Engineering Wireless Mesh Networks: Joint Scheduling, Routing, Power Control, and Rate Adaptation

Jun Luo, *Member, IEEE*, Catherine Rosenberg, and André Girard

***Abstract*—We present a number of significant engineering insights on what makes a good configuration for medium- to large-size wireless mesh networks (WMNs) when the objective function is to maximize the minimum throughput among all flows. For this, we first develop efficient and exact computational tools using column generation with greedy pricing that allow us to compute exact solutions for networks significantly larger than what has been possible so far. We also develop very fast approximations that compute nearly optimal solutions for even larger cases. Finally, we adapt our tools to the case of proportional fairness and show that the engineering insights are very similar.**

***Index Terms*—Column generation, power control, rate adaptation, routing, scheduling, wireless mesh networks (WMNs).**

I. Introduction

WIRELESS mesh networks (WMNs) such as IEEE 802.16 [2] are seen as a promising alternative to other (wired) broadband access technologies. In order to offer high throughput, WMNs will have to be tightly managed. Once an operator has placed his mesh routers and his gateway to offer appropriate coverage to a set of end-users, he will need to engineer his WMN to maximize the network performance. This means choosing among a number of sometimes conflicting options with complex interactions that can affect performance to various degrees. The main objective of this paper is to produce *quantitative* measures of the impact of these choices on the performance of networks of *realistic* sizes.

We examine these issues in the centralized framework developed in [3], where we assume that the position of the nodes, the flows, the interference, and propagation models are known at a central location where the optimal configuration is computed and then passed along to each mesh router. Note that we are *not* claiming that centralized solutions are necessarily the best way to operate WMNs. The point is that this framework provides an upper bound on the performance that can be achieved on WMNs using random access protocols or some form of distributed scheduling [4], [5]. It can also provide joint routing, scheduling, power control, and rate adaptation in scheduled networks whenever a centralized solution is deemed more appropriate. Note that with an additive interference model such as the one we use, finding a set of links that can be scheduled at the same time requires information from potentially widely separated areas in the network. In that context, implementing a decentralized solution would be difficult at best, if not outright impossible, due to the large information transfers that would be needed.

The **first contribution** of this work is to provide deep practical insights on the engineering of WMN networks when the objective function is to maximize the minimum throughput among all flows.

- First, we examine the impact of power and rate selection on the performance of WMNs. We show that while multiple power levels improve the performance of the network, a few power levels is enough as long as they are selected correctly. On networks with multiple rates, we show that an optimal configuration tends to trade spatial reuse for high link rate.
- Another result is linked to the multihop capability of WMNs. Multihop communication enables connectivity at much lower transmit powers than single-hop communication and yields the maximum achievable throughput at significantly lower transmit power at the gateway.
- We study routing in WMNs and show that multipath optimal routing is not much more efficient than single-path optimal routing and that not all min-hop routings are equally efficient. We also quantify how suboptimal is the "best" min-hop routing using realistic scenarios.
- A major advantage of WMNs is spatial reuse, the possibility of using the same channel in different areas of the network. We show that the relationship between spatial reuse and network performance is not straightforward.

These results can be obtained only by solving a hard *mixed integer linear program* (*MILP*). The tool developed in [3] used a commercial solver to calculate a solution after reformulating the problem into a *standard linear program* (*LP*). While it is true that state-of-the-art solvers can handle large LP instances, that approach was still limited in the scope and size of networks that could be solved and was clearly not adequate for the task since the number of variables of the LP formulation grows exponentially with the network size.

Some form of decomposition or column-generation formulation is then needed. While commercial solvers do not provide column generation automatically, they *can* be used to solve the

Manuscript received March 05, 2009; revised October 13, 2009; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Sarkar. Preliminary results were presented in *Proceedings of the 19th IEEE PIMRC*, 2008 [1].

J. Luo was with the University of Waterloo, Waterloo, ON N2L 3G1, Canada. He is now with the School of Computer Engineering, Nanyang Technological University, Singapore (e-mail: junluo@ntu.edu.sg).

C. Rosenberg is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: cath@ece.uwaterloo.ca).

A. Girard is with Groupe d'Études et de Recherche en Analyse des Décisions (GERAD), Montréal, QC H3T 1J4, Canada (e-mail: andre.girard@gerad.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2010.2041788

pricing subproblems, which have a smaller size. Nevertheless, this approach works only for relatively small network instances, and in order to get quantitative results for large networks, we had to develop new computation tools, both exact and approximate, that are efficient enough to study realistic WMN scenarios. These scenarios would have several tens of mesh routers, many flows, several possible modulation/coding schemes, and many possible power levels. The development of these tools is the **second contribution** of this work. To the best of our knowledge, no such tools have been developed since all the results that have been reported for this type of networks have been for at most 20 to 25 nodes [3], [6]. More precisely, in our second contribution, we do the following:

- We propose a column-generation technique which allows us to solve **exactly** medium-size problems. The difficulty is to solve the NP-hard pricing subproblem in an efficient manner. This is especially important since it has to be solved repeatedly. We do that by introducing a technique that we call "greedy pricing," which uses an enumeration-based algorithm on a restricted set of links.
- We show that this technique allows us to compute exact solutions for problems much larger than what an improved version of the original technique proposed in [3] can do. For networks small enough for both techniques to handle, our technique also turns out to be much faster.
- We also propose and compare two approximate algorithms that are fast and very accurate. They can be used to compute solutions for much larger networks.

Our **third contribution** is related to proportional fairness (PF). We adapt our tools to this case, which is very challenging since it yields a nonlinear problem. In our third contribution:

- we show how our technique blends smoothly with a sequential linear programming approach;
- we show some numerical results that illustrate, in the case of one power level and one rate, that the trends are similar to the ones we had seen for the max-min case;
- we also compare the performance of a network configured with a max-min versus a PF objective and show that the gain in social welfare for the PF configuration is not that great.

We provide in Section II some background on models and computational tools developed in related work. Then, we describe our network model and formally define our optimization problem in Section III. In Section IV, we present our algorithm based on column generation to solve it exactly. We compare its computation times to a benchmark based on the simplex algorithm using a smart technique to construct the coefficient matrix. In Section V, we report on the engineering insights that we obtained by using this tool on realistic WMN scenarios. They are based on exact computations. In Section VI, we propose and compare two fast and accurate approximate methods. Section VII addresses the case of a PF objective.

II. Related Work

Jain *et al.* [7] were among the first to formulate a joint routing and scheduling problem for wireless networks valid for all linear objective functions, including max-min. The framework they proposed is rather comprehensive: it includes both the protocol and the physical interference models, an extension of which is used in our paper, and it can accommodate physical technologies such as multiple radios and nonoverlapping channels. One limitation is that power control and rate adaptation are not considered. Jain *et al.* only provide upper bounds obtained by applying clique feasibility conditions and lower bounds obtained by using subsets of all schedulable link sets. The gap between the upper and lower bounds is nonzero unless the conflict structure induces a perfect graph [3]. The computation of these bounds is rather cumbersome, and hence the results presented in [7] are limited to the protocol interference model.

