Submitted to IEEE Network Magazine

Capacity Optimization of Mesh Networks

Peng Wang and Stephan Bohacek

Department of Electrical and Computer Engineering

University of Delaware

Newark, DE 19716

Abstract

Interference and collisions greatly limited 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 such schedules are so computationally complex that they can only be used on relatively small networks. This paper presents results from several recent works on techniques to efficiently compute capacity optimizing schedules. The techniques presented allow optimal schedules to be computed for networks with hundreds and perhaps even a few thousand links, satisfying the requirements of current and planned mesh networks. The techniques have been verified on networks generated with a realistic propagation simulator, hence, the findings are directly applicable to the networks planned in urban areas such as Philadelphia and San Francisco. In this setting, it is shown that optimal scheduling increases the capacity by between a factor of four and ten over 802.11 CSMA/CA, depending on the density of the mesh routers and gateways.

I. Introduction

One of the key advantages of mesh networks over cellular-based networks is that mesh nodes are relatively inexpensive. Hence, it is economically feasible to spread mesh nodes at a high enough density to provide high data rates to a dense user population. Specifically, the high node density results in a short distance between receivers and transmitters, and hence high SNR channels and high bit-rates are possible. However, the high node density

Stephan Bohacek is the corresponding author. Address: Department of Electrical and Computer Engineering. University of Delaware. Newark, DE 19716. Phone: 302-831-4274. Fax: 302-831-4316. Email: bohacek@udel.edu

also leads to interference and collisions, which, unless properly managed, will reduce the benefits provided by high node density.

There has been some effort focused on reducing interference by adjusting transmission times in an ad hoc fashion [1], [2], [3]. However, a more effective way to control interference and collisions is to use spatial TDMA (STDMA) where nodes transmit according to a schedule such that transmissions are not significantly affected by interference. Efforts in this direction include [4], [5], [6], [7], [8], [9], [10], [11], [12]. However, until recently, computing optimal schedules for networks with more than a few tens of links was intractable (e.g., in [13], the maximum network size was 15 links). However, recently, significant progress has been made. It is now possible to compute optimal schedules for moderate sized networks. In fact, in this paper an example of a 500 link network is given, which is considerably larger than the largest currently deployed mesh network [14].

This paper discusses techniques to compute the capacity and optimizing schedule for mesh networks. This paper also presents insights and results gained from extensive experimentation with capacity maximization. This experimentation utilized a realistic urban propagation simulator [15]. Therefore, these results are directly applicable to the mesh networks currently being deployed.

The paper proceeds as follows. In the next section, the propagation simulator is discussed. Section III provides a detailed problem definition. Section IV discusses techniques to solve the capacity optimization problems. This section has two parts. In Section IV-A, it is shown how the capacity optimization can be reduced to a problem of iteratively solving a simplified version of the capacity problem. However, this iterative approach requires solving the maximum weighted independent set (MWIS) problem from graph theory. The MWIS problem has been extensively studied. Section IV-B reviews some of the work on solving the MWIS problem and presents several alternatives for computing the MWIS. Section V presents a technique to alleviate the problem of aggregate interference. These techniques are then employed to maximize the capacity of an urban mesh network in downtown Chicago. Section VI presents the results of this computational experiment. Finally, Section VII briefly discusses areas for future research and Section VIII presents concluding remarks.

II. Realistic mesh network modeling

One of the important aspects of capacity optimization is that in theory, determining the optimal capacity has a theoretical worst-case computational complexity that makes computing capacity even for small networks (e.g., 30 links) intractable with today's computing abilities. 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. For this reason, this examination employed the UDel Models [15]. Along with a realistic mobility simulator, the UDel Models includes a map builder, a realistic propagation simulator, and large collection of data and tracefiles. 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 [16]. 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 are available online. Many of the results shown or referred to here are derived from a simulation of a 2 km^2 (13×9 block) region of downtown Chicago. Figure 1 shows part of the region along with the placement of the mesh nodes. There are 500 mesh nodes in the entire region. Different topologies were considered a randomly selected 6×6 region and by reducing the node density by eliminating some of the mesh nodes.

It is quite clear that propagation has a significant impact on network performance, and hence it is expected that propagation impacts not only the network capacity, but also the performance of algorithms that compute capacity. Since this study utilizes realistic propagation in realistic urban regions, it is expected that the results presented here are directly applicable to the urban mesh networks currently being deployed.

III. Problem definition

A. Constraints

A STDMA schedule is a convex combination of assignments. An assignment is a specification of which links transmit and at what power, and bit-rate. If multiple antennas are used, then the assignment specifies the correlation between the antennas. An assignment is abstractly denoted as $\mathbf{v}$, but in the simple case of single antenna and no power control, an assignment specifies which link is transmitting, and hence $\mathbf{v}$ is a vector of zeros and ones with

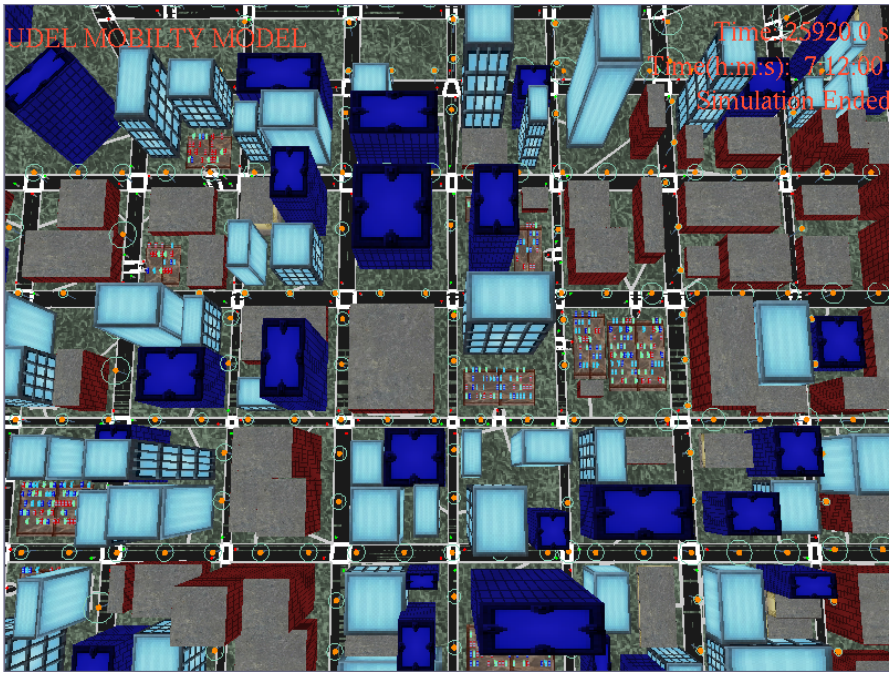

Fig. 1. A portion of map of simulated region used for determine the performance and behavior of capacity maximization techniques. Mesh routers are shown as orange dots with green circles. The full region is 13x9 blocks.

$\mathbf{v}_x = 1$ indicating that link x is transmitting, where all links are unidirectional. The bidirectionality of a physical link is represented by two unidirectional links. When the links transmit as specified by the assignment, the effective data rate across link x is denoted by $R(\mathbf{v}, x)$. (See In Section III-C, for further discussion of R and the effective data rate). Of course, if $\mathbf{v}$ specifies that link x is not transmitting, then $R(\mathbf{v}, x) = 0$.

