weighted conflict graph is constructed by assigning the weight $R_x \mu_x$ to vertex x, where R_x is the nominal data rate across link x and μ_x is the Lagrange multipliers associated with constraint (2b).

An independent set (or stable set) of a graph is a set of vertices where no two vertices in the set are neighbors. Letting I be an independent set, the weight of I is the sum of the weights of the vertices in I. Thus, an independent set of the conflict graph is a set of links that are not in conflict and hence able to transmit simultaneously. Therefore, if I is an independent set and $v(I)$ is the assignment generated by I via $v_x(I) = 1$ is $x \in I$, then under assignment $v(I)$ the data rate across link x is R_x. Furthermore, the weight of I is $\sum_{x \in I} R_x \mu_x$. By (1), $\sum_{x \in I} R_x \mu_x = \sum_{x=1}^{L} R(v, x) \mu_x$. Thus, the goal of solving (8) is the same as finding the maximum weighted independent set (MWIS).

Unfortunately, in the worst case, finding the MWIS is NP-hard. On the other hand, the MWIS problem has been extensively studied. For example, it is known to be solvable in polynomial time for many classes of networks including perfect graphs [27], interval graphs (which arise when a wireless network is confined to a road) [27], disk graphs [28], claw-free graphs [29], fork-free graphs [30], trees [31], sparse random graphs [32], and circle graphs [33]. Moreover, there has been extensive work on approximating the MWIS (see [34] for a review) and specialized algorithms have been developed for exactly computing a MWIS [35], [36], [37], [38]–[40], [41]. However, after evaluating several alternative approaches, we have found that a generic integer linear programming (ILP) solver provides the fastest way to find a MWIS. For example, the MWIS for a 500 link network is computed in approximately 250 msec on a PC with a 2.4MHz AMD FX-53 processor. The MWIS problem can be formulated as an ILP via

$$\max_{v} \sum_{x=1}^{L} R_x \mu_x \mathbf{v}_x \quad (9a)$$

$$\text{subject to: } v_x + v_y \leq 1 \text{ if } y \in \chi(x) \quad (9b)$$

$$v_x \in \{0, 1\}.$$

However, in large networks, there are many constraints (9b). The computation time can be dramatically improved if a clique decomposition is used. Specifically, a set of cliques $\{Q_i,\ i = 1, 2, ...M\}$ are found such that if $y \in \chi(x)$, then there is a clique Q_i such that $x \in Q_i$ and $y \in Q_i$. Then,

Problem (9a) becomes

$$\max_{v} \sum_{x=1}^{L} R_x \mu_x v_x$$
$$\text{subject to: } \sum_{x \in Q_i} v_x \leq 1 \text{ for } i = 1, 2, ..., M$$
$$v_i \in \{0, 1\}.$$

While an optimal clique decomposition might further improve the computation time, a simple greedy clique decomposition results in a factor of ten improvement over (9a).

Remark 7: This paper focuses on the SINR binary-conflict communication model. If co-channel interference does not arise, then the node exclusive model can be used. Under the node exclusive model, $y \in \chi(x)$ if link's x transmitter or receiver are the same as y's transmiter or receiver. In this case, the conflict graph is a line-graph, and the MWIS problem decomposes to a maximum weighted matching (WMM) problem [27]. The WMM problem is considerably less computationally complex than the MWIS. For example, in the general case, polynomial complexity algorithms exits (e.g., see [42] for an $O\left(N \cdot L + N^2 \log(N)\right)$ complexity algorithm where N is the number of nodes and L is the number of links). In the case of a tree (which is typical for today's mesh networks), optimal matching are possible with simple greedy algorithms and message passing algorithms [43]. In the general case, simple approximation is also possible since maximal matching are within a factor of two of optimal [44]. There exists distributed algorithms with approximation ratios of $1/2$ [45], [46] and $2/3$ [47]. However, since the node exclusive model neglects co-channel interference, the schedules produced under this model perform poorly when applied to actual networks where co-channel interference exists.

D. Numerical Experiments in Optimal Scheduling

1) Experimental Set-Up: As discussed above, determining the optimal capacity has a theoretical worst-case computational complexity that makes computing capacity intractable for even small networks. However, the theoretical worst-case performance provides little insight into the typical performance that occurs in mesh networks. Thus, it is imperative that the performance be examined in realistic mesh networks. To this end, the UDel Models [17] were employed. Along with a realistic mobility simulator, the UDel Models includes a map builder, a realistic propagation simulator, and large collection of data and trace files. The propagation simulator is based on ray-tracing and accounts for reflections off of the ground and off of buildings, transmission through building walls, and diffraction around and over buildings [48]. It also accounts for the impact that different materials have on reflections off of walls and transmission through walls. Data sets for several urban areas is available online.

For this investigations, two types of mesh network topologies were investigated. One class of topologies was generated from $6x6$ city block regions of downtown Chicago. The $6x6$ city block regions were randomly located in the 2 km^2 region. Various nodes densities were investigated. Specifically, the number of gateways was 1, 2, 3, 6 and the number of wireless routers was 18, 36, 54, 72, and 90. The wireless routers and gateways were uniformly distributed throughout the $6x6$ city block region. Ten samples of each topology were generated (hence, 200 topologies in total). Besides these, another 100 topologies were examined (with 4 and 5 gateways). The results from these topologies can be interpolated from the behaviors from the other numbers of gateways, hence, these results are not included. Along with these random topologies a single large topology was considered. This topology had 500 nodes and 12 gateways and covered a 2 km^2 region of Chicago. This is currently the largest mesh network that the UDelModels has generated. Furthermore, it is larger than the largest currently deployed mesh network[1].

In these experiments, all traffic flow from gateways to destinations (i.e., downstream traffic), where each mesh router in the topology was a destination of a flow. The routing was a single path least hop routing, where each link had a receiver signal strength of at least 75 dBm. Among paths with the same number of hops, the path selected was the one that had the highest minimum link channel gain, where the minimization is over each hop along the path. Each flow originates at the gateway such that the best route from the gateway to the destination of the flow is no worse than any route from any other gateway in terms of the minimum channel gain along the route. In Section V, the performance of optimal multipath routing is examined. In that case, the single path routing described above is used as a baseline.

Finally, 802.11a data rates were used. Specifically, for each link, the SNR was evaluated. This SNR yields a link data rate based on 802.11a relationship between data rates and SNR. Furthermore, it is assumed that 802.11a relationship between SNR and data rate is the same as the relationship between SINR and data rate.

2) Results from Numerical Experiments: Figure 1 shows how, in the 500 node topology, the capacity increases as the more assignments are added. The plots end when the integer programming failed to find a better assignment. Since the integer programming yields a MWIS, if a new assignment is not found, then the schedule is optimal. Thus, in the 500 node network, the optimal schedule was found after 117 iterations when the capacity is in terms of $\sum_{\phi \in \Phi} \log(f_\phi)$ and after 225 iterations when the capacity is $\min_{\phi \in \Phi} f_\phi$.

Figure 1 shows the average number of iterations required for convergence for the 6×6 block topologies. As expected, as more routers are added and as more gateways are added the topology becomes more complex and more iterations are required for convergence. For a fixed number of gateways, it appears that the number of iterations increases linearly with number of nodes in the network. Moreover, the number of iterations is rather small. For example, in all cases but 6 gateways with the $\max_{\phi \in \Phi} f_\phi$ metric, the number of iterations

[1]The largest mesh network currently deployed is Corpus Christi with 300 mesh routers [49].