There have been many attempts to extend this optimization framework and to improve the algorithms that solve it. Zhang *et al.* [8] apply column generation to solve a similar problem in multiradio and multichannel networks. However, it is not clear how their algorithm would scale since it is based on an exact pricing. Both [9] and [10] take a staged approach. They first solve a concurrent flow problem using the algorithm of [11] followed by a packing-based heuristic to approximate the optimal link channel assignment. Note that [8]–[10] only consider the protocol (interference) model and nonoverlapping channels. The results of [12] clearly show the importance of choosing an additive interference model. Their approach does not involve power control and rate adaptation. Karnik *et al.* [3] extend the framework of [7] to encompass multipower and multirate and focus only on the additive interference model. They propose an exact enumeration-based algorithm and derive an upper bound on the size of a schedulable link set (see also Section IV-A for details). Another distinct contribution of [3] is a characterization of the optimal max-min throughput by routing and clique feasibility conditions. A similar characterization is applied in [13] to construct a new routing metric called *interference clique transmission time*, though it is not clear how practical such a scheme can be.

There also exists another body of work applying online dynamic control for throughput maximization, e.g., [14]–[16] and the references therein. That approach considers the cases where no information about the environment is available *a priori*. However, the price paid for the lack of *a priori* information is the increased algorithmic complexity: NP-complete subproblems such as the maximum weight independent set problem need to be solved online repeatedly [16]. Moreover, any attempt at approximating the NP-complete subproblems may drastically reduce the performance [17].

On the algorithmic side, the column-generation method has been applied intensively to the cross-layer design of multihop wireless networks. However, these studies either rely on commercial solvers such as CPLEX [18] to deal with the NP-complete/hard subproblem that generates a column [19], [8], [6], [20] or make use of a greedy heuristic to obtain suboptimal solutions [21], [22]. While the first approach does not scale with an increasing problem size, the second one almost always fails to provide optimal solutions [20]. As described later in Section IV-B, our greedy pricing approach delivers both exact and approximate solutions and scales well with the problem size.

III. Network Model and Problem Formulation

We model the network as a set $\mathcal{N}$ of *nodes* (the mesh routers and the gateway) and a set $\mathcal{L}$ of *directed links*, with $|\mathcal{N}| = N$

and $|\mathcal{L}| = L$. Each node $i \in \mathcal{N}$ has a location (x_i, y_i). We denote by $\mathcal{L}_i$ the set of links incident (inbound or outbound) to a node i. A link $l \in \mathcal{L}$ is identified **not only** by its origin–destination pair, **but also** by its physical parameters, which are defined in Section III-A. Let $\mathcal{F}$ denote the set of *flows*, and let $|\mathcal{F}| = F$. A flow $f \in \mathcal{F}$ is identified by its source–destination pair (f_s, f_d) and has a rate λ_f. Let $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_F]$ be the flow rate allocation vector.

In the following subsections, we present models for the physical, interference, and network layers and formulate the joint routing, scheduling, rate adaptation, and power control problem whose solution can be used to configure the network. Note that, in all cases, we restrict ourselves to conflict-free scheduling.

A. Physical Layer Model

Each link $l \in \mathcal{L}$ is identified by four physical parameters:

- $o(l), d(l)$ — the origin and the destination nodes of l. A link is sometimes denoted (i, j) whenever the context is clear, where $i = o(l)$ and $j = d(l)$.
- P_l — the transmit power used by $o(l)$. It takes its value from a finite set $\mathcal{P}$. This represents the *power control* ability of a node. We assume a network-wide reference power level. All nodes use the same reference power and a finite number of power levels that have fixed offsets from the reference power.
- c_l — the link rate in bits per second. It takes its value from a finite set $\mathcal{C}$. This represents the *multirate* capability of a node. We assume that a particular rate can only be obtained from one modulation/coding scheme. Hence, there are $|\mathcal{C}|$ modulation/coding schemes.

Let $\mathbf{P} = [P_1, \ldots, P_L] \in \mathcal{P}^L$ and $\mathbf{c} = [c_1, \ldots, c_L] \in \mathcal{C}^L$ be the power and link rate vectors, respectively. Because of the way we define a link, it is more a *logical* entity rather than a *physical* link since there are potentially multiple links between two nodes i and j that differ from each other by the power used by i and/or the link rate. Strictly speaking, a link should be referred to by a set (o, d, P, c), but we use a single index l for ease of notation. There might be from zero up to $|\mathcal{P}| \times |\mathcal{C}|$ links between i and j. Two links between i and j with the same rate and different transmit power will differ in their robustness against interference as discussed below.

We assume that a link l characterized by $(o(l), d(l), P_l, c_l)$ is feasible if its *signal-to-noise ratio* (SNR) meets the following condition:

$$\mathrm{SNR}_l = \frac{G_l P_l}{N_0} \geq \beta(c_l) \tag{1}$$

where G_l denotes the channel gain on l, N_0 is the average thermal noise power in the operating frequency band, and $\beta(c_l)$ is the threshold related to the modulation/coding scheme that yields c_l. The channel gain between two points separated by distance d is assumed to be given by $F_l(d/d_0)^{-\eta}$, where d_0 is the close-in reference distance, F_l is the shadowing and fading gain, and η is the path loss exponent. The size of the link set $\mathcal{L}$ for a given set of powers $\mathcal{P}$ and a set of rates $\mathcal{C}$ is then given by the number of links such that (1) holds for each quadruple (i, j, P_l, c_l) and hence is a function of $|\mathcal{P}|, |\mathcal{C}|$ and $|\mathcal{N}|$. There is an implicit assumption here that the channel gain is quasi-time-invariant. This is a realistic assumption in urban/suburban areas with rooftop antennas [23].

B. Additive Interference Model

We now present the additive interference model we use in this paper, which extends the *physical interference model* of [24]. It is described using the concept of an *independent set* (ISet)[1], a set of links that can all operate at the same time, i.e., the interference they produce is not harmful to any of the links in the set. We denote by $\mathcal{I}$ the set of all ISets and by $\mathcal{I}_l$ the set of ISets that contain link l.

First, note that a set $s \subseteq \mathcal{L}$ is an ISet only if no two links in the set share a node, i.e.,

$$i \neq i' \wedge i \neq j' \wedge j \neq i' \wedge j \neq j' \quad \forall l, l' \in s. \tag{2}$$

We also assume that the interference on a given link is the cumulative interference from all the links that are active at the same time. Hence, under this interference model, a set $s \subseteq \mathcal{L}$ is an ISet iff it meets condition (2) and the following condition:

$$\gamma_l = \frac{G_l P_l}{N_0 + \sum_{l' \in s: l' \neq l} G_{l'l} P_{l'}} \geq \beta(c_l) \quad \forall\, l \in s. \tag{3}$$

Here, γ_l is the *signal-to-interference-plus-noise ratio* (SINR) of link l, and $G_{l'l}$ is the channel gain from $o(l')$ to $d(l)$. Recall that a link l is in fact a logical link represented by a tuple (o, d, P_l, c_l). Consider an ISet s containing some link l. Assume also that l' is another feasible logical link between o and d, i.e., $P_l \neq P_{l'}$ and/or $c_l \neq c_{l'}$. It should be clear that the set $s' = \{s \backslash l\} \bigcup \{l'\}$ is not necessarily an ISet, either because it produces too much interference at some other receiving nodes of s or is receiving too much interference from the transmitting nodes of the other links in s. These conditions are automatically checked by the construction algorithm described in Section IV-A.