A schedule is a convex combination of assignments. Specifically, a schedule is defined by $\{\alpha_{\mathbf{v}} : \mathbf{v} \in V\}$ where $\sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} \leq 1$, $\alpha_{\mathbf{v}} \geq 0$, and V is the set of all considered assignments. A schedule can be interpreted as a TDMA schedule, as a FDMA resource allocation, or a combination of both. In the TDMA case, the weights $\alpha_{\mathbf{v}}$ are the fraction of time that assignment $\mathbf{v}$ is used. Thus, if a schedule specifies that $\alpha_{\{1,0,1,0,0,1\}} = 2/3$ and $\alpha_{\{0,1,0,1,1,0\}} = 1/3$, then links 1, 3, and 6 simultaneously transmit for 2/3 of the time and links 2, 4, and 5 simultaneously transmit for 1/3 of the time. The schedule does not specify the TDMA period. However, if the period is long, then the time between when a particular link transmits might be long, resulting in high end-to-end delay. Note that this approach requires synchronization. Errors in synchronization can be accommodated with guard times, which reduces the total capacity. The impact of guard times on capacity can be quite large if the TDMA period is short. Thus, there is a trade-off between the end-to-end delay and the capacity wasted due to errors in synchronization.

In the FDMA approach, if the total bandwidth is 2.4 GHz to 2.42 GHz, and $\alpha_{\{1,0,1,0,0,1\}} = 2/3$ and $\alpha_{\{0,1,0,1,1,0\}} = 1/3$, then links 1, 3, and 6 continuously transmit on the bandwidth 2.4GHz-2.413GHz and links 2, 4, and 5 continuously transmit on the bandwidth 2.413GHz-2.42GHz. Since all links are continuously transmitting, this approach results in lower delay than TDMA. OFDMA radios can support flexible distribution of bandwidth. However, simultaneously transmitting and receiving on different but nearby frequency bands requires precise filtering and oscillators and can dramatically increase the cost of the radio.

Let f_ϕ be the data rate assigned to flow ϕ. The focus of this paper is on capacity of the mesh infrastructure. Thus, the source and destinations of the flows are gateways and mesh access points. Hence, flows may be the aggregate of several connections between various end-hosts.

Let $\mathcal{P}(\phi)$ be the set of links that flow ϕ traverses (i.e., flow ϕ's path). Thus, the aggregate of the demand data rate over link x is $\sum_{\{\phi: l \in \mathcal{P}(\phi)\}} f_\phi$. Under the schedule $\{\alpha_{\mathbf{v}} : \mathbf{v} \in V\}$, the data rate supplied over link x is $\sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} R(\mathbf{v}, x)$. Hence, a schedule must meet the link rate constraint

$$\sum_{\{\phi: l \in \mathcal{P}(\phi)\}} f_\phi \leq \sum_{v \in V} \alpha_v R(v, l) \text{ for all links } l. \tag{1}$$

B. Objective Functions

There has been considerable research that examines objective functions in the context of capacity maximization (e.g., [17], [18], [19]). Perhaps the most common is total network utility $\sum U_\phi(f_\phi)$, where U_ϕ is an increasing function. There are significant computational advantages if U_ϕ is convex. A popular example is $U_\phi(f_\phi) = w_\phi \log(f_\phi)$ where w_ϕ is an administrative weight that can be either set statically (e.g., depending on the service level agreement for the end hosts of flow ϕ) or may be allowed to vary according to changing traffic demands. In either case, the capacity optimization problem is

$$\max \sum U_\phi(f_\phi) \tag{2}$$

$$\text{such that: } \sum_{\{\phi: x \in \mathcal{P}(\phi)\}} f_\phi \leq \sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} R(\mathbf{v}, x) \text{ for all links } x \tag{3}$$

$$\sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} \leq 1 \tag{4}$$

$$\text{and } 0 \leq \alpha_{\mathbf{v}}.$$

The utility-based approach is general, and in cases such as $U_\phi(f_\phi) = w_\phi \log(f_\phi)$, the optimal flow rates satisfy certain fairness conditions [19]. However, a drawback is that the objective nonlinear significantly increasing the computational complexity.

Maximizing the minimum bit-rate is an alternative that results in a linear programming problem. Specifically,

$$\max F \tag{5}$$

$$\text{such that: } F \leq w_\phi f_\phi \text{ for all } \phi$$

$$\sum_{\{\phi: x \in \mathcal{P}(\phi)\}} f_\phi \leq \sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} R(\mathbf{v}, x) \text{ for all links } x \tag{6}$$

$$\sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} \leq 1 \tag{7}$$

$$\text{and } 0 \leq \alpha_{\mathbf{v}}.$$

Our experiments indicate that (5) can be solved in approximately a tenth the time required to solve (2). However, a drawback of (5) is that capacity is dominated by the data rate that the slowest flow can support. There are two ways to accommodate this problem. First, an iterative approach can be used to solve the max-min flow problem [19]. Specifically, first (5) is solved. Then, the slowest flows are identified. Let S be the set of slowest flows, i.e., $\theta \in S$ implies that $w_\theta f_\theta^* \leq w_\phi f_\phi^*$ for all flows ϕ where f_ϕ^* is the optimal flow rate found from solving (5). Then, (5) is modified

$$\max F$$

$$\text{such that: } F \leq w_\phi f_\phi \text{ for all } \phi \notin S$$

$$w_\phi f_\phi \geq w_\phi f_\phi^* \text{ for all } \phi \in S$$

$$\sum_{\{\phi: x \in \mathcal{P}(\phi)\}} f_\phi \leq \sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} R(\mathbf{v}, x) \text{ for all links } x$$

$$\sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} \leq 1 \text{ and } 0 \leq \alpha_{\mathbf{v}}.$$

Thus, a second optimization is performed to maximize the nonbottleneck flows subject to the constraint that the data rate allocated to the bottleneck flows is not decreased. This second optimization can be used to identify a second set of bottleneck flows and the process can be repeated. While this approach has the advantage that it yields

max-min data rates, we have found that in realistic mesh networks, no significant change in data rates occur after the first iteration. The reason for this is that all links share the same bandwidth and no flow is able to utilize the entire bandwidth of any link (at least not when networks are dense). That is, all flows share the bandwidth. Hence, at optimal capacity, it is not possible to increase the data rate of one flow without reducing the data rate of another.

A second way to reduce the impact of bottleneck flows is to mimic the idea of utility maximization. For example, one approach is to maximize capacity under the constraint that each flow gets the same air time, where the air time of a flow is the average transmission time required to deliver a bit from the source mesh node to the destination mesh node. The air time of flow ϕ is denoted T_ϕ with $T_\phi = \sum_{x \in \mathcal{P}(\phi)} \tau_x$, where τ_x is the average time it takes to transmit a data bit across link x. Note that τ_x depends on the schedule. The optimal capacity subject to air time constraints can be solved by solving a series of LP problems,

$$\left\{ f_\phi^{k+1} \right\} = \arg\max F$$

$$\text{such that: } F \leq T_\phi^k f_\phi$$

$$\sum_{\{\phi : x \in \mathcal{P}(\phi)\}} f_\phi \leq \sum_{v \in V} \alpha_v R(v, l) \text{ for all links } l$$

$$\sum_{v \in V} \alpha_v \leq 1 \text{ and } 0 \leq \alpha_v$$

where $T_\phi^k = \sum_{x \in P(\phi)} \tau_x^k$ with τ_x^k the average time required to transmit a bit across link x under the schedule found from the kth iteration, with $T_\phi^1 = 1$.