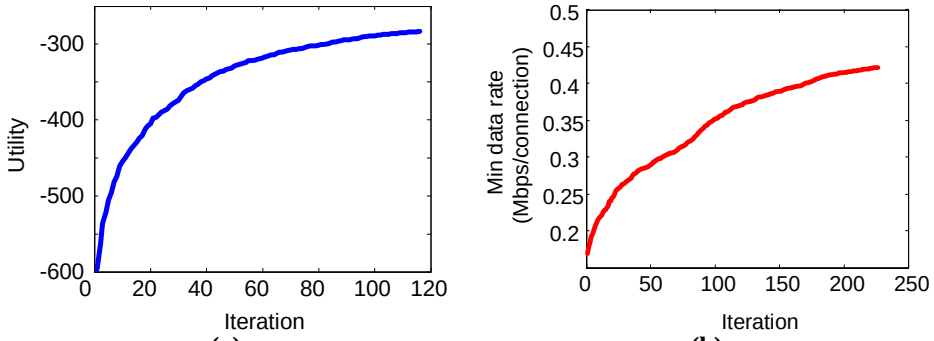

Fig. 1. Variation in the computed capacity as assignments are added. In (a) the capacity is the total utility, i.e., $\sum_{\phi\in\Phi}\log(f_\phi)$. In (b) the capacity is $\min_{\phi\in\Phi} f_\phi$. These plots are for the 500 node topology.

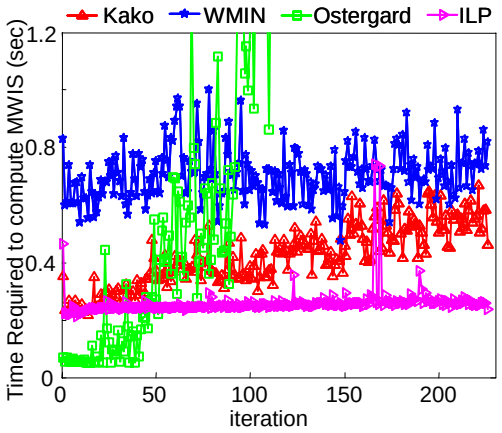

Fig. 3. For each iteration of Algorithm 1, a MWIS must be computed. The above shows the time required to compute the MWIS at each iteration for several different techniques to compute MWIS.

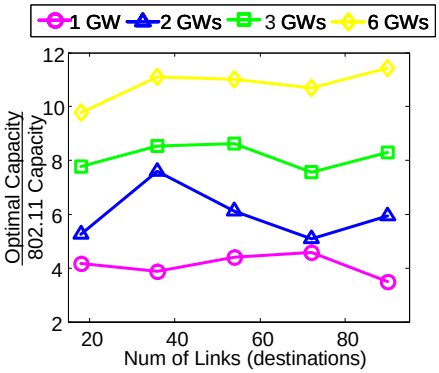

Fig. 4. Comparison between optimal scheduling and 802.11 in mesh networks covering 6×6 block regions of downtown Chicago.

required for convergence is less than the number of links in the network. Note that small number of iterations to reach convergence is partly due quality of the initial assignments found by Algorithm 2.

Another important aspect of the performance of Algorithm 1 is the time that each iteration takes, specifically, the time it takes to compute a MWIS. Figure 3 shows the time require to compute the MWIS for each iteration for several methods to compute MWIS. Kako's method is an approximate method that was presented in [50]. WMin is another approximate methods presented in [51]. Ostergrad's method was presented in [41] and is regarded as one of the top performing methods to compute MWIS [40]. Finally, the ILP with a clique decomposition discussed in Section III-C. Here the CPLEX [52] optimization package was used for ILP. From Figure 3, it is clear that ILP finds the MWIS the fastest and takes about 250 msec to find a MWIS for a 500 node mesh network.

With the ability to compute optimal schedules, the impact of optimal schedules on the capacity as compared to 802.11 with CSMA/CD can be investigated. Figure 4 shows the ratio of the optimal capacity to the capacity that 802.11 CSMA/CD can achieve. Here the capacity metric is $\min f_\phi$. Qualnet was used to estimate the capacity of 802.11. Here RTS/CTS and Qualnet's automatic rate fallback scheme were used. The 802.11 CSMA/CD capacity was determined by sending data to each destination at a constant rate (1000 B packets were used). The sending rate was adjusted until the maximum of $\min_{\phi\in\Phi} f_\phi$ was found. Confidence intervals were generated via bootstrapping [53] to ensure that the estimated capacity was accurate within 10%.

Figure 4 shows that for networks with a large number of gateways, optimal scheduling can have a dramatic improvement in the capacity. On the other hand, when there is only one gateway in a 6×6 block region, only is factor of four improvement is observed. While a factor of four is large, in this comparison, a standard version of 802.11 CSMA/CD was used. It is conceivable that if 802.11 is better tuned (e.g., by tuning CCA [54]) and a better virion of the ARF is used, that the performance of 802.11 could be improved.

IV. Optimal Routing

With tractable computation of optimal schedules, it is possible to consider optimal routing. A naive approach to optimal routing is to let each connection ϕ to use all possible routes. That is, for each ϕ, $\mathbf{P}(\phi) = \{P(\phi,k) : k = 1, 2, \ldots |\mathbf{P}(\phi)|\}$ is the set of all possible paths between the source and destination of connection ϕ. With this set of paths, the optimal schedule can be found. The schedule will determine the portion of the flow $\sum_{k=1}^{|\mathbf{P}(\phi)|} f_{\phi,k}$ that uses path $P(\phi,k)$. In this way,

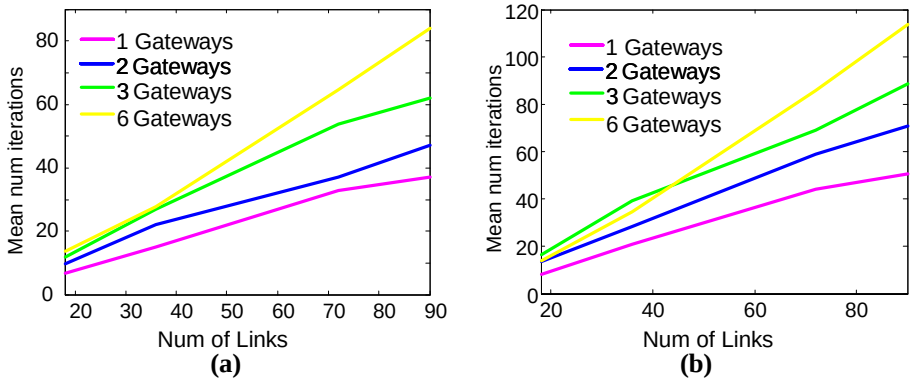

Fig. 2. Mean number of iterations until convergence for different topologies in different 6×6 block regions of downtown Chicago . In (a), the capacity is $\sum_{\phi\in\Phi}\log\left(f_\phi\right)$ while in (b) the capacity is $\min_{\phi\in\Phi} f_\phi$

optimal scheduling also performs optimal routing. While such an approach is possible in theory, in practice, it is not tractable since the number of possible paths between a source and destination grows exponentially with the size of the network.

As an alternative, we consider adding paths only when they will improve the performance. Specifically, suppose that connection ϕ uses the set of paths $\mathbf{P}\left(\phi\right)$. Now consider the candidate path $P\left(\phi,k^+\right)\notin\mathbf{P}\left(\phi\right)$. This path should be added to the set of considered paths only if the addition of this path improves the capacity. The question of whether including the path $P\left(\phi,k^+\right)$ into the set of considered paths will increase the network capacity is answered by the following.