C. Network Model

The network model proposed here is based on the assumption that the traffic is static or quasi-static. We believe that it is reasonable since the traffic seen by the mesh routers is aggregated.

We will consider both multipath and single-path routing. For multipath routing, we denote by $\mathcal{R}_f$ the set of all routes that can be used by flow f and by $\mathcal{R}_f^l$ the set of all routes that can be used by f going through link l. The amount of flow f routed on $r \in \mathcal{R}_f$ is denoted by ϕ_f^r and $\sum_{r \in \mathcal{R}_f} \phi_f^r = \lambda_f$. Let $\boldsymbol{\phi} = [\phi_f^r]_{r \in \mathcal{R}_f, f \in \mathcal{F}}$ be the *routing vector*.

A *link schedule* is an $|\mathcal{I}|$-dimensional vector $\boldsymbol{\alpha} = [\alpha_s]_{s \in \mathcal{I}}$ such that $\alpha_s > 0$ if ISet $s \in \mathcal{I}$ is scheduled, and $\alpha_s = 0$ otherwise. We interpret α_s as the fraction of time allocated to an ISet s. Obviously, $\sum_{s \in \mathcal{I}} \alpha_s \leq 1$. We only schedule ISets since we are interested in conflict-free scheduling.

In summary, we want to compute the flow rate allocation vector $\boldsymbol{\lambda}$, the routing vector $\boldsymbol{\phi}$, and the scheduling vector $\boldsymbol{\alpha}$ using the following optimization framework.

D. Problem Formulation

The following joint routing, scheduling, rate adaptation, and power control problem (JP) is based on the parameters and

[1]The term "independent set" is not the same as the notion of independent sets as used in graph theory. However, we use it in order to be consistent with the literature.

variables defined above. The multipath formulation given in Section III-D1 comes from [3], while the single-path formulation presented in Section III-D2 is new.

1) Multipath Formulation: We define the link-set incidence matrix Q such that $q_{l,s} = 1$ if $l \in s \in \mathcal{I}$, and 0 otherwise. Note that each column $\mathbf{q}_s$ of Q is a vector that represents an ISet s and that the number of columns is $|\mathcal{I}|$, which is generally very large. For a flow $f \in \mathcal{F}$, we also define the standard node-arc incidence matrix A^f such that $a^f_{i,l} = +1$ if $i = o(l), -1$ if $i = d(l)$, and 0 otherwise. The dependence on f is useful to prevent certain flows from using some links. Define also the node-flow incidence vector $\mathbf{d}^f$, where $d^f_i = 1$ if $i = f_s$, and 0 otherwise. Let $x^f_l = \sum_{r \in \mathcal{R}^l_f} \phi^r_f$ be the amount of flow f going over a link l, and denote by $\mathbf{x}^f = [x^f_1, \ldots, x^f_L]$ the link flow vector associated with f. Finally, let $\mathbf{x} = [\mathbf{x}^f]_{f \in \mathcal{F}}$.

Given the network model and the definitions, we want to maximize the minimum throughput of all the flows, i.e., $\max \min_f\{\lambda_f\}$. In this form, the objective function is not differentiable and the problem is transformed by the standard technique of introducing a scalar variable $\lambda = \min_f\{\lambda_f\}$ and adding a set of constraints (5) to put a bound on the flows. We can then formulate JP as (4)–(7).

$$\max_{\lambda, \mathbf{x}, \boldsymbol{\alpha}} \quad \lambda \tag{4}$$

$$\left(\mu^f_i\right) \qquad A^f \mathbf{x}^f \geq \lambda \mathbf{d}^f \qquad \forall f \in \mathcal{F} \tag{5}$$

$$(\nu_l) \quad c_l \sum_{s \in \mathcal{I}} q_{l,s} \alpha_s \geq \sum_{f \in \mathcal{F}} x^f_l \qquad \forall l \in \mathcal{L} \tag{6}$$

$$(\zeta) \qquad \sum_{s \in \mathcal{I}} \alpha_s \leq 1$$

$$\mathbf{x}, \boldsymbol{\alpha} \geq \mathbf{0} \tag{7}$$

where we have put the Lagrangian multipliers corresponding to each constraint in parenthesis. In this formulation, we have $\mu^f_i \geq 0, \nu_l \geq 0$ and $\zeta \leq 0$. The maximization is explicitly taken with respect to the maximum flow λ, link load allocation $\mathbf{x}$, and link scheduling vector $\boldsymbol{\alpha}$, but it is also implicitly taken over the transmit power vector $\mathbf{P}$ and link rate vector $\mathbf{c}$ since they are implied by the scheduling of ISets and the links that make up these sets. We call this problem JP-Primal. It is a standard but very large linear program (LP) and its difficulty lies in the computation of the incidence matrix Q which grows exponentially with the problem size.

Note that the solution of this problem does not yield full information about the optimal configuration: The routing vector $\boldsymbol{\phi}$ is replaced by its aggregated form $\mathbf{x}$. Nevertheless, there are standard procedures to reconstruct a set of compatible path flows from the arc flow formulation. Obviously, even though we have chosen a max-min objective function, our tools could be directly adapted for any other linear objective function.

2) Single-Path Formulation: To be able to take into account single-path routing where a flow is constrained to use only one path, we add a binary variable y^f_l such that $y^f_l = 1$ if link l is used to carry flow f, and $y^f_l = 0$ otherwise. We also add the following constraints to the JP problem (4)–(7):

$$x^f_l \leq c_l y^f_l \quad \forall l \in \mathcal{L}, f \in \mathcal{F} \tag{8}$$

$$\sum_j y^f_{(i,j)} \leq 1 \quad \forall i \in \mathcal{N}, f \in \mathcal{F} \tag{9}$$

where (i, j) denotes the links out of node i. Constraint (8) states that if l is not used to carry f, i.e., if $y^f_l = 0$, the load x^f_l imposed by f on l is zero. Equation (9) states that, for a given node i, at most one outgoing link is used to carry a certain flow f. A similar formulation was used in [7]. Note that this makes the problem much harder to solve since we now have an integer problem due to the presence of the binary variables $\mathbf{y}$.

IV. Tools for Exact Solutions

In order to get more engineering insights than those obtained in [3], we are faced with the task of computing solutions for relatively large networks with many mesh routers, flows, levels of powers, and possible rates. This means solving a very large LP with a coefficient matrix that grows exponentially with the size of the network.

