Capacity Optimization in Wireless Mesh Networks

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

1 Introduction

Capacity optimization by optimizing transmission schedules of wireless networks has been an active area of research for at least 20 years. The challenge is that the space over which the optimization is performed is exponential in the number of links in the network. For example, in the simple SISO case where no power control is used and only one bit-rate is available, the optimization must be performed over a space of size 2^L where there are L links in the network. Thus, the optimization cannot be performed for even moderate sized networks of a few tens of links.

This abstract discusses recent advances that allow capacity maximization in realistic mesh networks. With these techniques, the maximum capacity of a 500 link network can be determined in approximately 6 minutes on a 2.8GHz PC. This represents a dramatic improvement over the techniques of [1] that can only be applied to networks with fewer than 16 links.

With tractable schedule optimization, it is possible to consider optimal routing. Again, a naive approach to optimal routing results in exponential computational complexity. However, using an approach that is somewhat similar to that presented in [1], it is possible to determine optimal routes. This abstract briefly explains how optimal routing is performed.

An important aspect of this work is that the computational techniques have been tested on realistic mesh networks generated by the UDelModels simulator [2]. Thus, the results presented here are directly applicable to the mesh networks currently being deployed.

2 Problem Definition

A spatial TDMA (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 (i.e., MIMO), then the assignment also 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 links are transmitting, and hence $\mathbf{v}$ can be represented as a vector of zeros and ones with $\mathbf{v}_x = 1$ indicating that link x is transmitting. It is assumed that all links are unidirectional. Bidirectional links are represented by two unidirectional links. For this abstract, we only consider the SISO case without power control and with only a single bit-rate for each link. It is possible to extend this approach to these other cases. In any case, when the links transmit as specified by the assignment, the effective data rate across link x is denoted by $R(\mathbf{v}, x)$. Note that the data rate across a link depends on $\mathbf{v}$, the assignment, that is, the data rate across a particular link depends on which other links are transmitting.

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 bandwidth 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. In the case of FDMA, the weights α_v is the amount of bandwidth (in Hz) allocated to assignment $\mathbf{v}$.

Let f_ϕ denote the data rate assigned to flow ϕ. The focus of this paper is on capacity maximization for 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.

A general form of capacity maximization is

$$\max U(\mathbf{f}) \tag{1}$$

$$\text{such that:} \quad \sum_{\{\phi : x \in \mathcal{P}(\phi)\}} f_\phi \leq \sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} R(\mathbf{v}, x) \text{ for all links } x \tag{2}$$

$$\sum_{\mathbf{v} \in V} \alpha_{\mathbf{v}} \leq 1 \tag{3}$$

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

where $\mathbf{f}$ is the vector of flows, U is the objective function, V is the set of consider assignments (See the next section for further discussion of the set of considered assignments), and $\mathcal{P}(\phi)$ is the set of links that flow ϕ traverses (i.e., flow ϕ's path). One approach is to set $U(\mathbf{f}) = \sum_\phi w_\phi \log(f_\phi)$ or more generally $U(\mathbf{f}) = \sum_\phi u_\phi(f_\phi)$, where u_ϕ is the a smooth concave increasing utility function [3, 4, 5]. An alternative is for $U(f) = \min_\phi(f_\phi)$. In this case, (1) can be redefined to be a linear programming problem. While $U(f) = \min_\phi(f_\phi)$ is not the same as max-min fairness, we have found that in realistic mesh networks, $U(f) = \min_\phi(f_\phi)$ results in max-min fairness.

Depending on U, there are several ways to solve (1). While the performance of different solvers varies, the basic challenge is that a naive approach results in the set V being too large for any computational technique. This problem is solved next.

3 Computation of Capacity Maximizing Schedules

3.1 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 2 has 2^L elements. Consequently, if V contains all possible assignments, then with current computational abilities, the optimization problems given in Section 2 cannot be solved for networks with more than approximately 30 links.

An alternative approach is to not solve the problems of Section 2 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. In [7], the considered assignments were selected according to a heuristic. Selecting a set of considered assignments in either of these ways does not provide any guarantees on how close the resulting schedule is to optimal. However, the idea of considering only a subset of all assignments is justifiable in that it may result in the optimal capacity.

Theorem 1 *There exists a set of assignments $\tilde{V}$ with L assignments such that the solution to (1) with V replaced with $\tilde{V}$ yields the same solution as if the maximization were performed over all possible assignments.*

Thus, the main challenge is not solving (1), but finding a good set of assignments V. Algorithm 1 finds such a set of assignments. The intuition behind Algorithm 1 is as follows. For each link there is a constraint (2). 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, 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. The geometric view shown in Figure 1 also provides insight into Algorithm 1.

3.2 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

Algorithm 1 Computing network capacity

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

$$\sum \mu_x(k) R(v, x) > \lambda(k). \tag{4}$$

if such an assignment is found **then**
set $V(k+1) = V(k) \cup v^*$, set $k = k + 1$, and go to Step 3.
else
if no assignment *exists*, then stop, the optimal solution has been found.
end if
3: Remove any redundant assignments in $V(k)$. Then go to Step 1.