Proposition 8: Let $P\left(\phi,k^+\right)\notin P\left(\phi\right)$ be the path of a candidate path potentially to be added to $P\left(\phi\right)$. Furthermore, let $P\left(\phi,k^o\right)\in P\left(\phi\right)$ be such that $f_{\phi,k^o}>0$. Then, adding the candidate path $P\left(\phi,k^+\right)$ to $P\left(\phi\right)$ will increase the capacity if and only if

$$\sum_{x\in P(\phi,k^+)}\mu_x<\sum_{x\in P(\phi,k^o)}\mu_x, \tag{10}$$

where μ_x is the Lagrange multiplier associated with constraint (2b) that arises from solving Problem (2a) without the inclusion of the path $P\left(\phi,k^+\right)$.

Employing the economic interpretation of Lagrange multipliers, Proposition 8 says a flow that uses path $P\left(\phi,k^+\right)$ should be included into the set of considered paths if its end-to-end cost $\sum_{x\in P(\phi,k^+)}\mu_x$ is less than the end-to-end cost of the currently used paths.

Following the approach of the proof of Proposition 8, it is can be shown that data is only sent down paths where the cost is minimum.

Proposition 9: If $\sum_{x\in P(\phi,k_1)}\mu_x<\sum_{x\in P(\phi,k_2)}\mu_x$, then $f_{\phi,k_2}=0$.

Corollary 10: If $f_{\phi,k_1}>0$ and $f_{\phi,k_2}>0$, then $\sum_{x\in P(\phi,k_1)}\mu_x=\sum_{x\in P(\phi,k_2)}\mu_x$.

Since Problem (2a) is convex, it is straightforward to show that if there does not exist a path that improves the capacity, then the current routing is optimal. Thus, Algorithm 3 can be used to incrementally add paths until no further improvement is possible, at which point the optimal capacity has been determined.

Algorithm 3 Optimal Routing

0: Select an initial set of paths, $\mathcal{P}(0)$. Set $m=0$.
1: Compute the optimal schedule for the set of Considered paths $\mathcal{P}(m)$.
1.a: Solve Problem 2a
1.b: As a byproduct of solving Problem 2a, compute the Lagrange multiplers μ_x associated with Constraint (2b).
2: Search for a path $P(\phi,k^+)\notin\mathcal{P}(m)$ that satisfies (10).
if such a path exists **then**
  Add this path to the set of considered paths for connection ϕ. Set $m=m+1$, and go to Step 4.
else
  If no such path exists, then stop, the optimal routing has been found.
end if
4: Remove any unecessary paths from consideration. Then go to Step 1.

There are several challenges facing the application of Algorithm 3. A critical one is related to determining μ_x for all links. This issue is addressed in the next section. Another important issue is related to removing unnecessary paths as discussed in Step 4. One approach is to remove paths that carry no data. That is, if the optimal schedule dictates that $f_{\phi,k}=0$, then the path $P\left(\phi,k\right)$ can be removed from consideration without impacting the capacity. However, as discussed in the next sections, in some cases, the removal of paths is more complicated.

In [55], the idea of updating routing based on the sum of the end-to-end Lagrange multipliers was investigated for wired networks. However, the focus there was on single path routing and on the conditions under which single path routing is optimal. Note that in the wired case, μ_x is typically know for all links x.

A. Evaluating the Path Cost

1) Approaches to Evaluating the Path Cost: One significant drawback of Algorithm 3 is that it requires μ_x to be known for each link x. However, the scheduling algorithm will only

compute μ_x if there is a path in $\mathcal{P}$ that uses link x. Hence, if there are some links that are not used by any path, then μ_x is unknown for some x, and, hence (10) cannot be computed for all candidate paths $P(\phi, k^+)$. There are three ways to address this drawback. First, Algorithm 3 can be applied but neglecting links that are not used by any path in $\mathcal{P}$. The result will be suboptimal. Specifically, only links considered by the initial routing will be used when the algorithm terminates. Based on computations with realistic mesh networks (See Section III-D.1), we have found that this approach performs poorly, especially when the network is dense (i.e., when there are many links that have high channel gain and are not included into the routing).

A second approach to accommodate Algorithm 3's need for μ_x for all links is to ensure that at each iteration, the routing is such that each link is used by at least one path. This brute force suffers from high computational complexity since there maybe a large number of links. On the other hand, this approach is guaranteed to yield the optimal routing.

A third approach is to find an upper bound, $\bar{\mu}_x$ such that $\mu_x \leq \bar{\mu}_x$ for the links that are not included in any path and $\bar{\mu}_x = \mu_x$ for links that are included in at least one path. If such a bound is known (one is given in the next section), then from Proposition 8 we have the following.

Corollary 11 (Sufficient Condition to Include a Path): Let $\mu_x \leq \bar{\mu}_x$. Let $P(\phi, k^+) \notin P(\phi)$ be the path of a candidate path potentially to be added to $P(\phi)$. Furthermore, let $P(\phi, k^o) \in P(\phi)$ be such that $f_{\phi,k^o} > 0$. Then, adding the candidate path $P(\phi, k^+)$ to $P(\phi)$ will increase the capacity if

$$\sum_{x \in P(\phi,k^+)} \bar{\mu}_x < \sum_{x \in P(\phi,k^o)} \mu_x. \tag{11}$$

Note that since the path $P(\phi, k^o)$ is already included in $\mathcal{P}$, μ_x is known for all $x \in P(\phi, k^o)$, and hence, the right-hand side of (11) can be computed. Of course, the drawback of using a bound $\bar{\mu}_x$ is that some paths $P(\phi, k^+)$ might satisfy (10), but not (11). Such paths would not be included into $\mathcal{P}$. Therefore, replacing condition (10) with condition (11) in Algorithm 3 results in suboptimal routing. On the other hand, at each iteration of Algorithm 3, more paths are added to $\mathcal{P}$. Thus, if, due to the inaccuracy of $\bar{\mu}_x$, a path is not added, it might be added at later iterations once μ_x is determined for more links.

The performance of Algorithm 3 with condition (11) can be improved by adjusting when paths are removed in Step 4. On the one hand, if more links that are included in the $\mathcal{P}$, then μ_x is known for more links x, and hence the decision of which paths to include can be made more accurately. On the other hand, the computational complexity of the scheduling problem is closely related to the size of $\mathcal{P}$. Thus, Algorithm 4 was developed to determine when paths should be removed from consideration. Roughly speaking, Algorithm 4 removes paths after they have not been used by the routing algorithm in any way for more than M iterations. As demonstrated in Section V, $M = 5$ works well.

In summary, as an alternative to ensuring that each link is used by some route, Algorithm 5 is used.

Algorithm 4 Heuristic for Removing Paths from Consideration

The path $P(\phi, k)$ is not removed from the set of considered paths if any of the following holds.
0: The path $P(\phi, k)$ was added to the set of considered paths within the past M iterations.
1: $f_{\phi,k} \neq 0$ during at least one of the past M iterations.
2: There exists a link $y \in P(\phi, k)$ such that $y \notin P(\varphi, j)$ for any other flow (φ, j) and μ_y has been used in computing the upper bound $\bar{\mu}_z$ for some link z at least once during the past M iterations.

Algorithm 5 (Sub)Optimal Routing

Use Algorithm 3 with the following changes.
1: In Step 2, Condition (10) is replaced with Condition (11) where $\bar{\mu}_x$ in Condition (11) is computed with (12).
2: In Step 4, Algorithm 4 is used.