We now present two efficient algorithms that solve JP exactly. The first one is a direct application of the simplex algorithm, where we construct the Q matrix using an efficient enumeration of the ISets. It will serve as a benchmark to measure the gain in computation time of our second proposed algorithm based on column generation. We will show in Section V that the range of networks that can be solved exactly by the algorithm based on column generation is quite extensive and includes many realistic scenarios. We will propose and compare in Section VI fast and accurate approximate algorithms for even larger networks.

A. Solution by the Simplex Algorithm

The solution technique proposed in [3] was a straightforward use of the simplex algorithm. While this requires a complete enumeration of all the ISets, there is a definite advantage to this approach. Once we have built the set $\mathcal{I}$, we can easily solve different problem instances as long as they all have the same ISets, e.g., different objectives, flow patterns, different gateway positions, etc. This might not be possible with the column-generation technique presented later, which requires us to start anew for each change in the input set. It is thus worth the effort of trying to design an efficient exact method based on the simplex algorithm even if we know that other exact methods can be more efficient in other situations (see Section IV-B).

The difficulty with this approach is that there is a huge number of ISets that have to be constructed beforehand. A naive construction procedure is to enumerate all the $2^{|\mathcal{L}|}$ elements of the power set of $\mathcal{L}$ and to check whether each of them forms an ISet. Karnik *et al.* [3] improved this brute force algorithm by deriving an upper bound $B \ll L$ on the maximum size of an ISet, such that only $\mathcal{O}(|\mathcal{L}|^B)$ elements of $\mathcal{I}$ need to be enumerated. However, the bound is still too loose to allow efficient enumeration.

We propose here an efficient algorithm that constructs all possible ISets but no more. The complexity of this method is only $\mathcal{O}(|\mathcal{L}|^M)$ as opposed to $\mathcal{O}(|\mathcal{L}|^B)$ in [3], where M is the maximum ISet size and typically $M \ll B$. We describe it using a recursive depth-first algorithm, but we have also implemented an iterative breadth-first version. While the recursive form is simpler to program and is well suited for enumerating all ISets, the iterative form is better suited for enumerating only the *maximal* ISets. The algorithm is based on the following proposition that is trivial to prove.

Proposition 1: If $s \in \mathcal{I}$ is an independent set, then any subset of s is also an independent set.

Algorithm *Prune The Candidate List*
1. define `prune`(s, Λ)
2. $\Lambda' \leftarrow \emptyset$
3. $\bar{l} \leftarrow \max_{l \in s} l$
4. **for** $l \in \Lambda$
5. **if** $l > \bar{l}$ and $\{l\} \cup s$ is an ISet
6. $\Lambda' \leftarrow \Lambda' \cup \{l\}$
7. **return** Λ'

Fig. 1. Pruning function.

The algorithm builds ISets of increasing sizes and stops when this is no longer possible. This is done using an enumeration tree as follows. The root node is at depth 0. A node at depth k contains an ISet s of k links and a list of links Λ that are candidate for addition to this ISet. We assume that $\mathcal{L}$ is implemented as an ordered data structure indexed by increasing link number. Consequently, $l' > l$ means that l' appears later than l in $\mathcal{L}$ and $\max_{l \in s} l$ returns the link whose index number is the largest in s.

We define two functions. The first one is `prune` (s, Λ), which returns a reduced candidate list of links Λ' constructed as described in Fig. 1. The condition on line 5 for adding a link l to an ISet has two parts. The first is used to avoid enumerating ISets more than once, and the second tests the set $s \cup \{l\}$ against the appropriate interference model defined in Section III-B. The first condition is due to the fact that the ISets are built in a precise order. At a given depth k in the tree, the tree nodes contain ISets of k links and are built left to right with the link numbers in an increasing lexicographic order. Suppose that $s = \{\ldots \bar{l}\}$ and that $l < \bar{l}$. Since there is already a node to the left of the current node containing the ISet $s' = \{\ldots l \ldots\}$, there is no need to add l to s.

The other function `makeSons` $(s, \Lambda, \mathcal{I})$ is a simple recursive procedure that builds all the children of a node s for a given candidate list Λ. It should be clear that a child of s is an ISet that differs from s by one link. The set $\mathcal{I}$ of ISets is accumulated at each node until the whole collection is built up. Finally, the algorithm is initialized with the root node empty, and the first candidate list is the set of all links.

B. Solution by Column Generation

Even with the most efficient enumeration technique, the approach of Section IV-A will eventually become infeasible due to the large size of set $\mathcal{I}$. On the other hand, we know that only a few ISets will be active in the optimal solution by the following proposition.

Proposition 2: The number of nonzero elements of $\boldsymbol{\alpha}$, i.e., the number of ISets that are effectively used, in a basic solution of JP-Primal is at most $L + 1$.

This follows directly from Carathéodory's theorem. This suggests that we use column generation [25] to solve the problem, thus avoiding the explicit generations of $\mathcal{I}$.

If we write the constraints (4)–(7) in standard matrix form, the column corresponding to constraints (6) and some variable α_s has the form $\tilde{\mathbf{q}}_s = [c_1 q_{1,s}, c_2 q_{2,s}, \ldots c_L q_{L,s}]^T$. These are used for pricing the ISets as follows.

1) Exact Pricing: Column generation is basically the revised simplex algorithm with a particular pricing technique. The technique uses only a subset of columns $\tilde{\mathbf{q}}_s$, which is called the *Restricted Master Problem* (RMP). At a given iteration, we have, for the RMP, a basic feasible solution $\mathbf{x}, \boldsymbol{\alpha}$, and λ as well as the current estimate of the dual variables $\boldsymbol{\mu}, \boldsymbol{\nu}$, and ζ.

The first step of the next simplex iteration is to price the off-basis columns. The reduced cost r_s of an off-basis column $\mathbf{q}_s$ is given by the standard formula

$$r_s = 0 + \zeta + \sum_{l \in \mathcal{L}} \nu_l c_l q_{l,s} \tag{10}$$

since the cost coefficient of the off-basis variable α_s in the objective function is zero. When the objective is to maximize, the standard pivoting rule of the simplex is to choose the column with the largest reduced cost. The stopping rule is also simple: If there is no off-basis column with a strictly positive reduced cost, the current solution is optimal. This means that the pricing requires the solution of the following *Maximum Weight ISet* (MWIS) problem with $c_l \nu_l$ as the link weights

$$\max_{s \in \mathcal{I}} \sum_{l \in \mathcal{L}} c_l \nu_l q_{l,s} \tag{11}$$

subject to constraints (2) and (3). It can be easily shown that it is an NP-hard problem.

If we want an optimal solution, we must make sure that there is no off-basis column with a strictly positive reduced cost at the last simplex iteration. This in turn means that we have to solve the MWIS pricing subproblem to optimality. It will then become very difficult to compute large networks by a straightforward column-generation technique. This is why we propose another method called *greedy pricing*, which has been proven to be very fast at delivering exact solutions.