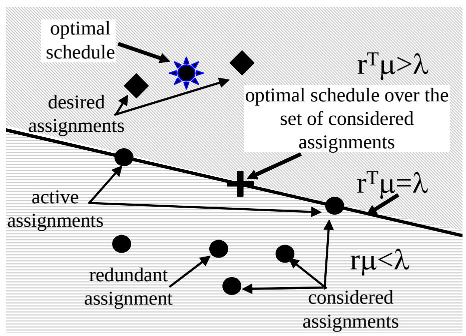

Figure 1. A geometric view of the optimal scheduling problem. The above shows the space of all data rates, where we assume that there are only two links, and hence the space of all data rates is the plane. The Lagrange multipliers found from optimizing over the set of considered assignments divides the space of data rates, r, into two regions, according to whether $r^T\mu \le \lambda$ (shown as the lower region in the figure above) or $r^T\mu > \lambda$ (shown as the upper region in the figure above). The assignments used in the schedule 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 $r^T\mu > \lambda$.

(1) is not the concern, but to where the main challenge is Step 2 of Algorithm 2, that is, finding assignments that satisfy (4). This task can be done by solving

$$\max_{\mathbf{v}} \sum_x R(\mathbf{v}, x)\, \mu_x. \tag{5}$$

It is not hard to show that (5) is equivalent to the maximum weighted independent set (MWIS) problem, which, in the worst-case, is NP-hard. However, the MWIS problem is not NP-hard in all cases. For example, in the case of perfect graphs [8], trees, interval graphs, claw-free graphs, fork-free graphs, sparse graphs, and disk graphs, there exists polynomial algorithms. Also, there exist a large number of computationally efficient algorithms for the general case and there are a large number of approximation schemes.

Techniques to find the approximate MWIS can often find the actual MWIS. However, a 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 exist a better assignment (and hence the current schedule is optimal) or there does exist 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 $\mathcal{X}$, the weighted chromatic number. In general, both numbers are NP-hard to compute. However, the Lovász number [8], $\mathcal{V}$ can always be computed in polynomial time and $\omega \le \mathcal{V} \le \mathcal{X}$. In some cases (e.g., in the case of a perfect graph), $\omega = \mathcal{V}$.

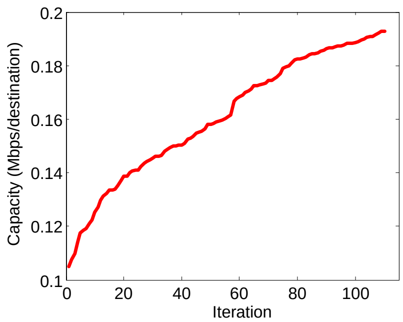

Figure 2. A realistic 500 node mesh network in 2km^2 region of downtown Chicago was constructed with the UDelModels urban mesh networking simulation tools [2]. There were 12 gateways and 500 destinations. The bit-rate for each link was set according to the maximum bit-rate that 802.11a would support over the link. Each destiantion was assigned a nearby gateway and the data flowed from the assigned gateway to the desitination. The capacity was maximized where $U(f) = \min_\phi f_\phi$. The figure on the left shows how the capacity increases as better assignments are found. After 110 iterations, no better assignment could be found (this took around 6 minutes on a 2.8GHz PC). Since an exact method was used to solve the MWIS problem (Step 2 in Algorithm 1), the inability to find a better assignment implies that the found schedule is optimal.

Thus, in such cases it is possible to compute the MWIS in polynomial time. In general, if a method to find an approximate MWIS method fails to find a new assignment that satisfies (4), 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, $\lambda < V$ does not imply that the assignment found by the approximation is not optimal. However, we have found that for the realistic mesh networks examined (i.e., those generated by the UDelModels), 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 memory and we have been unable to confirm that $\omega = \mathcal{V}$ for networks with more than 100 links.

Figure 2 shows an example of how the computed capacity varies as better assignments are added to the set of consider assignments. After 110 iterations, no better assignments exist, hence the schedule is optimal.

4 Optimal Routing

The above presents how optimal schedules can be computed. It is also possible to compute optimal routes. In the optimization scheme above, the Lagrange multipliers were used to find assignments that provide good performance. The same approach can be used to find good routes. Intuitively, the Lagrange multiplier μ_x can be interpreted as the link cost in dollars/bit for link x. Suppose that flow ϕ uses a path $\mathcal{P}(\phi)$. Then the cost along this path is $\sum_{\phi \in \mathcal{P}(\phi)} \mu_x$. If flow ϕ uses multiple paths, then they will all have the same cost. Now, if there exists an alternative path Q such that $\sum_{x \in Q} \mu_x < \sum_{\phi \in \mathcal{P}(\phi)} \mu_x$, then this path Q should be used. That is, using this path will increase the capacity. Alternatively, if flow ϕ is spread over multiple paths, only the ones with minimal cost are used. That is, paths with higher cost can be removed from consideration. In this way, paths can be added and removed from consideration. We have found that this approach converges after a few iterations (typically under 10 iterations for a 90 node network with 6 gateways). Furthermore, as compared to a shortest path routing, optimal routing improves the capacity by 10 to 50%.

References

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

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

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

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

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

[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] 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.

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