2) An Upper Bound on μ_x: Let $\mathcal{L}$ be the set of links used in any path, i.e., $\mathcal{L} = \{x : \text{there exists a } P \in \mathcal{P} \text{ such that } x \in P\}$. Thus, for each $x \in \mathcal{L}$, μ_x is determined by the scheduling. Denote by $\mathcal{CG}[\mathcal{L}]$ to be the conflict graph induced by the links $\mathcal{L}$. Furthermore, the subgraph of the conflict graph induced by removing link x is denoted with $\mathcal{CG}[\mathcal{L}\backslash x]$. Assign the weight $\mu_x R_x$ to the vertex in the conflict graph that corresponds to link x, where, as in Section II, R_x is the data rate over link x when there are no links $y \in \chi(x)$ are transmitting. Let $MWIS(CG[L])$ denote the links in the maximum weighted independent set of the weighted conflict graph $CG[L]$. Similarly, let $MWIS(CG[L\backslash(\{z\} \cup \chi(z))])$ be the set of links in a maximum weighted independent set of the conflict graph $CG[L\backslash(\{z\} \cup \chi(z))]$, which is the conflict graph derived from the set of link $\mathcal{L}$, excluding link z and all links in conflict with z. Employing this notation, we have the following.

Proposition 12: Let $z \notin \mathcal{L}$, then $\bar{\mu}_z \geq \mu_z$ where

$$\bar{\mu}_z = \left(\lambda - \sum_{x \in MWIS(\mathcal{CG}[\mathcal{L}\backslash(\{z\} \cup \chi(z))])} \mu_x R_x \right) / R_z. \tag{12}$$

The economic interpretation of Lagrange multipliers can be used to intuitively explain (12). Specifically, μ_x is the price to transmit a bit across link x. Let $v \in \{0,1\}^L$ be an assignment without conflicts (i.e., $v_x = 1$ implies $v_y = 0$ for all $y \in \chi(x)$). Then the revenue per second generated by this assignment is $\sum_{x=1}^{L} R_x \mu_x v_x$. Recall from Proposition 3 that the best assignments are those that achieve $\lambda = \sum_x R_x \mu_x v_x$. Thus, the network is "willing" to multiplex between any assignments as long as the revenue per second generated by the assignments is λ. Thus, the network is willing to allow transmission over z if the price per bit to transmit across link z is (12).

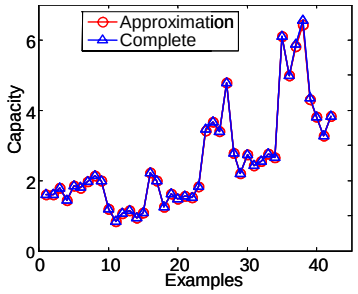

Fig. 5. Capacity found from optimal and suboptimal routing for urban topologies with 18 and 36 nodes and one and three gateways. Since the two computed capacities are the same, only one set of points is visible.

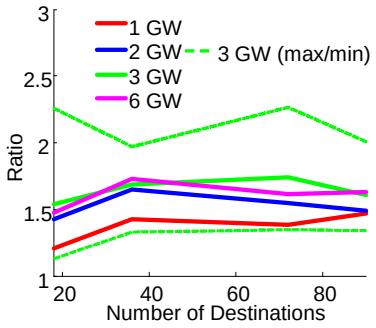

Fig. 6. The median of the ratio of the capacity with (sub)optimal routing and the capacity with the least hop routing. Also shown is the maximum and minimum value of the ratio when there are three gateways. The maximum and minimum for other numbers of gateways are similar.

While (12) yields an useful bound on μ, it requires the computation of a maximum weighted independent set. While we have found that in realistic networks, such set can be computed relatively quickly, computing a maximum weighted independent set for a large number of links is computational complex.

V. Numerical Experiments with Joint Scheduling and Routing

In this section, the behavior of the different routing algorithms is compared. In all cases, optimal scheduling is used. Hence, only the routing is changed. In the following computational experiments, the capacity metric is $G\left(\vec{f}\right) = \min_{\phi_{\in\Phi}} w_\phi \sum_{k=1}^{|P(\phi)|} f_{\phi,k}$. However, the results for $G\left(\vec{f}\right) = \sum_{\phi\in\Phi} w_\phi \log\left(\sum_{k=1}^{|P(\phi)|} f_{\phi,k}\right)$ have been examined and yield qualitatively similar results. The topologies used here are the 6×6 city block topologies described in Section III-D.1. The baseline routing is also described in Section III-D.1.

A. Comparison of Algorithms

As discussed in Section IV-A, Algorithm 3 yields optimal routing if the Lagrange multiplier for each possible link in the network is always included into some path. Typically, there are many possible links in the network, hence, this approach is computational complex. Algorithm 5 is less computational complex, but is suboptimal. On the other hand, since Algorithm 4 is conservative in removing a path from the set of considered paths, Algorithm 5 tends to include a fairly large number of links, and hence has the potential to accurately determine which paths to include.

Figure 5 compares the capacity found by these algorithms for topologies with 18 and 36 mesh routers and with 1 and 3 gateways. For each number of routers and gateways, ten samples were considered (hence, 40 topologies in total were considered). As can be observed, Algorithm 5 yields the same capacity as the optimal algorithm. We have found that Algorithm 4 is critical to the performance of Algorithm 5. On the other hand, Algorithm 5 tends to include a large number of links, increasing the computational complexity. Nonetheless, Algorithm 5 is considerably more efficient than including all possible links, and hence optimal routing with all possible links is not considered in the remainder of this paper. Nonetheless, since we have not verified that Algorithm 5 is indeed optimal for all topologies, the capacity found by Algorithm 5 is referred to as *(sub)optimal*.

B. The Impact of Optimal Routing

The left-hand side of Figure 6 shows the improvement in the capacity of (sub)optimal routing as compared to least hop routing. As can be observed, optimal routing can significantly improve the capacity. In Section III-D.2, the same topologies used here were considered and it was found that, as compared to 802.11's MAC, optimal scheduling yields an improvement in the capacity by between a factor of four and ten (depending on the topology). Hence, we conclude that joint optimal routing and scheduling provides between a factor of 5 and 15 improvement over 802.11 CSMA/CD with least hop routing!

C. Path Length

It is often claimed that high performance can be achieved by taking many short hops where, since each hop is short, it has a high SNR, and hence can support high bandwidth. While high data rates might be possible over a single short hop, a connection composed of many short hops will require many transmissions, and hence, will experience self-interference and interference with other connections. In this way, perhaps, shorter paths and longer hops might be preferable [56].

Figure 7 shows the average number of hops for (sub)optimal routing and least hop routing while Figure 8 shows the ratio of the path lengths. We observe that question of whether paths should be short or long depends on the details of the topology. For example, in the case of one gateway in a 6×6 city block region, (sub)optimal routing uses significantly longer paths than least hop paths. However, as shown in Figure 8, the ratio of the length of the (sub)optimal path and the length of the least hop path decreases as the density of gateways increases, and this ratio increases as the density of destinations increases.

Note that since multipath routing is used and since the scheduling may assign small data flow to some paths, the path lengths used in Figure 7 and 8 are weighted by the amount of data flow that crosses the path. Specifically, the path length for connection ϕ is $\frac{1}{\sum_{k=1}^{|P(\phi)|} f_{\phi.k}} \sum_{k=1}^{|P(\phi)|} f_{\phi,k} \left(|P\left(\phi,k\right)|-1\right)$, where $|P\left(\phi,k\right)|-1$ is the length of path $P\left(\phi,k\right)$.

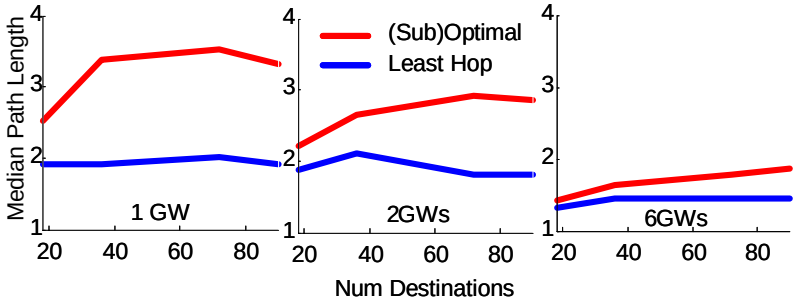

Fig. 7. Median path length for different topologies and different routing algorithms.