2) Greedy Pricing: We can reduce the amount of computation if we use a greedy pricing rule at each iteration. This is based on the fact that choosing *any* column with a positive reduced cost may potentially produce a new solution with a higher value of the objective. We can also speed up the calculation by solving the pricing subproblem over a set of links $\tilde{\mathcal{L}} \subset \mathcal{L}$ with positive reduced costs only. The reason is that if there exists a solution to the MWIS problem where some links with zero reduced costs appear in the solution, then there is an optimal solution made up of only the links with strictly positive reduced costs. This follows from *Proposition 1*.

The algorithm to find a new column for the RMP with a positive reduced cost uses two functions, denoted as `greedy` and `enuoracle`. They take a link set $\tilde{\mathcal{L}} \subseteq \mathcal{L}$ as the input and return an ISet with a positive reduced cost, if such a set exists, or an empty set.

In a basic implementation of `greedy`, the algorithm simply orders the links in decreasing weights. The link with the largest weight is first chosen, then the link with the largest possible weight that is still independent of the first one is selected and so on until an ISet is constructed. Instead of naively using the $c_l \nu_l$ of (11) as the weights, we use a more sophisticated definition taking into account also some measure of the interference; it usually works better than just the $c_l \nu_l$. We refer to the Appendix for more details on this.

If the ISet produced by `greedy` has a positive reduced cost, it is pivoted into the basis of the RMP and the next iteration is started. If the reduced cost is not positive, we try to find a better ISet that might have a positive reduced cost using `enuoracle`. It is similar to the efficient enumeration presented in Section IV-A, but it stops when either finding an ISet with a positive reduced cost or failing to find such a set upon enumerating all the possible ISets that can be constructed from the

TABLE I
LINK RATES AND CORRESPONDING THRESHOLDS

Modulation	coding rate	c_l	$\beta(c_l)$ (dB)
BPSK	1/2	1	6.4
QPSK	1/2	2	9.4
	3/4	3	11.2
16-QAM	1/2	4	16.4
	3/4	6	18.2

current link set $\tilde{\mathcal{L}}$. If we want to guarantee the optimality of the final solution of the column-generation algorithm, a full enumeration of ISets is needed to solve the MWIS exactly, but this is required only in the last iteration and on a smaller set of links.

In practice, our greedy pricing approach runs very fast mainly due to the following three reasons:

- The number of iterations to terminate the algorithm is often far less than $|\mathcal{I}|$.
- The algorithm `greedy` does find an off-basis column with a positive reduced cost in most iterations.
- The size of the link set $\tilde{\mathcal{L}}$ is usually much smaller than that of $\mathcal{L}$ thanks to the dual degeneracy that happens frequently.

As we will now see, our algorithm based on column generation using greedy pricing is much faster than the simplex algorithm with enumeration presented in Section IV-A.

C. Settings for Numerical Results

We now report the computation time of the LP benchmark and the column-generation algorithm. Both are programmed in C++. We use GLPK [26] as the LP solver. All the computations have been done on a machine with a 3.2-GHz Pentium 4 CPU and 1 GB of RAM.

Because of space limitation, we choose to present results only for a subset of the many network scenarios we have studied: one with 25 nodes (Grid25), two with 30 nodes (Rand30a and Rand30b), one with 40 nodes (Rand40), two with 50 nodes (Rand50a and Rand50b), and one with 80 nodes (Rand80). Apart from Grid25, whose nodes are on a square grid and the gateway is in the middle, all the others have their nodes placed at random in a square with the gateway in the center. Note that we scale the network dimension in proportion to the number of nodes so that the node density is always the same. For each network, we require every node to send or receive a flow to or from the gateway. We call these flow patterns *converging* and *diverging*. Since we have not seen any significant differences between results obtained for converging and diverging flow patterns, we will mostly focus on converging flows.

For radio propagation, we assume $N_0 = -100$ dBm, $d_0 = 0.1$ m, $F_l = 1, \forall l \in \mathcal{L}$, and $\eta = 3$. The five normalized link rates c_l and their corresponding thresholds are listed in Table I. They are taken from the IEEE 802.16 standard. For power control, the finite power set $\mathcal{P}$ is represented by a *base power* $P = \max(\mathcal{P})$ and a *step size* p. Therefore, $\mathcal{P} = \{P-(k-1)p, \ldots, P-2p, P-p, P\}$ if there are k power levels. All our results are shown as a function of the base power P. The curves all start at $P_{\min}$, where a network first becomes connected, and stop at P_{SH} that allows every node to have a single-hop connection with the gateway.

D. The Computation Costs of Obtaining Exact Solutions

We show in Fig. 2 the computation times needed for producing the throughput curves as a function of P for Grid25 and Rand30a with a single power and a single rate using the enumeration and the column-generation algorithms.

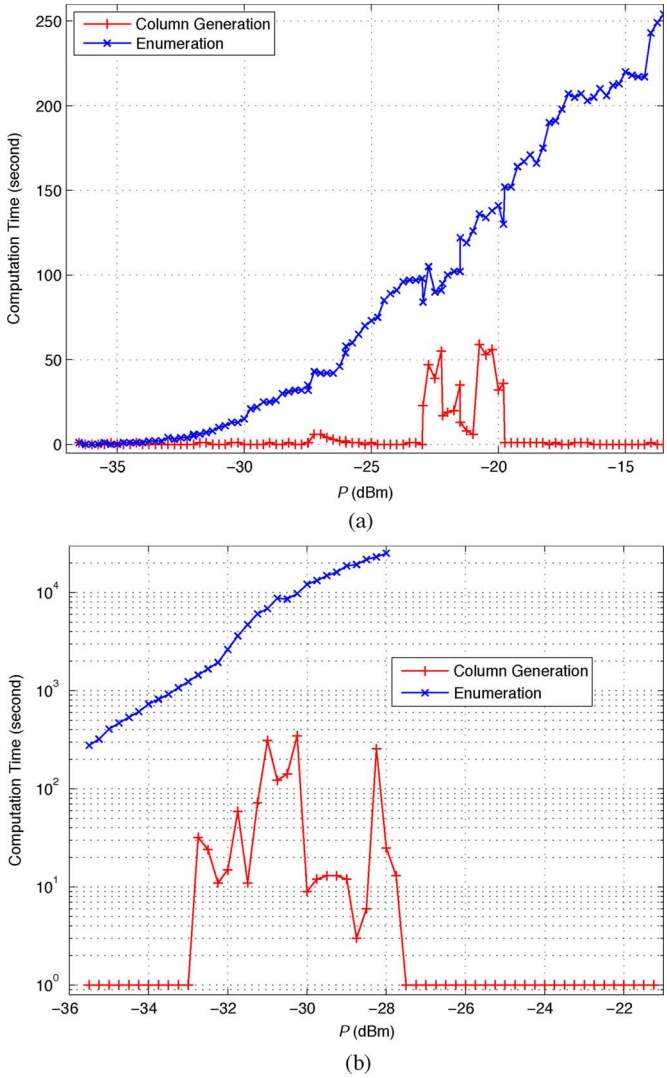

Fig. 2. Computation times for the two exact algorithms for the case with converging traffic: (a) Grid 25 and (b) Rand30a.