It is straightforward to extend the above to multipath case, i.e., where a single flow ϕ is split across a set of sub-flows N_ϕ flows. In this case, the sub-flows associated with flow ϕ are denoted $\{(\phi, n) : n = 1, 2, ..N_\phi\}$. Thus, the data rate for flow ϕ is the sum of the data rate over each sub-flow, i.e., $f_\phi = \sum_{n=1}^{N_\phi} f_{\phi,n}$, where $f_{\phi,n}$ is the data rate along the sub-flow (ϕ, n). Then (2) becomes

$$\max \sum U_\phi \left(\sum_{n=1}^{N_\phi} f_{\phi,n} \right)$$

$$\text{such that: } \sum_{\{(\phi,n) : x \in \mathcal{P}(\phi,n)\}} f_{\phi,n} \leq \sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} R(\mathbf{v}, x) \text{ for all links } x$$

$$\sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} \leq 1 \text{ and } 0 \leq \alpha_{\mathbf{v}}.$$

While splitting a single flow along multiple paths does increase the dimension of the optimization problem, if the number of paths is small, then the size of the problem is not significantly impacted. Rather, the number of links and the number of assignments has a much more significant impact on the computational complexity. Thus, to reduce notational clutter, this paper focuses only on the single path case.

C. Communication models

The communication model specifies the function $R(\mathbf{v}, x)$ and plays an important role in capacity maximization. Communication models are often classified as either protocol communication models or physical communication models. The defining feature of protocol communication models is that they are defined by a binary relation. Consequently, protocol communication models induce graphs and have significant theoretical and computational advantages over physical communication models. On the other hand, protocol communication models can be unrealistic. Specifically, a protocol communication model might predict that an assignment has a high data rate. But when applied in pratice, the assignment might have very low data rates. However, as will be shown in Section III-C, this problem is easily rectified. Thus, this paper focuses only on protocol models.

A protocol communication model defines rules that determine whether two links can transmit simultaneously. More abstractly, for each link x, a protocol communication model defines a set of conflicting links $\chi(x)$. If link $y \in \chi(x)$, then links x and y cannot transmit simultaneously. Different protocol communication models result in different sets of conflicting links. For example, the node exclusive protocol model stipulates that $y \in \chi(x)$ if either the receiver or transmitter nodes of link x are the same as either the receiver or transmitter node of link y. The disk communication model specifies that $y \in \chi(x)$ if either the receiver or transmitter nodes of link x are within d of the either the receiver or transmitter node of link y, where d is the communication range [20]. The disk model can be generalized to the two-hop disk model or, more generally, the k-hop disk model where the d in the disk model is replaced with $k \times d$ [21]. Arguably, the $SNIR$ protocol communication model is one of the most relevant. It specifies that $y \in \chi(x)$ if the signal strength from the transmitter of link x at the receiver of link y is above a threshold, or if the signal strength from the transmitter of link y at the receiver of link x is above a threshold. While there are a wide array of reasonable thresholds, one approach is to specify the threshold so that an SNIR constraint is met. Specifically, define $SNIR(y, x)$ to be the SNIR at the receiver of link x when both links x and

y are transmitting, i.e.,

$$SNIR\left(y,x\right) := \frac{H_{x,x}^{t,r} P_x}{H_{y,x}^{t,r} P_y + \mathcal{N}},$$

where $H_{x,x}^{t,r}$ is the channel gain from the transmitter of link x to the receiver of link x, $H_{y,x}^{t,r}$ is the channel gain from the transmitter of link y to the receiver of link x, P_x is the transmit power of link x, and $\mathcal{N}$ is the noise power. Thus, $H_{x,x}^{t,r} P_x$ is the signal strength at the receiver of link x, and $H_{y,x}^{t,r} P_y$ is the strength of the interference from y at x. The SNIR protocol communication model stipulates that $y \in \chi\left(x\right)$ if $SNIR\left(x,y\right) < \gamma_y$ or $SNIR\left(y,x\right) < \gamma_x$, where γ_y is the SNIR required for communication across link y for modulation scheme used by link y. It is common to set γ to the minimum value of the SNR required for the modulation scheme. However, noise and interference are quite different. We have performed extensive experiments and found that constants derived in that way are incorrect [22]. Nonetheless, for each modulation scheme there does exist a threshold γ, but the value of γ must be determined from detailed experiments of realistic simulation.

The protocol communication model also includes many aspects of the MAC. For example, the SNIR protocol model described above is more properly called the SNIR, without ACK, protocol communication model, as it does not consider transmission of ACKs. If ACKs are used, then the interference caused by ACKs must be considered. These ACKs could be transmitted shortly after the data packet, in bursts as in 802.11e, or the transmission of the ACKs could be determined by the schedule. Indeed, a wide array of MAC schemes can be considered by adjusting the details of the protocol model. We have found that ACKs can cause considerable interference and reduce the capacity by 20%. However, in the sequel, only the SNIR, without ACK, protocol model is considered.

Besides defining link conflicts, the communication model also defines the data rate. In the case of physical models, the data rate can be quite complex. But in the case of the protocol model, the bit-rate over link x is R_x as long as no links in $\chi\left(x\right)$ are also transmitting. Thus, when MIMO and power control are not used,

$$R\left(\mathbf{v},x\right) = \begin{cases} R_x \text{ if } x \in v \text{ and } \chi\left(x\right) \cap \left\{y \,|\mathbf{v}_y = 1 \text{ and } y \neq x\right\} = \varnothing \\ 0 \text{ otherwise} \end{cases} . \tag{8}$$

Clearly, the value of R_x depends on the modulation scheme used by link x. However, R_x also depends on the frame overhead (e.g., synchronization preambles, and physical and MAC headers). In the case that ACKs are used,

it is possible to transmit a packet several times before successfully receiving an ACK. In this case, R_x accounts for the average number of retransmissions[1].

Most of today's physical layer protocols support several bit-rates. Unfortunately, including this capability into capacity optimization may greatly increase the size of the problem. Suppose that link x can support modulation schemes[2] $M(1)$, $M(2)$, ..., $M(m)$. This physical link can be represented by m logical links, x_1, x_2, ..., x_m, with logical link x_i using modulation scheme $M(i)$. Of course, the logical links representing a single physical link would all be in conflict; a link can only transmit at one bit-rate at a time. Unfortunately, representing each physical link by m logical links increases the size of the problem by a factor of m. Heuristics can be developed in order to only include bit-rates that are likely to be useful for maximizing the capacity. That is, only a subset of the logical links x_1, x_2, ..., x_m are included. The best bit-rates to include remains an open question. In this paper, all bit-rates are used.

Along these same lines, today's radios support multiple transmit powers. These too can be included as logical links. This issue is discussed in Section VII.

IV. Computation of Capacity Maximizing Schedules

A. Assignment Iteration

Recall that an assignment $\mathbf{v}$ specifies which links are transmitting. If MIMO and power control are not used, then the assignment can be represented as a binary number where if the xth bit is one, then link x is transmitting. Clearly, there are 2^L assignments in the set of all assignments, where there are L links in the network. That is, if all assignments are considered, the set V in Section III-B has 2^L elements. Consequently, if V contains all possible assignments, then with current computational abilities, the optimization problems given in Section III-B cannot be solved for networks with more than 30 links.

An alternative approach is to not solve the problems of Section III-B for the full set of assignments, but use a reduced set. This reduced set of assignments is referred to as the *set of considered assignments*. In [6], the set of considered assignments was selected in an arbitrary way. Of course, selecting a set of considered assignments in this

[1]In 802.11, a transmission failure induces backoff. However, when transmissions are scheduled, backoff should be disabled.
[2]Associated with each modulation scheme is a bit-rate and γ.