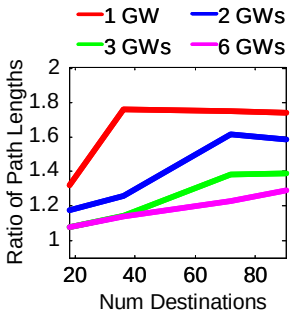

Fig. 8. Ratio of the length of the optimal path and the length of the least hop path.

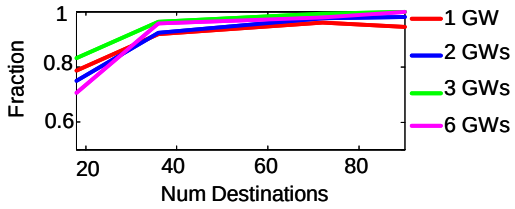

Fig. 9. Fraction of connections that use multiple paths.

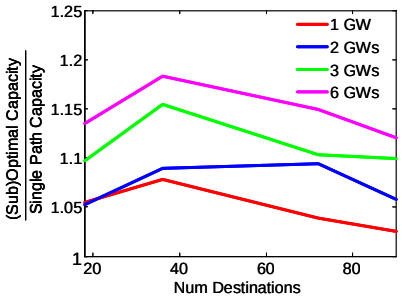

Fig. 10. Ratio of the (sub)optimal capacity and the capacity provided by a single path routing that is a quantized version of (sub)optimal routing.

D. Multipath versus Single Path Routing

There has been extensive work on multipath routing (e.g., [57] and [58]). One motivation for using multipath routing is that multiple paths can increase robustness to disconnection and improve load balancing. It is expected that load balancing can increase capacity. While there is some evidence of this behavior in single connection ad hoc routing [59], the behavior in mesh networks at full capacity is unclear. In theory, optimal multipath routing is more general than optimal single path routing, and hence multipath routing cannot provide any lower capacity than single path. However, it is unknown the degree to which multipath routing is required for maximum capacity. Figure 9 shows the fraction of connections that use multiple paths. Note that while Algorithm 5 might generate multiple paths, the scheduling algorithm might not allocate data flow to some paths. Thus, we say that a connection ϕ uses multiple paths if there exists k and j with $k \neq j$ and $f_{\phi,k} > 0$ and $f_{\phi,j} > 0$ [2]. As can be observed in Figure 9, a large majority of the connections require multiple paths. Note that a significant number of the paths were only a single hop. Thus, Algorithm 5 even chooses to augment paths that are one hop paths with multi-hop paths.

While Figure 9 indicates that optimality makes significant use of multiple paths, it does not indicate the impact of multipath routing on capacity. Since optimal single path routing is a non-convex optimization [55], a precise comparison between optimal multipath routing and optimal single path routing is computationally complex. Nonetheless, an idea of the performance of optimal single path routing can be gleaned by considering a single path routing that is a "quantized" version of multipath routing as follows. As mentioned above, even when there are multiple paths between a single source-destination pair, the scheduling algorithm need not allocate data flow to all available paths. For the paths that do have non-zero data flow, each path does not necessarily have the same data flow. Thus, for each connection ϕ, we select a single path $P(\phi, k)$ such that $f_{\phi,k} \geq f_{\phi,j}$ for all j. If there are multiple paths with the same maximal data rate, then one is selected arbitrarily. Once the paths have been selected, the scheduling algorithm is used to compute the optimal capacity under the single path routing. Figure 10 shows the ratio of the capacity given by this single path routing and the capacity provided by the (sub)optimal multipath routing. As can be observed, the (sub)optimal routing provides no more than 20% higher capacity and generally less than 15% higher capacity than the single path routing.

Note that when multipath routing is used, the address of the destination does not specify the route (or next hop). Hence, a labeling scheme or full source routing must be used. Moreover, if a single TCP flow is split over multiple paths, packet reordering is likely to occur. If packets are reordered by more than three, then TCP will assume a packet has been dropped and will decrease the congestion window and sending rate. While solutions to this problem exist (e.g., [60]), they are not widely deployed. Thus, when including the reduction in capacity due to the overhead to support multipath routing, the difficulty in that multipath routing has on TCP, and the minimal increase in capacity provided multipath routing, there appears to be little motivation to deploy multipath routing.

[2]Due to numerical quantization, a path was declared to have no data flow if $f_{\phi,k} \leq 10^{-8}$.

VI. Conclusions

This paper presented tractable computational techniques were to compute optimal schedules and routing for multihop wireless networks. While optimal schedules can be quickly computed, optimal routing is computationally complex for large networks. Thus, a suboptimal scheme was developed. In the cases where optimal routing was computed, the suboptimal and optimal routing were the same. The impact of optimal scheduling and routing is significant. For example, as compared to 802.11's CSMA/CD with least hop routing, optimal scheduling and routing improves performance by between a factor of 5 and 15, with the improvement increasing as the density of gateways increases. The impact of power control and multiple bit-rates was not studied.

VII. Appendix

A. Proof of Propositions 3 and 4

Proof: The Lagrange function is

$$L(f,\alpha,\mu,\lambda) = -\sum_{\phi\in\Phi} w_\phi \log(f_\phi) + \lambda\left(\sum_{v\in V}\alpha_v - 1\right) + \sum_{l=1}^{L}\mu_l\left(\sum_{\{\phi:l\in\mathcal{P}(\phi)\}} f_\phi - \sum_{v\in V}\alpha_r R(v,l)\right). \quad (13)$$

After some manipulation, the dual function is found to be

$$q(\mu,\lambda) = \inf_{f,\alpha\geq 0} -\sum_{\phi\in\Phi}\log(f_\phi)\,w_\phi - \lambda + \sum_{l=1}^{L}\mu_l\sum_{\{\phi:l\in\mathcal{P}(\phi)\}} f_\phi - \sum_{v\in V}\alpha_v\left(\sum_{l=1}^{L} R(v,l)\,\mu_l - \lambda\right). \quad (14)$$

We immediately note that if $\sum_{l=1}^{L} R(v,l)\,\mu_l - \lambda > 0$ for some l, then $q(\mu,\lambda) = -\infty$. Hence, we restrict the domain of q, to be such that $\sum_{l=1}^{L} R(v,l)\,\mu_l - \lambda \leq 0$. On the other hand, when solving the dual problem, an objective is to maximize q with respect to λ. It is not hard to see that this is equivalent to minimizing λ over the domain $\sum_{l=1}^{L} R(v,l)\,\mu_l - \lambda \leq 0$. Thus,

$$\lambda^* = \max_{v\in V}\sum_{l=1}^{L} R(v,l)\,\mu_l, \quad (15)$$

proving Proposition 3. Furthermore, for this λ, the $\sum_{x=1}^{L} R(v,x)\,\mu_x - \lambda < 0$ for $v \notin V$, thus, the infimum in (14) must have $\alpha_v = 0$ for $v \notin V$, proving Proposition 4. ■

Therefore, we can rewrite the dual function as[3],

$$q(\mu) = \inf_{f\geq 0} -\sum_{\phi\in\Phi}\log(f_\phi)\,w_\phi + \sum_{l=1}^{L}\mu_l\sum_{\{\phi:l\in\mathcal{P}(\phi)\}} f_\phi - \max_{v\in V}\sum_{l=1}^{L} R(v,l)\,\mu_l, \quad (16)$$

[3]There are other, more straightforward ways to arrive at (16). However, these methods do not provide the important expression (15).

where α_v has been eliminated since λ^* results in the infimum being achieved for $\alpha_v = 0$.