As expected, the computation times increase very fast using the enumeration approach. This directly follows from the exponential increase of the number of ISets as the transmit power grows. The times for the column-generation algorithm increase much more slowly when power grows. Actually, the time depends heavily on the fraction of "useful" ISets out of all ISets. At an optimal solution, the algorithm produces a basis with a scheduling vector $\boldsymbol{\alpha}$ whose $\alpha_s \geq 0$. We find that these solutions are highly degenerate and that there are many other optimal vectors with the same value of the objective function. An ISet is said to be *useful* if it belongs to this set of potentially optimal solutions. If the optimal solution were unique with respect to $\boldsymbol{\alpha}$, the fraction of useful ISets would be very low, according to Proposition 2, which fortunately turns out not to be the case. It can be rather high, though still far lower than that of the enumeration algorithm. Note that we were unable to compute an exact solution in a reasonable time using the simplex algorithm for the random network with 30 nodes for power larger than -28 dBm,

whereas we had no problems doing it with our column-generation algorithm.

Under the protocol interference model, both our algorithms compute an optimal solution in the order of seconds and actually less than 1 s in most cases; we omit the numerical results due to space limitations. This is much faster than the algorithm described in [7], where the CPU time is of the order of minutes. Recall that no numerical results were presented in [7] for an additive interference model that incurs much longer computation times.

V. Engineering Insights

We now report numerical results obtained with our computation tools with the engineering insights that they provide.

A. Summary of Previous Results

The main engineering insights that were reported in [3] and [12] can be summarized as follows:

1) Assuming that each node in the network uses the same transmitting power P, the max-min throughput is a non-decreasing function of P.
2) The largest achievable max-min (per-flow) throughput in a N node network is c^*/N if the flow pattern is diverging or converging where c^* is the largest value in $\mathcal{C}$. This throughput can be achieved through single-hop if the transmit power is larger than or equal to $P_{\text{SH}} = \beta(c^*)N_0(D/d_0)^\eta$, where D is the largest distance between the gateway and a node. Note that the throughput is limited by the fact that the gateway cannot receive or transmit more than one packet at a time.
3) There are usually many optimal configurations. They are generally so complex that no simple rule can be deduced from them.
4) In a scenario with a fixed power P_0, if only a single rate, among a given set $\mathcal{C} = \{c_1 < \ldots < c_{\max}\}$, can be used for all nodes, it should be the highest rate allowing connectivity at P_0.
5) The choice of the interference model has a great impact on the solution and the additive interference model yield max-min rates and configuration parameters very different from those obtained with simpler models.

B. Power Control

We first investigate the effect of power control on the optimal throughput. While there is a common belief that many power levels, or even a continuous tuning of the power, are preferable, our results show that, at least for the converging and diverging traffic patterns, only a small number of power levels are needed if they are chosen carefully. All the nodes may use a finite number of power levels that have fixed offsets from the reference power P.

Fig. 3 shows the max-min throughput as a function of P for Rand30a for different $\mathcal{P}$ when the traffic pattern is converging and $c_l = 1$, i.e., there is only one modulation/coding scheme. In Fig. 3(a), we assume two power levels and we show curves for different step size p. It can be easily seen that having access to two power levels yields better performance results than if we have access to only one power level and that the value of the step size has a significant impact on the performance gain. In the case illustrated in the figure, the step size $p = 7$ dB offers

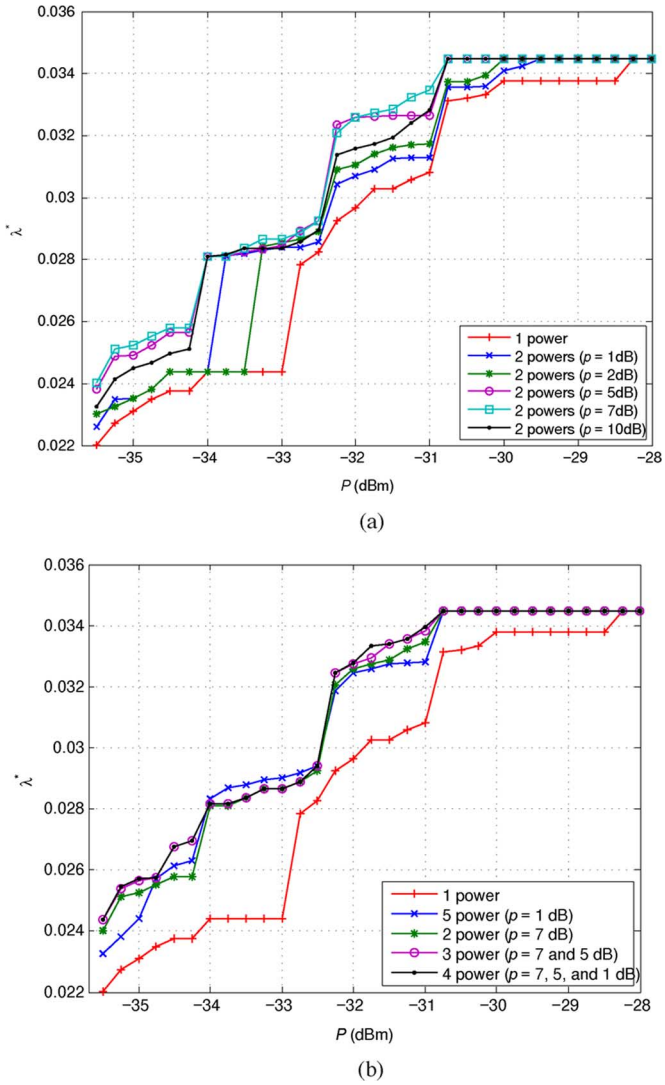

Fig. 3. The impact of power step size and number of power levels on max-min throughput for Rand30a with converging traffic: (a) different step sizes for two power levels and (b) multiple power levels.

an overall performance that is always better than the others. The improvement in throughput when $p = 7$ dB is around 20% as compared to a single power level throughout the whole power range, whereas other step sizes may result in marginal improvement. The results indicate the existence of an optimal power step size, and we can determine this step size relatively quickly with our tools.

In Fig. 3(b), we compare the case with five power levels and a 1-dB step size with the best two power levels. The fact that they are quite close to each other suggests that the number of power levels has a less significant impact on the throughput than the step size. We also show on that figure other curves corresponding to three and four power levels with good step sizes obtained by trial and errors. Again, we observed that adding power levels does not significantly increase the throughput. In other words, we still observe that the higher the reference power, the better the max-min throughput. Having multiple power offsets does improve throughput, but one or two offsets are enough to bring most of the performance gain.