Algorithm 1 Computing network capacity

0: Select an initial set of assignments $V(0)$, set $k = 0$.
1: Solve (2) (or (5)) for $V = V(k)$ and compute $\mu(k)$ and $\lambda(k)$, the Lagrange multipliers associated with constraints (3) (or (6)) and (4) (or (7)), respectively.
2: Search for an assignment $v^* \notin V(k)$ such that

$$\sum \mu_x(k) R(v, x) > \lambda(k). \tag{9}$$

if such an assignment is found **then**
set $V(k+1) = V(k) \cup v^*$, set $k = k + 1$, and go to Step 4.
else
if no assignment *exists*, then stop, the optimal solution has been found.
end if
4: Remove any redundant assignments in $V(k)$. Then go to Step 1.

way provides no guarantee on how close the resulting schedule is to optimal. In [12], the considered assignments were selected according to a heuristic. For example, assignments with links that are in conflict should not be included in the set of considered assignments. Unfortunately, no simple heuristic has been found that guarantees that the set of considered assignments results in the optimal capacity and that the size of the set considered assignments is small enough so that the problem is tractable.

The idea of considering only a subset of all assignments is justifiable in that it may result in the optimal capacity. Specifically, it can be proved that there exists a set of no more than L assignments such that if the set of considered assignments includes these L assignments, then the schedule that results will yield the same performance as the optimal schedule found by optimizing over all assignments [23] Thus, the main challenge facing capacity maximization is to determine these L assignments. Algorithm 1 is an iterative approach that searches for these L assignments.

The intuition behind Algorithm 1 is as follows. For each link there is a constraint such as (3). Each of these constraints yields an associated Lagrange multiplier. These Lagrange multipliers quantify the impact of adding and subtracting bandwidth to the corresponding link. Indeed, from sensitivity analysis of Lagrange multipliers (page 307 [24]), the Lagrange multiplier exactly measures how much the objective function would change if bandwidth were added to the corresponding link. The knowledge of which links are most important allows one to construct good assignments, e.g., find assignments that give bandwidth to important links.

A geometric interpretation is also useful. Consider the highly simplified case of two links as shown in Figure

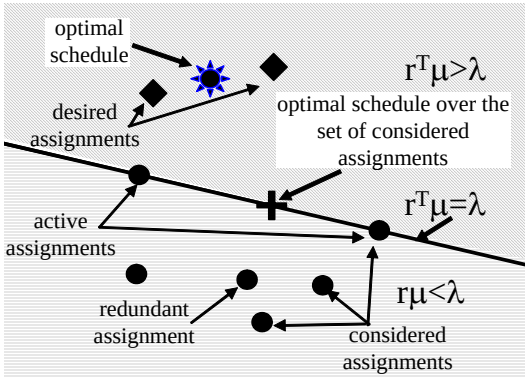

Fig. 2. A geometric view of the optimal scheduling problem. The above shows the space of all data rates, where we assume that there are only two links, and hence the space of all data rates is the plane. The Lagrange multipliers found from optimizing over the set of considered assignments divide the space of data rates, r, into two regions, according to whether $r^T\mu \leq \lambda$ (shown as the lower region in the figure above) or $r^T\mu > \lambda$ (shown as the upper region in the figure above). The active assignments and the schedule found by optimizing over the considered assignments are on the boundary of this division. An assignment will only improve the performance if the vector of data rates associated with the assignment satisfies $r^T\mu > \lambda$. For example, in the figure above, the assignments that compose the optimal schedule are currently in the region $r^T\mu > \lambda$.

2. Thus, the space of all combinations of link data rates (feasible or not) is the 2-D plane. An assignment yields a data rate for each link, and hence is a point in the space of all data rates. As mentioned in Section III-A, a schedule is a convex combination of assignments. Thus, the schedule can produce any combinations of bit-rates within the convex hull of the set of considered assignments. In any nontrivial scenario, the optimal schedule will always be on the edge of this convex region.

From the economic interpretation of Lagrange multipliers, $\sum_x R(\mathbf{v}, x)\,\mu_x$ is the revenue generated by assignment $\mathbf{v}$. All assignments used by the schedule also generate revenue λ, and hence, the schedule generates revenue λ. Assignments not used in the schedule generate less revenue than λ. Thus, the Lagrange multipliers found while optimizing over the set of considered assignments divide the space of all data rates into two regions based on whether the data rates generate more revenue than the schedule (See Figure 2). If the current schedule is not optimal, then the optimal schedule provides more revenue than the current schedule. Hence, there must be some assignments that provide more revenue than the current schedule. Searching for an assignment such that $\sum_x R(\mathbf{v}, x)\,\mu_x > \lambda$ is equivalent to searching for an assignment that is in region of the space of all data rates where none of the currently considered assignments are (See Figure 2).

B. Searching for new assignments

In light of the above, it is clear that it is possible to convert the challenge of maximizing capacity from one where the dimension of the set of considered assignments, V, is intractably large, to one where the size of V and solving (5) or (2) is not the concern, but to where the main challenge is finding assignments such that $\sum_x R(\mathbf{v}, x)\,\mu_x > \lambda$. In most cases, it is not necessary to find the assignment that maximizes $\sum_x R(\mathbf{v}, x)\,\mu_x$, but just one that satisfies (9). On the other hand, if no assignment exists, then the schedule found from the currently considered assignments yields the optimal capacity. Thus, in order to ensure that the currently considered assignments result in the optimal capacity,

$$\max_{V} \sum_{x} R(\mathbf{v}, x)\,\mu_x \tag{10}$$

must be solved.

Unfortunately, in general, solving (10) is NP-hard. However, as shown shortly, in the case of the protocol communication model, solving (10) is equivalent to a graph theoretic problem known as the maximum weighted independent set (MWIS) problem, which has been extensively studied. Thus, for the cases where there are a moderate number of links, there are a large number of algorithms that efficiently solve (10). For networks with more links, there are a large set of algorithms that approximately solve (10). And finally, there exist extensive theory regarding the solutions and solvability of (10). The richness of the theory of solving (10) when the protocol communication model is used is the most significant advantage of the protocol model over the physical model. Of course, the fact that protocol models result in schedules that yields very low data rate when actually deployed must be addressed (see Section V).

The connection between solving (10) and the maximum weighted independent set problem is made by the *conflict graph*. As discussed in Section III-C, for each link x, the protocol communication model defines a set conflicting links $\chi(x)$. The sets of conflicting links induce the conflict graph 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 $y \in \chi(x)$ (note, the we require that if $y \in \chi(x)$,

then $x \in \chi(y)$).

An assignment is a selection of links, or equivalently, an assignment is a selection of vertices from the conflict graph. In order for data to be delivered over the selected links, none of the vertices selected can be neighbors in the conflict graph. In graph theory, a selection of vertices where none of the selected vertices are neighbors is referred to as an independent set or a stable set. We denote selections of vertices as vectors $\mathbf{v}$ of zeros and one, with $\mathbf{v}_x = 1$ implying that vertex x is selected.

From the conflict graph, the weighted conflict graph is constructed by assigning the weight $R_x\mu_x$ to each vertex x. Note that from (8), the data rate across link x is R_x as long as no link in $\chi(x)$ is transmitting. Therefore, the total weight of an independent set is $\sum_{\{x:\mathbf{v}_x=1\}} R_x\mu_x$, which coincides with the revenue generated by the assignment $\mathbf{v}$. The problem of solving (10) is equivalent to selecting an independent set that maximizes the total weights assigned to the selected vertices; this is precisely the maximum weighted independent set (MWIS) problem.