Proof: [Theorem 2] The optimal average data rates over each link is a convex sum of the links rates from different assignments, that is, the optimal bit-rate over link x is $\sum_{v\in\mathcal{V}}\alpha_v^* R(v,x)$, where α^* defines the optimal schedule. In other words, the set of feasible link bit-rates is a convex set where the extreme points some of the rows of R. Obviously, the vector of optimal link rates is the vector $\sum_{v\in\mathcal{V}}\alpha_v^* R(v,:) \in \mathbb{R}^L$, the space of vectors with L elements. Due to Caratheodory's Theorem (e.g., Theorem B.6 in [24]), a point within a convex hull in $\mathbb{R}^L$ is specified by at most $L+1$ extreme points. That is, there exists a set, $\mathcal{V}'$ with $L+1$ elements such that

$$\sum_{v\in\mathcal{V}}\alpha_v^* R(v,:) = \sum_{v\in\mathcal{V}'}\alpha_v' R(v,:),$$

where α' might be different set of weights from α^*. Hence, the optimal link bit-rates found by optimizing over V^{**} , the set of all possible assignments, can be achieved by only using the set of assignments $\mathcal{V}'$. Thus, the resulting utility is unchanged when $\mathcal{V}'$ is used as opposed to $\mathcal{V}^{**}$.

Now it is shown that $\mathcal{V}'$ can be selected so that $\mathcal{V}'$ has less than $L+1$ elements. Suppose otherwise, that is, V' has exactly $L+1$ elements $\mathcal{V}'$ is the smallest set such that the optimal schedule is in $Co(\{R(v,:) : v\in\mathcal{V}'\})$, the convex hull of $\{R(v,:) : v\in\mathcal{V}'\}$. Since the faces of $Co(\{R(v,:) : v\in\mathcal{V}'\})$ are defined by no more than L extreme points, the assumption that the optimal bit-rates cannot be specified by L points implies that the optimal bit-rates must be in the interior of $Co(\{R(v,:) : v\in\mathcal{V}'\})$. That is, there is an open set that contains the optimal point and this open set is in the interior of $Co(\{R(v,:) : v\in\mathcal{V}'\})$. For example, letting $\mathbf{r}^{**}$ be the vector of optimal bit-rates, the vector $\mathbf{r}^{**}+\varepsilon\mathbf{r}^{**}$ is also in the interior of $Co(\{R(v,:) : v\in\mathcal{V}'\})$, where $\varepsilon > 0$ is small enough. Since $\mathbf{r}^{**}$ is the optimal vector of bit-rates over the interior of $Co(\{R(v,:) : v\in\mathcal{V}'\})$, the utility of $\mathbf{r}^{**}$ must be higher than the utility of $\mathbf{r}^{**}+\varepsilon\mathbf{r}^{**}$. However, this is a contradiction since the link bit-rates $\mathbf{r}^{**}+\varepsilon\mathbf{r}^{**}$ result in uniformly large flow rates than $\mathbf{r}^{**}$, which will increase the capacity. Hence, $\mathcal{V}'$ can be selected to have fewer than $L+1$ elements. ■

Proof: [Theorem 5] We slightly modify Problem (2a) to

$$\begin{aligned}&\min G(\mathbf{f})\\ &\text{subject to:} \sum_{\{\phi:x\in P(\phi)\}} f_\phi - \sum_{v\in\mathcal{V}}\alpha_v R(v,x) \leq \rho_x\\ &\qquad\qquad \sum_{v\in\mathcal{V}}\alpha_v - 1 = A,\end{aligned}$$

so (2a) is the case where $\boldsymbol{\rho} = 0$ and $A = 0$. We will denote the value of the optimal solution of the above problem as $G^*(\boldsymbol{\rho}, A)$. From sensitivity analysis (e.g., [24]), we have

$$\mu_x^* = \frac{\partial U^*(\rho,A)}{\partial\rho_x} \quad (17)$$

$$\lambda^* = \frac{\partial U^*(\rho,A)}{\partial A}. \quad (18)$$

Equation (17), implies that if the amount of bit-rate that is applied to link x is increased by an small amount ε, then the total utility will increase by $\mu_x \varepsilon$. It is critical to note that in this analysis, the bit-rate applied to link x does not come at the expense of bit-rates of other links.

Now consider the multiplier, λ^*. The constraint $\sum_{v\in\mathcal{V}} \alpha_v = 1 + A$ can be interpreted as the allowing the total bandwidth of size $1 + A$ to be shared among all assignments. Thus, if the bandwidth is increased from size 1 to size $1 + \varepsilon$, then the capacity will increase by $\lambda\varepsilon$. Similarly, if the bandwidth is decreased by ε, then the utility will decrease by $\lambda\varepsilon$.

While the analysis above assumed that the extra bandwidth is allocated to link x without impacting the bit-rate of the other links, we now consider the more relevant problem where this extra assignment comes at the expense of other links. Specifically, if we allocate assignment v^+ with ε of the bandwidth, then the total bandwidth allocated to the other assignments must be decreased by ε. In particular, let $\mathcal{V}' = \{v_1, ...v_N\}$ and when optimizing over the set of assignments $\mathcal{V}'$, let the associated optimal bandwidth allocated to v_i be of size α_i^*, where, of course, $\sum_{i=1}^N \alpha_i^* = 1$. Now in order to allocate bandwidth ε to assignment v^+, we adjust the allocation to $\alpha_i^+ = (1-\varepsilon)\,\alpha_i^*$, and hence the assignments $\{v^+, v_1, v_2, ..., v_N\}$ are allocated bandwidths of width $\{\varepsilon, (1-\varepsilon)\,\alpha_1, (1-\varepsilon)\,\alpha_2, ..., (1-\varepsilon)\,\alpha_N\}$, respectively. Based on the discussion above, the change in capacity is

$$\varepsilon\left(\sum_{x=1}^{L} \mu_x^* R\left(v^+, x\right) - \lambda\right), \tag{19}$$

which is positive if (6) holds. ■

B. Proof of Proposition 8 and 12

Proof: [Proof of Proposition 8.] We consider the optimization with nonlinear objective function. The specialization to the linear objective function follows the same approach.

Consider the optimization problem

$$\max \sum_{\substack{\phi\in\Phi\\ \phi\neq\phi^+}} U_\phi\left(\sum_k f_{\phi,k}\right) + U_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right) \tag{20}$$

$$\text{subject to: } \sum_{\{(\phi,k)|x\in\mathcal{P}(\phi,k)\}} f_{\phi,k} \le \sum_{v\in V} \alpha_v R\left(v,x\right) \ \ \forall x$$

$$f_{\phi^+,k^+} \le b_{\phi^+,k^+}$$

$$0 \le f_{\phi,k}$$

$$\sum_{v\in V} \alpha_v \le 1$$

$$0 \le \alpha_v \text{ for each } v \in V.$$

Setting $b_{\phi^+,k^+} = 0$ results in flow $\left(\phi^+, k^+\right)$ not being used, and hence, in this case, the solution is the same as the original optimization (e.g., the multipliers $\boldsymbol{\mu}$ and λ are unchanged). The Lagrangian for is (20) is