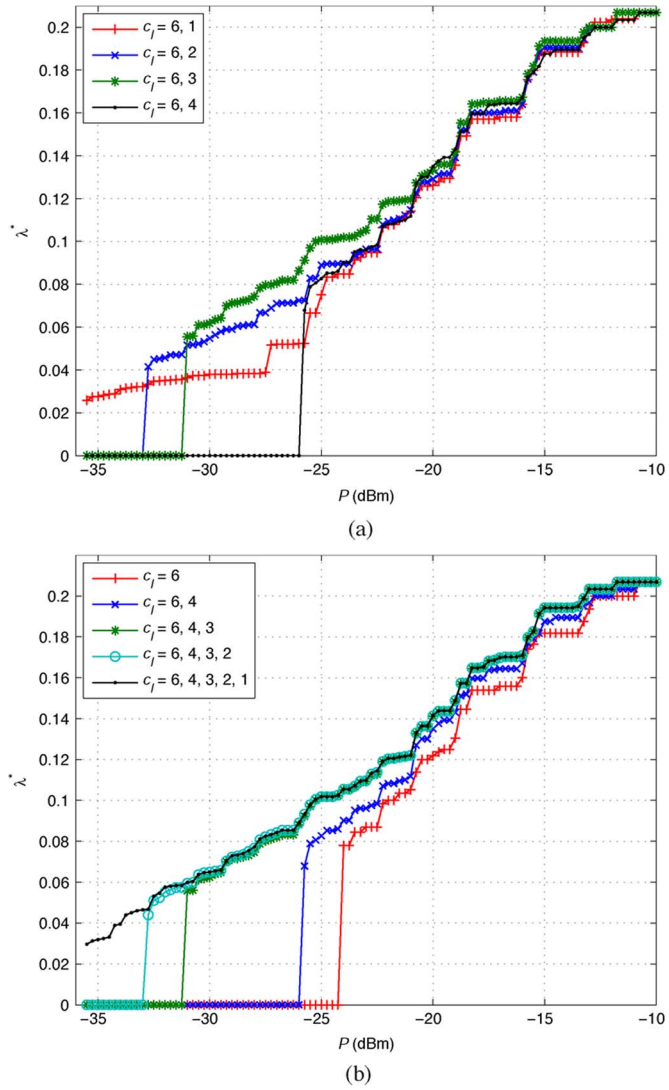

Fig. 4. Multirate versus single-rate for Rand30a in the case of converging traffic: (a) different two-rate combinations, and (b) multirate versus single-rate.

C. Rate Adaptation

We now study the benefit of rate adaptation assuming one power level. It is known that, compared to a single high rate, allowing multiple lower rates enables a network to be connected at lower transmit powers. Our results, in addition to confirming that, demonstrate an interesting trend: The relative gain obtained at high power by having multiple rates over one single high rate is low.

We compare in Fig. 4 the optimal max-min throughput of Rand30a with and without rate adaptation as a function of P in the case of a single power with a converging traffic pattern. Fig. 4(a) compares the max-min throughput as a function of P for the cases where two rates are available, either (6, 4), (6, 3), (6, 2), or (6, 1). We see that if the power is above -31 dB, the best pair of rates is almost always (6, 3). Similarly to the power adaption discussed in the previous section, the results indicate the existence of a "better" pair of rates, and we can determine this pair relatively quickly with our tools.

Fig. 4(b) compares the max-min throughput as a function of P when there is rate adaptation—e.g., two, three, four, and five possible rates—with the case where only the single highest rate is available. As expected, rate adaptation enables connectivity at lower powers than a single high rate. Rate adaptation with three rates yield almost the same max-min throughput than those with four and five rates after the network becomes connected. This seems to indicate that at an optimal configuration, only links with relatively high rates are used. Since operating links with high rates tends to reduce the spatial reuse because they are more susceptible to interference, our results indicate that an optimal configuration tends to trade spatial reuse for high link rate. This will be confirmed later when we study the effect of spatial reuse in Section V-F.

TABLE II
MULTIHOP ADVANTAGE: $P_{\rm SH}$, $\bar{P}$ AND $P_{\rm SH}/\bar{P}$ DIVERGING TRAFFIC

Network	$P_{\rm SH}$ (dBm)	$\bar{P}$ (dBm)	$P_{\rm SH}/\bar{P}$ (dB)
Grid25	-13.75	-18.74	5.00
Rand30a	-22.50	-29.00	6.50
Rand30b	-23.25	-28.75	5.50
Rand50a	-18.25	-25.50	7.25
Rand50b	-19.00	-27.25	8.25

D. The Multihop Advantage

One obvious advantage of multihop communication is that connectivity can be enabled at much lower transmit powers than with single-hop. This section will show and quantify another advantage of multihop over single-hop: In a mesh network with **diverging** flows, the maximum achievable throughput can be obtained by using a much lower transmit power at the gateway using multihop communication than with single-hop communication. In an N-node network with N diverging flows and a single power and a single rate, it can be shown that the maximum achievable max-min throughput is c_l/N if there is only one power and one rate c_l [3]. At $P_{\rm SH}$, i.e., at the transmit power that allows every node to be connected to the gateway through a single hop, we can easily achieve this throughput by scheduling the links between the nodes and the gateway one at a time. Let $\bar{P}$ be the minimum transmit power for which the maximum achievable throughput can be obtained via multihopping. We characterize what we call the *multihop advantage* by $P_{\rm SH}/\bar{P}$ (in dB). Table II shows the multihop advantage for different networks, assuming $c_l = 1$, i.e., $\beta(c_l) = 6.4$ dB. We see that multihop communication achieves the maximum achievable max-min throughput with a transmit power at the gateway from 3.5 up to 6.7 times lower than the power needed for single-hop communication. This is made possible by allowing spatial reuse, i.e., the activation of more than one link at a time. Of course, the energy saving at the gateway is obtained by spending more energy at the intermediate nodes.

Using multihop improves performance both by providing connectivity at low power, something that we cannot do with single-hop, and by offering the maximum achievable throughput at much lower transmit power at the gateway. These two advantages come at the cost of a more complex network operation and more energy spending at intermediate nodes. In addition, our results show that, for a particular network, the multihop advantage as defined above is rather sensitive to the value of β. This is not surprising since the larger the β, the lower the potential for spatial reuse. Similar results have been obtained for multiple power levels and rates.