As mentioned, in general, MWIS problem is NP-hard. Indeed, it is one of the first problems to be shown to be NP-hard [25]. However, while it is NP-hard in general, it is not always the case. If the conflict graph is a perfect graph, then the MWIS can be found in polynomial time [26]. The class of perfect graphs includes a wide variety of graphs (See pages 279-283 in [26]). For example, the interval graph is a perfect graph. In an interval graph, each vertex is associated with an interval of the real number line. Two vertices are neighbors if their corresponding intervals overlap. It is not hard to show that if the network is restricted to one dimension (e.g., a network along a road), then the conflict graph that results from protocol communication models is an interval graph. A 2-D version of the interval graph is the disc graph, which, while not a perfect graph, allows the maximum weighted independent set to be computed in polynomial time [27]. Besides perfect graphs, maximum independent sets can be computed in polynomial time for several other classes of graphs., for example, for "claw-free" graphs [28], for fork-free graphs [29], for trees (see [30] for a linear time algorithm), for sparse random graphs (see [31] for a linear time algorithm), and for circle graphs (see [32] for a linear time/space algorithm). Unfortunately, besides the interval graph, it is unknown whether these graphs typically arise in mesh networks. However, it does give hope that the MWIS problem can be solved.

There has been considerable effort focused on computing the MWIS in the general case. These research efforts

tend to take one of two approaches. In one approach, the goal is to develop algorithms that have good worst-case performance. For example, [33] reports an algorithm with worst-case time complexity of $O\left(1.2461^{L}\right)$, while [34] achieves $O(1.2431^{L})$.

One drawback of the worst-case analysis is that most graphs can be solved much faster than predicted by the worst-case analysis. And hence, another approach is to not optimize the worst-case performance, but focus on the time to find the MWIS in graphs that have been deemed interesting. Computational methods that work well in this respect can be found in [35], [36], [37], [38], and [39], where [39] is sometimes cited as the current state-of-the-art.

Another way to compute the MWIS is to use mixed integer programming. Specifically, the MWIS problem can be written as

$$\begin{aligned} &\max_{\mathbf{v}} \sum R_x \mu_x \boldsymbol{\nu}_z \\ &\text{such that: } \mathbf{v}_x + \boldsymbol{\nu}_y \leq 1 \text{ if } y \in \mathcal{X}(x) \\ &\mathbf{v}_x \in \{0,1\}. \end{aligned} \tag{11}$$

Note that since $v_x \in \{0,1\}$, this problem can also be solved with binary programming. There are many commercially available mixed integer and binary programming tools (e.g., CPLEX [40]).

Our work has found that these exact methods work well in practice. Nonetheless, approximate solutions to the MWIS are also of interest since, in many cases, they can quickly provide assignments that pass Step 3 of Algorithm 1, and in some cases they find the exact MWIS. However, one should not expect too much from approximate solutions since it is NP-hard to approximate the MWIS with an approximation ratio that is better than $L^{1-\varepsilon}$ for any $\varepsilon > 0$. Thus, in general, the worst-case, the total weight of a polynomial time approximation is a factor of L less than the total weight of the MWIS. However, as discussed above, the worst-case performance may be considerably worse than what typically occurs in conflict graphs that arise from mesh networks.

Finding approximate solutions to the MWIS problems is also an active area of research (e.g., see [41] for a review of some methods). An algorithm suggested by Kako [42] is simple and often provides good results; it achieves a worst-case approximation ratio of $\bar{d}$, where $\bar{d}$ is the average weighted degree, i.e., $\bar{d} = \frac{\sum_x W(x)}{\sum_x R_x \mu_x}$, where $W(x)$ is

the weighted degree of vertex x and is given by $W(x) := \frac{\sum_{y \in \chi(x)} R_y u_y}{R_x \mu_x}$. Kako's algorithm is a greedy optimization that builds the independent set by selecting the vertex with least weighted degree. Once selected, the vertex and all the neighbors of the vertex are removed from the graph. The weighted degree is recomputed for all remaining vertices. The vertex with least weighted degree is added to the independent set and the process continues until there are no more vertices in the graph.

Another algorithm that has been found to perform well in realistic mesh networks is the WMIN algorithm developed in [43]. WMIN has approximation ratio Δ, the maximum degree. Other algorithms have approximation ratios $\left(\bar{d}_w + 1\right)/2$, $O\left(\bar{d}_w \log\log \bar{d}_w / \log \bar{d}_w\right)$, and $O\left(\delta_w \log\log \delta_w / \log \delta_w\right)$, where δ_w is the weighted intuitiveness of the graph [42]. See [41] for further discussion of some approximation algorithms for the MWIS problem.

One drawback of using approximate solutions to the MWIS problem is that if the approximation scheme does not result in a better assignment, then, in general, it is not possible to determine whether there does not exists a better assignment (and hence the current schedule is optimal) or there does exists a better assignment, but the approximation algorithm is unable to find it. While in general, such a determination is not possible in polynomial time, as discussed next, in some cases it is possible, and, indeed, in all the realistic mesh network we have examined, such a determination is possible in polynomial time.

Two important metrics of weighted graphs are ω, the total weight of the MWIS and $\mathcal{K}$, the weighted chromatic number. In general, both numbers are NP-hard to compute. However, the Lovász number, $\mathcal{V}$ can be computed in polynomial time and $\omega \leq \mathcal{V} \leq \mathcal{K}$. In some cases (e.g., in the case of a perfect graph), $\omega = \mathcal{V}$. Thus, in such cases it is possible to compute ω in polynomial time. Therefore, in general, if an approximation method fails to find a new assignment with revenue greater than λ, then one can compute the $\mathcal{V}$ in polynomial time and check whether $\lambda = \mathcal{V}$. If $\lambda = \mathcal{V}$, then the optimal schedule has been found. On the other hand, since $\mathcal{V}$ is only an upper bound on ω, $\lambda < \mathcal{V}$ does not imply that the assignment found by the approximation is not optimal. However, we have found that for the realistic mesh networks examined, at convergence (i.e., after Algorithm 1 has converged), we have $\lambda = \mathcal{V}$. Unfortunately, while it is possible to compute $\mathcal{V}$ in polynomial time, it does take a considerable amount of time and we have been unable to confirm if $\omega = \mathcal{V}$ for networks with more than 100 links.

Figure 3 compares the average computation time for computing an (approximate) maximum weighted independent

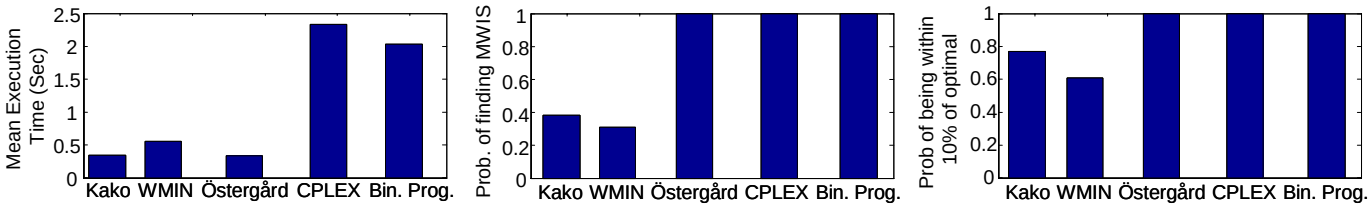

Fig. 3. Performance of techniques to compute the maximum weighted independent set. Left: The computation time on a AMD Athlon 64 FX-53 PC. The network had 500 links. Center: The probability of finding the MWIS. Östergård's methods, CPLEX, and binary programming are exact methods, and hence always compute the exact MWIS. Right: The probability of the found independent set having a weight within 10% of optimal.

set for a 500 link realistic network[3]. We see that the exact method develop by Östergård performs the best, even surpassing approximate solutions. However, the approximate algorithms were implemented in Matlab, whereas the other methods were implemented in C, hence the comparison is slightly unfair. Nonetheless, Figure 3 confirms that the MWIS problem is tractable for moderate sized networks.