$$\begin{aligned}
&L\left(\overrightarrow{f}, \overrightarrow{\alpha}, \gamma_{\phi^+,k^+}, \overrightarrow{\mu}, \lambda, \overrightarrow{\sigma}\right)\\
=\ & -U_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right)\\
&+\sum \mu_l\left(\sum_{\{(\phi,k)|x\in P(\phi,k)\}} f_{\phi,k} - \sum \alpha_v R\left(v,x\right)\right)\\
&+\gamma_{\phi^+,k^+}\left(f_{\phi^+,k^+} - b_{\phi^+,k^+}\right) - \sigma_{\phi^+,k^+} f_{\phi^+,k^+}\\
&+\lambda\left(\sum_{v\in V}\alpha_v - 1\right) - \sum_{\substack{\phi\in\Phi\\ \phi\neq\phi^+}} U_\phi\left(\sum_k f_{\phi,k}\right)\\
&-\sum_{(\phi,k)\neq\left(\phi^+,k^o\right)} \sigma_{\phi,k} f_{\phi,k}
\end{aligned}$$

where all Lagrange multipliers, $\gamma_{\phi^+,k^+}, \boldsymbol{\mu}, \lambda, \boldsymbol{\sigma}$ are non-negative. The dual function is

$$\begin{aligned}
&q\left(\gamma_{\phi^+,k^+}, \boldsymbol{\mu}, \lambda, \boldsymbol{\sigma}\right) \qquad (21)\\
=\ & \inf_{f_{\phi^+,k^o}} -U_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right)\\
&+f_{\phi^+,k^+}\left(\sum_{x\in P\left(\phi^+,k^+\right)} \mu_x + \gamma_{\phi^+,k^+} - \sigma_{\phi^+,k^+}\right) + D
\end{aligned}$$

where D represents other terms that do not depend on f_{ϕ^+,k^+}. From (21), f_{ϕ^+,k^+} is such that

$$\begin{aligned}
0 \ =\ & \frac{d}{df_{\phi^+,k^+}}\left(-U_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right)\right.\\
&\left.+f_{\phi^+,k^+}\left(\sum_{x\in P\left(\phi^+,k^+\right)} \mu_x + \gamma_{\phi^+,k^+} - \sigma_{\phi^+,k^+}\right)\right)
\end{aligned}$$

or

$$\begin{aligned}
&U'_{\phi^+}\left(\sum_{k\neq k^o} f_{\phi^+,k} + f_{\phi^+,k^+}\right) \qquad (22)\\
=\ & \sum_{l\in P\left(\phi^+,k^+\right)} \mu_l + \gamma_{\phi^+,k^+} - \sigma_{\phi^+,k^+},
\end{aligned}$$

where $U'_{\phi^+}(f) = \frac{d}{df} U_{\phi^+}(f)$.

Now we consider the case where $b_{\phi^+,k^+} = 0$. In this case, at the optimal point, $f_{\phi^+,k^+} = 0$, and hence, $U'_{\phi^+}\left(\sum_{k\neq k^o} f_{\phi^+,k} + f_{\phi^+,k^o}\right) = U'_{\phi^+}\left(\sum_{k\neq k^o} f_{\phi^+,k}\right)$. Thus, from (22), if

$$U'_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k}\right) < \sum_{x\in P\left(\phi^+,k^+\right)} \mu_x \tag{23}$$

then $\gamma_{\phi^+,k^o} > 0$. Thus, by sensitivity analysis of Lagrange multiplier theory, if and only if (23) holds, then the capacity will increase if b_{ϕ^+,k^+} is increased. That is, the capacity will increase if data is allowed to be sent over path $P\left(\phi^+,k^+\right)$.

Next we show that

$$U'_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k}\right) = \sum_{l\in P\left(\phi^+,k^o\right)} \mu_l. \tag{24}$$

To see this, the same analysis as above can be done but with k^+ replaced with k^o. Furthermore, since data is allowed to flow over flow $\left(\phi^+,k^o\right)$ we have $b_{\phi^+,k^o} = \infty$. Hence, the constraint $f_{\phi^+,k^o} \leq b_{\phi^+,k^o}$ is not active, and hence $\gamma_{\phi^+,k^o} = 0$. Moreover, since $f_{\phi^+,k^o} > 0$, the constraint $f_{\phi^+,k^o} \geq 0$ is not active, and hence $\sigma_{\phi^+,k^o} = 0$. Hence, (22) implies (24). Finally, substituting the right-hand side of (24) into the left-hand side of (23) results in (10). ■

Proof: [Proposition 12] In Problem (2a), not all flows are considered. We denote the set of considered paths $\mathcal{P}$. We define a shadow problem where all possible paths are included, but the data rate along any path not in $\mathcal{P}$ are restricted so that the data rate crossing these flows is no greater than ε. Hence, as $\varepsilon \to 0$, the solution to the shadow problem and the original problem are the same. However, in the case of the shadow problem, the Lagrange multipliers for each link is known. Specifically, let $\mathcal{L}$ be the set of links used by any path in $\mathcal{P}$ and let $\mathcal{A}$ be the set of all links. Then, by computing the optimal schedule when the paths $\mathcal{P}$ are considered, we can determine μ_x for $x \in \mathcal{L}$. And, from the optimal schedule where all paths are considered, μ_x is known for all links $x \in \mathcal{A}$. It is straightforward to verify that the for $x \in \mathcal{L}$, the μ_x is the same in both cases. Moreover, λ is also the same.

Let $z \in \mathcal{A}\backslash\mathcal{L}$. An assignment that includes z is $\{z\} \cup MWIS\left(\mathcal{CG}\left[\mathcal{A}\backslash\left(\{z\}\cup\chi\left(z\right)\right)\right]\right)$. By Proposition 3,

$$\mu_z R_z + \sum_{x\in MWIS(\mathcal{CG}[\mathcal{A}\backslash(\{z\}\cup\chi(z))])} \mu_x R_x \leq \lambda.$$

Since

$$\sum_{x\in MWIS(\mathcal{CG}[\mathcal{L}\backslash(\{z\}\cup\chi(z))])} \mu_x R_x \leq \sum_{x\in MWIS(\mathcal{CG}[\mathcal{A}\backslash(\{z\}\cup\chi(z))])} \mu_x R_x$$

we have

$$\mu_z R_z \leq \lambda - \sum_{x\in MWIS(\mathcal{CG}[\mathcal{L}\backslash(\{z\}\cup\chi(z))])} \mu_x R_x$$

which is the desired result. ■

REFERENCES

[1] K. Dell, "Welcome to wi-fi-ville," *Time*, Jan. 05, 2007.

[2] D. J. Baker, J. Wieselthier, and A. Ephremides, "A distributed algorithm for scheduling the activation of links in a self-organizing mobile radio network," in *IEEE ICC*, 1982, pp. 2F.6.1–2F.6.5.

[3] E. Arikan, "Some complexity results about packet radio networks," *IEEE Trans. on Info. Theory*, vol. 30, no. 4, pp. 681–685, Jul 1984.

[4] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on crosslayer rate control in multihop wireless networks," in *Proc. Of INFOCOM*, 2005.

[5] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint congestion control and distributed schedulingin multihop wireless networks with a node-exclusive interference model," in *Infocom*, 2006.

[6] A. Eryilmaz and R. Srikant, "Joint congestion control, routing, and MAC for stability and fairness in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1514 – 1524, August 2006.

[7] A. Kashyap, S. Sengupta, R. Bhatia, and M. Kodialam, "Two-phase routing, scheduling and power control for wireless mesh networks with variable traffic," in *Sigmetrics07*, 2007.

[8] S. Sanghavi, L. Bui, and R. Srikant, "Distributed link scheduling with constant overhead," in *SIGMETRICS*, 2007, pp. 313 – 324.

[9] S. Sarkar and L. Tassiulas, "End-to-end bandwidth guarantees through fair local spectrum share in wireless ad-hoc networks," *IEEE Trans. on Automatic Control*, vol. 50, no. 9, pp. 1246–1259, Sep 2005.