V. Correcting Protocol Communication Models

The previous section indicates that there are a wealth of techniques to compute optimal capacity for mesh networks when protocol communication models are considered. However, such models ignore the aggregate of interference. For example, suppose that $y \notin \chi(x)$, $z \notin \chi(x)$, and $y \notin \chi(z)$. Thus, according to the protocol communication model, all three links can transmit simultaneously. In the case of the SNIR protocol communication model, this implies $SNIR(y,x) > \gamma_x$, $SNIR(z,x) > \gamma_x$. However, it is possible that

$$\frac{H_{x,x}^{t,r} P_x}{H_{y,x}^{t,r} P_y + H_{z,x}^{t,r} P_z + \mathcal{N}} < \gamma_x. \tag{12}$$

Hence, the aggregate interference from both y and z makes communication across link x impossible. Consequently, a schedule that uses an assignment with links x, y, and z simultaneously transmitting will not achieve the predicted performance when deployed in practice. One alternative is to use a physical communication model. However, the drawback of the physical communication model is that much of the machinery developed for the maximum weighted set cannot be directly applied. Thus, we seek to repair the protocol model.

The root of the problem is that the protocol communication model only considers binary conflicts, whereas physical conflicts such as (12) result from multiple links simultaneously transmitting. Such conflicts are referred to

[3]The largest mesh network currently deployed is Corpus Christi with 300 mesh routers [14].

as *multi-conflicts*. A naive use of the protocol model ignores multi-conflicts. However, this problem can be solved by identifying assignments with multi-conflicts and removing them from consideration. A brute force approach is to identify all assignments that have multi-conflicts and remove them. However, such an approach is computationally complex. Instead, when a new assignment is made (in Step 2 of Algorithm 1), it is checked whether it has any multi-conflicts. Specifically, suppose the newly found assignment is $\mathbf{v}$. This assignment contains a multi-conflicts if

$$\frac{H_{z,z}^{t,r} P_z \mathbf{v}_z}{\sum_y H_{y,z} P_y \mathbf{v}_y + \mathcal{N}} < \gamma_z.$$

Upon identifying multi-conflicts, assignments with the multi-conflict must be removed from consideration when searching for new assignments. Techniques to do this are discusses shortly. Once the assignment with multi-conflicts is removed, a new assignment is generated. This assignment is also tested for multi-conflicts. If there are any, they are removed, and the process continues until an assignment is generated that has no physical conflicts. This assignment is then added to the set of considered assignments and a new schedule is created. Thus, Step 2 of Algorithm 1 is replaced with generating a new assignment that is free from physical conflicts, and this step may involve several attempts at finding a conflict-free assignment.

There are two ways that multi-conflicts can be removed. One way is a conservative approach that might reduce the final computed capacity. The second way is not conservative, but requires a slight departure from the MWIS setting, and must be solved with the mixed integer or binary programming.

In the first approach, a multi-conflict is removed by adding a binary conflict. Suppose that vertices in the assignment with the multi-conflict are z and $y_1, y_2, ...y_M$, where we assume that the multi-conflict makes transmission over link z impossible. Without loss of generality, assume that the links y_i are ordered such that $H_{y_i,z}^{t,r} \geq H_{y_{i+1},z}^{t,r}$ (i.e., link y_i causes more interference than link y_{i+1}). The smallest set of vertices that cause the conflict is y_1, y_2,..., y_{m^*}, where

$$m^* = \min \left\{ m \leq M \left| \frac{H_{z,z}^{t,r} P_z}{\sum_{i=1}^{m} H_{y_i,z} P_{y_i} + \mathcal{N}} < \gamma_z \right. \right\}.$$

Then, from this set of vertices, find the one with the least weight, i.e., $i^* = \arg\min_{i \leq m^*} R_{y_i} \mu_{y_i}$. Finally, add binary conflicts such that y_{i^*} and z do not simultaneously transmit, i.e., the sets of conflicting links are updated via

$\mathcal{X}(z) = \mathcal{X}(z) \cup y_{i^*}$ and $\mathcal{X}(y_{i^*}) = \mathcal{X}(y_{i^*}) \cup z$. With these new conflicts, a new conflict graph is made and a new set of assignments can be generated using the methods discussed in Section IV-B. Note that this approach might result in a decrease in performance; the multi-conflict is being approximated by a more restrictive binary-conflict. Also, since the construction of the added binary conflicts depends on μ, these conflicts are removed after a new μ is found.

A second way to remove multi-conflicts is to change the mixed integer programming problem (11) to

$$\begin{aligned} & \max_{v} \sum R_x \mu_x \nu_x \\ \text{such that:} \quad & v_x + \nu_y \leq 1 \text{ if } y \in \mathcal{X}(x) \\ & v_z + v_{y_1} + ... v_{y_m} \leq m \\ & v_x \in \{0, 1\}, \end{aligned} \tag{13}$$

where, as above, it is assumed that the multi-conflict is from the combination of links $z,\ y_1, ..., y_m$. Note that condition (13) ensures that these links will not all be simultaneously selected. As additional multi-conflicts are identified, more such conditions are added. Unlike the previous approach, this approach is not conservative.

Figure 4 shows the computed capacity when the multi-conflicts are ignored, when they are repaired by adding binary-conflicts, and when they are repaired by adding multi-conflicts. Note that all three approaches yield nearly the same computed capacity. The fact that the capacity found when multi-conflicts are ignored is the same as the capacity when multi-conflicts are repaired does not mean that the multi-conflicts can be ignored. Figure 4 also shows the actual capacity when multi-conflicts are ignored; clearly, ignoring multi-conflicts can dramatically reduce the actual performance of the computed schedule. On the other hand, either of the methods described above can eliminate multi-conflicts in a way that has a negligible impact on the computed capacity. The fact that the two methods for repairing physical conflicts perform equally well implies that all machinery for computing the MWIS can still be exploited while ensuring that the resulting schedule delivers the expected capacity when applied to the physical network.

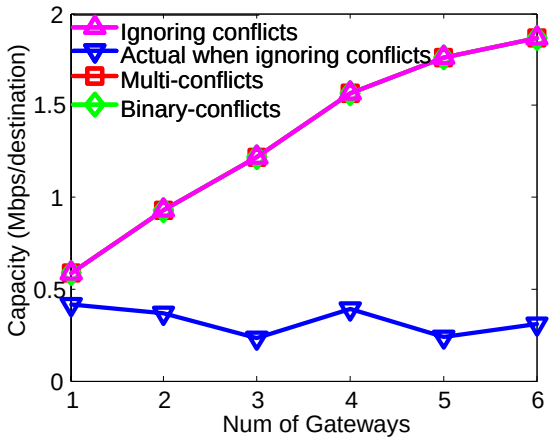

Fig. 4. Actual and computed performance of the protocol communication model. A naive use of the protocol communication model ignores physical conflicts due to multi-link conflicts. Consequently, the actual capacity of the schedule is substantially less than the computed capacity. Physical conflicts can be eliminated by adding binary conflicts or multi-conflicts. In the figure above, the computed capacity when multi-conflicts are ignored, the capacity when multi-conflicts are repaired by adding binary-conflicts, and the capacity when multi-conflicts are repaired by adding multi-conflicts are all nearly the same, and hence only a single line is shown. The topologies used to make this figure are discussed in Section VI-A.

VI. Capacity of urban mesh networks

A. Experimental setup

In order to examine the capacity of urban mesh networks, a large set of simulated urban mesh networks were generated. Each network was based on a 6×6 block region of downtown Chicago that was randomly selected from a 2 km^2 region. Figure 1 shows part of this region. As discussed in Section II, the radio propagation was determined with the UDelModels [15].

In order to estimate the typical performance of an urban mesh network, several different types of topologies were generated, and ten trial topologies were generated for each topology type. The topology types are characterized by the number of wireless mesh routers and the number of wired gateways. In these experiments, all traffic flowed from gateways to destinations, where each mesh router in the topology was a destination of a flow. The impact of routing on capacity is outside the scope of this investigation. Instead, packets were forwarded to their destination over least hop paths. 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 worst than any route from any other gateway. Mesh routers and gateways were assumed to be placed on lampposts. Apart of this restriction, mesh routers were uniformly spread throughout the region. The simulated area was partitioned into equal sizes

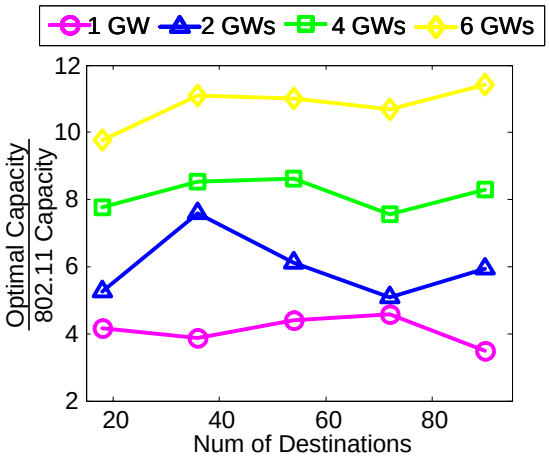

Fig. 5. Average ratio of optimal capacity and the capacity provided by 802.11 CSMA/CA with RTS/CTS.

regions where the number of regions is the same as the number of gateways. A gateway was randomly located within each region.

Finally, the number of nodes ranged from 18 to 90 so that the average number of nodes per block ranged from 0.5 to 2.5 in steps of 0.5. The number of gateways ranged from 1 to 6. Since 10 samples of each topology were generated, a total of 300 topologies were used.

Here the optimal capacity generated by including all possible modulation schemes supported by 802.11a and accounting for physical conflicts with the multi-conflict approach. The optimal schedule resulted from solving (5) (recall that in practice this gives the max-min fair data rate).

B. Improvement over 802.11 CSMA/CA

Here the optimal capacity given by solving (5) is compared to the capacity achievable with 802.11a with CSMA/CA, RTS/CTS, and automatic rate fallback (ARF) [44] (or 802.11 for short). In order to compute the capacity with 802.11, constant bit-rate traffic is sent at a fixed rate to each destination. For a particular sending rate, the arrival rate at each destination was computed, and the minimum arrival rate over all destinations was computed. This minimum is denoted $MN\,(SR)$, where SR is the sending rate. The sending rate was varied in order to determine $\max_{SR} MN\,(SR)$. This maximum is the 802.11 capacity for the topology. Note that for each sending rate, repeated trials were performed so that the confidence intervals (computed via bootstrapping [45]) ensured that error in the estimated capacity was less than 10% with 95% confidence.

Figure 5 shows the ratio of the optimal capacity to the 802.11 capacity. For a high density of gateways (e.g., one gateway per six blocks), optimal scheduling provides an order of magnitude improvement over 802.11. On the other hand, for networks with a low density of gateways (e.g., one gateway per 36 blocks), capacity optimization provides an improvement of approximately a factor of four. While these improvements are substantial, they are idealized. In practice, the actual performance will likely be reduced due to factors such as synchronization errors, uncontrolled interferers, and time-varying or incorrectly measured channel gains. Nonetheless, with effort to overcome these problems, it is reasonable to expect substantial improvements in capacity even in practice.

C. The Impact of the Topology on Capacity

Figure 6 (a) shows the optimal capacity (Mbps/destination) for a wide range of topologies. The general trends are expected. As the number of destinations increases, the data rate per destination decreases and as the number of gateways increases, the data rate per destination increases. Recall that there are 36 blocks in the simulated region. Thus, Figure 6 (a) shows that if there is one gateway for every six blocks, and 2.5 mesh routers/destinations for each block (i.e., 90 destinations in 36 blocks), then the infrastructure can provide approximately 1.5Mbps per destination.

Figure 6 (a) shows that the capacity increases as base stations are added. A more detailed view of this behavior is given in Figure 6 (b). This figure shows that as more destinations are added, the impact of adding more gateways increases, however, it increases rather slowly. It is important to note that power control is not used. It is possible that power control will have a significant impact when the node density is high.

Figure 6 (c) shows the average utilization of each gateway in the network. Ideally, each gateway transmits at all times. When there is only a single gateway, this ideal is nearly achieved. Recall that the communication model used here does not utilize ACKs; only one-way communication is required. Note that with TDMA, even when packets must be transmited over multiple hops, the gateway is able to nearly continuously transmit. For example, suppose that the gateway transmits to node A during the first time slot and to node B during the second time slot. Furthermore, during the second time slot, node A forwards the packet received during the first time slot to its neighbor, node C. If node C is far enough away from the gateway and node B is far enough away from node A, then all transmissions will succeed. Of course, if node A requires an ACK from node C, this ACK would likely

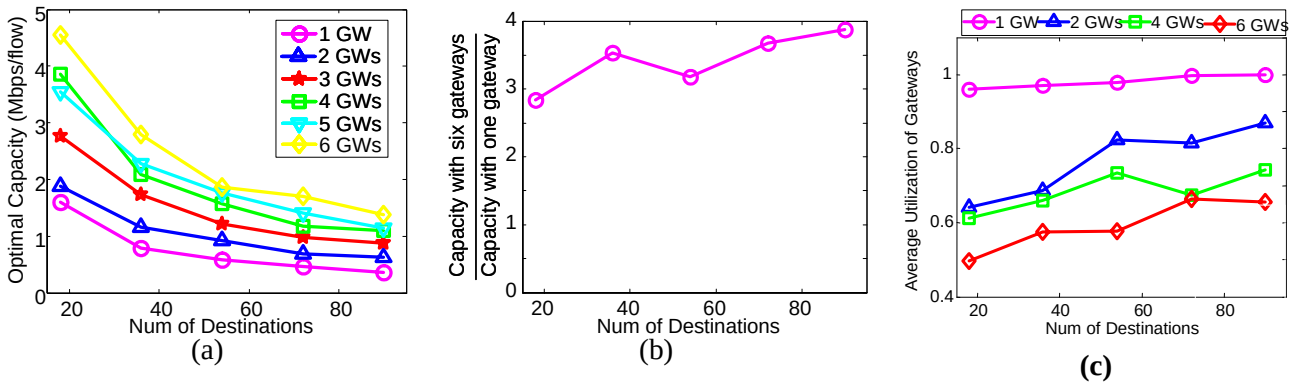

Fig. 6. (a) Capacity of mesh network infrastructure in a 6Œ6 block region for various numbers of wired gateways and various numbers of mesh routers/destinations. (b) Ratio of the capacity with six wired gateways and the capacity with one wired gateway. (c) Average utilization of each gateway.

not be received since the interference from gateway's transmission to node B would likely be substantial. Thus, ACKs may decrease the network capacity.

VII. Future Work and Other Issues

The techniques described above can be used to compute optimal schedules. However, several challenges remain. As mentioned above, factors such as synchronization, uncontrolled interferers, and time-varying or incorrectly measured channel gains must be addressed. As described next, transmit power, MIMO, flow control, and schedule computation and distribution are also relevant to capacity maximization.