[10] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint asynchronous congestion control and distributed scheduling," in *Infocom*, 2006.

[11] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *Information Theory, IEEE Transactions on*, vol. 34, no. 5, pp. 910–917, 1988.

[12] A. Brzenzinski, G. Zussman, and E. Modiano, "Enabling distributed throughput maximization in wireless mesh networks - a partitioning approach," in *Mobicom*, 2006.

[13] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput and fairness guarantees through maximal scheduling in wireless networks," in *Proccedings of the Allerton Conference on Communication, Control, and Computing*, 2005.

[14] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *Infocom*, 2006.

[15] A. Behzad and I. Rubin, "On the performance of graph-based scheduling algorithms," in *IEEE Globecom*, 2003, pp. 3432–3436.

[16] J. Grnkvist and A. Hansson, "Comparison between graph based and interference based STDMA scheduling," in *Proc. Of ACM Mobihoc*, 2001.

[17] S. Bohacek, V. Sridhara, and J. Kim, "UDel Models," available at: http://udelmodels.eecis.udel.edu/.

[18] P. Wang and S. Bohacek, "Capacity optimization of mesh networks," *Submitted. Available at http://udelmodels.eecis.udel.edu*, 2007.

[19] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, Nov 1998.

[20] F. P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, January 1997.

[21] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 525–536, August 2003.

[22] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking*, vol. 8, pp. 556–567, 2000.

[23] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proceedings of ACM MobiCom*, San Diego, CA, September 2003, pp. 66–80.

[24] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 2003.

[25] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 2000.

[26] T. ElBatt and A. Ephremides, "Joint scheduling and power control for wireless ad-hoc networks," in *Proceedings of IEEE INFOCOM*, New York, NY, June 2002, pp. 976–985.

[27] M. Grotschel, L. Lovasz, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*. Berlin: Springer-Verlag, 1993.

[28] T. Matsui, "Approximation algorithms for maximum independent set problems and fractional coloring problems on unit disk graphs," in *JCDCG*, 1998, pp. 194–200.

[29] G. Minty, "On maximal independent sets of vertices in claw-free graphs," *J. Combinatorial Theory*, vol. B, no. 28, pp. 284–304, 1980.

[30] V. Alekseev, "A polynomial algorithm for finding the largest independent sets in fork-free graphs," *Discrete Applied Mathematics*, vol. 135, pp. 3–16, 2004.

[31] G. H. Chen, M. T. Kuo, and J. P. Sheu, "An optimal time algorithm for finding a maximum weight independent set in a tree," *BIT*, vol. 23, pp. 353–356, 1988.

[32] R. Karp and M. Sipser, "Maximum matchings in sparse random graphs," in *FOCS*, 1981.

[33] G. Valiente, *A New Simple Algorithm for the Maximum-Weight Independent Set Problem on Circle Graphs*. Springer, 2003, vol. 2906, pp. 129–137.
[34] M. M. Halldórsson, *Approximation Algorithms for Combinatiorial Optimization*. Springer Berlin / Heidelberg, 2004, ch. Approximations of Independent Sets in Graphs, pp. 24–45.
[35] M. Fürer and S. P. Kasiviswanathan, "Algorithms for counting 2-SAT solutions and colorings with applications," *Electronic Colloquium on Computational Complexity (ECCC)*, no. 033, 2005.
[36] F. V. Fomin, S. Gaspers, and S. Saurabh, "Branching and treewidth based exact algorithms," in *ISAAC*, 2006, pp. 16–25.
[37] L. Babel, "A fast algorithm for the maximum weight clique problem," *Computing*, vol. 52, pp. 31–38, 1994.
[38] E. Balas and J. Xue, "Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring," *Algorithmica*, vol. 15, no. 5, pp. 397–412, 1996.
[39] ——, "Minimum weighted coloring of triangulated graphs, with application to maximum weight vertex packing and clique finding in arbitrary graphs," *SIAM J. Comput.*, vol. 20, no. 2, pp. 209–221, 1991.
[40] J. S. Warren and I. V. Hicks, "Combinatorial branch-and-bound for the maximum weight independent set problem," 2007.
[41] P. R. J. Östergård, "A fast algorithm for the maximum clique problem," *Discrete Applied Mathematics*, vol. 120, no. 1-3, pp. 197 – 207, August 2002.
[42] H. N. Gabow, "Data structures for weighted matchings and nearest ancestors with linking," in *ACM-SIAM Symposium on Discrete Algorithms*, 1990, pp. 434–443.
[43] Y.-S. Cheng, M. Neely, and K. M. Chugg, "Iterative message passing algorithm for bipartite maximum weighted matching," in *2006 IEEE International Symposium on Information Theory*, 2006, pp. 1934–1938.
[44] R. Preis, "Linear time 1/2-approximation algorithm for maximum weighted matching in general graphs," in *Proceedings of The16th Annual Symposium on Theoretical Aspects of Computer Science (STACS 99)*, 1999.
[45] M. Hanckowiak, M. Karonski, and A. Panconesi, "A faster distributed algorithm for computing maximal matching deterministically," in *Proceedings of PODC 99, the Eighteen Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, 1999, pp. 219–228.
[46] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 302–315, April 2006.
[47] A. Czygrinow, M. Hanckowiak, and P. E. Szymanska, "Distributed algorithm for approximating the maximum matching," *Discrete Applied Mathematics*, vol. 143, pp. 62–71, 2004.
[48] V. Sridhara and S. Bohacek, "Realistic propagation simulation of urban mesh networks," *The International Journal of Computer and Telecommunications Networking Computer Networks and ISDN Systems (COMNET)*, 2007.
[49] T. C. of Corpus Christi, "Corpus christi WiFI," 2007.
[50] A. Kako, T. One, T. Hirata, and M. M. Halldorsson, "Approximation algorithms for the weighted independent set problem," available at: http://www.hi.is/ mmh/papers/WIS_WG.pdf.
[51] S. Sakai, M. Togasaki, and K. Yamazaki, "A note on greedy algorithms for the maximum weighted independent set problem," *Discrete Applied Mathematics*, vol. 126, pp. 313–322, 2003.
[52] ILOG, "CPLEX," http://www.ilog.com/products/cplex/.
[53] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. Chapman and Hall/CRC, 1994.
[54] J. Zhu, X. Guo, L. Yang, and W. S. Conner, "Leveraging spatial reuse in 802.11 mesh networks with enhanced physical carrier sensing," in *Proc. Of IEEE ICC*, 2004.
[55] J. Wang, L. Li, S. H. Low, and J. C. Doyle, "Cross-layer optimization in TCP/IP networks," *IEEE/ACM Transactions on Networking*, vol. 13, pp. 582–568, 2005.
[56] M. Haenggi and D. Puccinelli, "Routing in ad hoc networks: A case for long hops," *IEEE Communications Magazine*, October 2005.
[57] S.-J. Lee and M. Gerla, "AODV-BR: backup routing in ad hoc networks," in *IEEE WCNC*, 2000, pp. 1311–1316.
[58] A. Nasipuri and S. R. Das, "On-demand multipath routing for mobile ad hoc networks," in *ICCCN*, 1999, pp. 64–70.
[59] Y. Liu, X. Hu, M. Lee, and T. Saadawi, "A region-based routing protocol for wireless mobile ad hoc networks," *IEEE Network Magazine*, vol. 18, 2004.
[60] S. Bohacek, J. P. Hespanha, J. Lee, C. Lim, and K. Obraczka, "TCP-PR: TCP for persistent packet reordering," *IEEE/ACM Transactions on Networking*, 2006.