A. Transmit power

The techniques above do not efficiently accommodate power control. It is possible to include logical links that represent a physical link transmitting at different powers. However, as mentioned in Section III-C, this approach greatly increases the computational complexity. Another approach is to directly maximize the revenue generated by varying the transmit powers, i.e., maximize (10) where the maximization is not only over all assignments, but also over all transmit powers. It was shown in [46] and [47], that such optimizations are convex when the SNR is high enough. This means that once it has been decided which links should transmit, if the SNR for each link is high (which is the typical case in mesh networks), then it is possible to optimize the transmit powers. However, since links that are not transmitting have low SNR, the optimization of transmit power will not change which links transmit. Thus, some other technique must be used to determine which links should transmit. Consequently, maximizing capacity with power control remains an open problem.

B. MIMO

MIMO offers tremendous improvements in data rates. It also adds flexibility, and hence has the potential to greatly increase the computational complexity. Specifically, when MIMO is used, an assignment not only specifies whether the link is transmitting, but also, for the links that do transmit, the assignment specifies a $N \times N$ non-negative definite matrix that defines the correlation of the transmitted signal over the N transmit antennas. By adjusting this matrix, the transmission can act to increase the data rate or reduce interference. As a result, if N is large, then full optimization is intractable for even small networks. However, in [48] it is shown that if the link SNR is large (as it typically is in mesh networks), then full optimization will provide little improvement in performance; it is possible to simply treat each MIMO link as a SISO high data rate link.

C. Flow Control

Section III-B presented several capacity optimization problems. In all cases, a schedule is designed in order to maximize some function of the flow data rates. While it is conceivable that network operators could deploy such a scheduling scheme, it is unclear how the flow data rates can be controlled to meet the data rates offered by the network. There has been effort in using packet drops to cause TCP flows to adjust their data rate when the objective is a specific type of network utility [18]. However, it is unclear how to control other connection types such as VoIP and multimedia streaming, which are expected to be important applications in mesh networks.

D. Schedule Computation and Distribution

The discussion above did not address where and when the schedules are determined. There has been extensive effort focused on distributed techniques to compute schedules (e.g., [49], [10], and [50]). However, these techniques perform poorly in terms of convergence rate and overhead [51]. In their current form, they are not suitable for mesh networks. On the other hand, there does not appear to be any fundamental reason that schedules cannot be determined in a distributed fashion. Clearly, this area requires further effort.

VIII. Conclusion

This paper provides techniques and insight on capacity maximization of the mesh infrastructure. With these techniques, optimal capacity is achievable. While capacity optimization can be computationally intractable for even

moderate size networks, the techniques presented here have been successfully applied to 500 link realistic networks (the largest infrastructure network currently provided by the UDelModels realistic mesh network simulator). This optimization technique has two key parts. First, the computational approach focuses on solving a simplified capacity optimization problem. The solution to this problem is used to improve the relevance of the simple capacity problem, which is solved again. This iterative process continues until convergence, at which point the simple capacity optimization provides the capacity of the full network. A second key aspect of the technique described is that the method leverages the huge body of research focused on the graph theoretic problem of finding the maximum weighted independent set. The wealth of results on this problem have yet to be fully exploited.

REFERENCES

[1] T. Salonidis and L. Tassiulas, "Distributed dynamic scheduling for end-to-end rate guarantees in wireless ad hoc networks," in *MobiHoc*, 2005, pp. 145–156.

[2] B. J. Wolf, J. L. Hammond, and H. B. Russell, "A distributed load-based transmission scheduling protocol for wireless ad hoc networks," in *IWCMC*, 2006.

[3] J. Yackoski and C.-C. Shen, "Managing delay and jitter in mesh networks through path-aware distributed transmission scheduling," in *Mobicom*, 2006.

[4] J. Gronkvist and A. Hansson, "Comparison between graph-based and interference-BAsed STDMA scheduling," in *MobiHoc*, 2001.

[5] 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.

[6] 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.

[7] G. Sharma, R. Mazumdar, and N. Shroff, "On the complexity of scheduling in wireless networks," in *MobiCom*. ACM, 2006.

[8] X. Lin and S. Rasool, "A distributed and provably-efficient joint channel-assignment, scheduling and routing algorithm for multi-channel multi-radio wireless mesh networks," Purdue University, Tech. Rep., 2006.

[9] M. Chiang, "Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 104–116, 2005.

[10] J. Yuan, Z. Li, W. Yu, and B. Li, "A cross-layer optimization framework for multihop multicast in wireless mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2092–2103, Nov 2006.

[11] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, 1992.

[12] 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.

[13] R. Cruz and A. Santhanam, "Optimal routing, link scheduling and power control in multi-hop wireless networks," in *IEEE INFOCOM*, March 2003.

[14] T. Networks, "Corpus Christi case study," 2007, http://www.tropos.com/pdf/corpus$_c asestudy.pdf$.

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

[16] 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.

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

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

[19] L. Massoulie and J. Roberts., "Bandwidth sharing: Objectives and algorithms." *IEEE/ACM Transactions on Networking*, no. 3, June 2002.

[20] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM Press, 2005, pp. 73–87.

[21] G. Sharma, R. Mazumdar, and N. Shroff, "On the complexity of scheduling in wireless networks," in *Mobicom*, 2006.

[22] V. Sridhara, H. Shin, and S. Bohacek, "Performance of 802.11b/g in the interference limited regime," in *Submitted*, 2007.

[23] S. Bohacek and P. Wang, "Toward tractable computation of the capacity of multihop wireless networks," in *Infocom*, 2007.

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

[25] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum Press, March 1972, pp. 85–103.

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

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

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

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

[30] 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.

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

[32] G. Valiente, *A New Simple Algorithm for the Maximum-Weight Independent Set Problem on Circle Graphs*. Springer, 2003, vol. 2906, pp. 129–137.

[33] 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.

[34] F. V. Fomin, S. Gaspers, and S. Saurabh, "Branching and treewidth based exact algorithms," in *ISAAC*, 2006, pp. 16–25.

[35] L. Babel, "A fast algorithm for the maximum weight clique problem," *Computing*, vol. 52, pp. 31–38, 1994.

[36] 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.

[37] ——, "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.

[38] J. S. Warren and I. V. Hicks, "Combinatorial branch-and-bound for the maximum weight independent set problem," 2007.

[39] 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.

[40] ILOG, "CPLEX," http://www.ilog.com/products/cplex/.

[41] M. M. Halldórsson, *Approximation Algorithms for Combinatiorial Optimization*. Springer Berlin / Heidelberg, 2004, ch. Approximations of Independent Sets in Graphs, pp. 24–45.

[42] 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.

[43] 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.

[44] A. Kamerman and L. Monteban, *WaveLAN-II: A High-performance wireless LAN for the unlicensed band*. Bell Lab Technical Journal, 1997, pp. 118–133.

[45] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. Chapman and Hall/CRC, 1994.

[46] S. Ye and R. S. Blum, "Optimized signaling for MIMO interference systems with feedback," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2839–2848, 2003.

[47] M. Chiang, "Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control," *IEEE Journal of Selected Areas on Communications*, vol. 23, pp. 104–116, 2005.

[48] S. Bohacek, "On maximizing capacity in fixed mesh networks with MIMO links," in *Handbook of Wireless Mesh Sensor Networks*, G. Aggelou, Ed. McGraw-Hill, 2007, ch. On Maximizing Capacity in Fixed Mesh Networks with MIMO Links.

[49] 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.

[50] X. Wang and K. Kar, "Cross-layer rate control for end-to-end proportional fairness in wireless networks with random access," in *MobiHoc*, May 2005.

[51] P. Wang and S. Bohacek, “The practical performance of subgradient computational techniques for mesh network utility optimization,” in *NET-COOP.*, 2007.