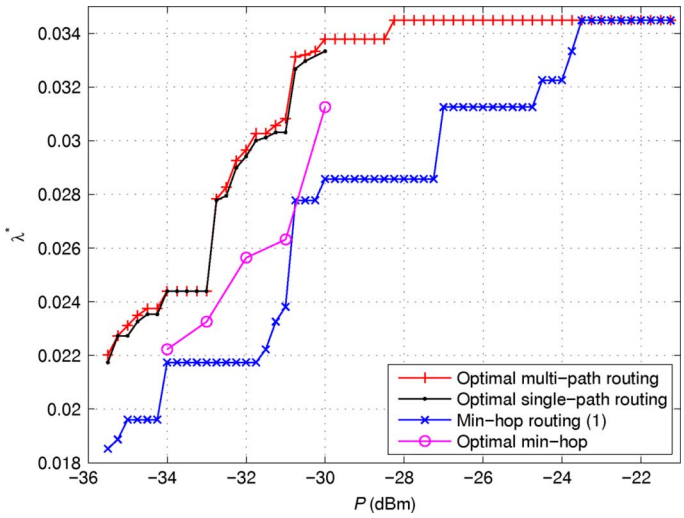

Fig. 5. Optimal multipath and single-path routings versus min-hop routings: Rand30a with converging traffic.

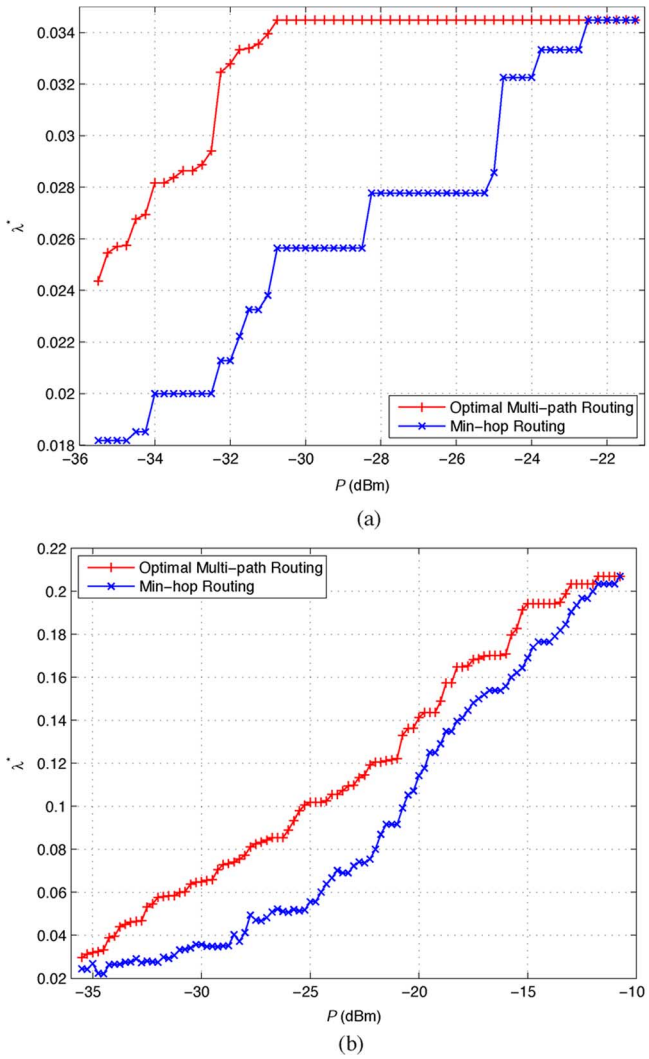

Fig. 6. Optimal multipath routing versus min-hop routing: Rand30a with converging traffic: (a) four power levels $(p, p+7, p+7+5, p+7+5+1)$ dBm and one rate; (b) five rates and one power level.

E. Does Multipath Have an Advantage, and What About Min-Hop Routing?

We now want to address the following two questions:

1) How much do we gain in throughput by allowing each flow to be routed on as many routes as necessary?
2) Can a min-hop routing achieve good performance?

1) Single-Power and Single-Rate: To answer the first question, we solved the single-path version of our JP problem defined in Section III-D2. This is an integer program that cannot be handled by our column-generation approach, which works only for noninteger LPs. Instead, we use the enumeration technique of Section IV-A to produce all the ISets, and then use CPLEX [18] to solve the resulting integer program. Fig. 5 shows the max-min throughput obtained for Rand30a for both the single-path and the original multipath problems. Clearly, multipath does not produce much of an increase in throughput since the single-path max-min throughput is never more than 2% below the multipath value. This is true for all the scenarios that we have studied with converging and diverging flow patterns. Note, however, that this result may be due to the particular max-min objective that we are using. Whether it remains so for other objectives such as proportional fairness or total throughput is still an open question.

In order to answer the second question, we compare our optimal joint routing and scheduling with a min-hop routing on top of an optimal scheduling, i.e., we solve a pure scheduling problem by fixing the routing in JP. Many networks use min-hop routing because it is simple to compute and implement. This can be done either without any consideration for lower layers—for example, by using a simple Dijkstra's algorithm—or by using some information about the lower layers to find a "good" min-hop solution among the many available. We want to compare these two options with our optimal solution.

We first compute the max-min throughput for a min-hop routing obtained by using Dijkstra's algorithm, This min-hop is represented by the curve labeled "min-hop" in Fig. 5. We have also formulated a "cross-layer" min-hop problem in which we compute for each P the best possible min-hop path for each flow. This problem is computationally hard to solve, and this is why the corresponding curve in Fig. 5 is limited to a small section of the power range. One can see that a simple min-hop routing can be very inefficient, while the "best" min-hop routing, i.e., the cross-layer optimized one, is much better but still somewhat far from the optimal.

2) Multipower or Multirate: Min-hop routing has to be carefully defined for multipower or multirate scenarios. The way we obtain a min-hop path for a given flow is as follows. A min-hop path is first produced using Dijskstra's algorithm on all the physical links. We consider that there is a physical link between two nodes if there exists at least one logical link between them. Then, for each physical link that belongs to the min-hop path, we select the logical link with the lowest power (for multipower) or the highest rate (for multirate) to form the actual routing path. Fig. 6 compares our cross-layer design with this min-hop routing. Clearly, a simple min-hop routing can yield significantly lower throughput than our cross-design approach especially in the case of a single rate.

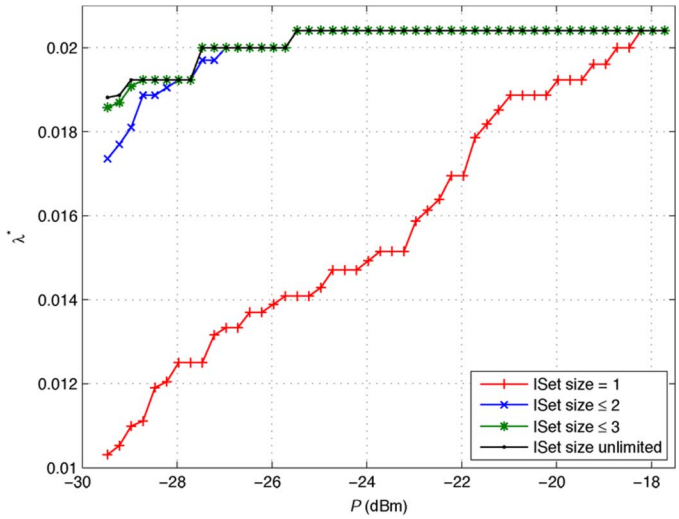

Fig. 7. Optimal max-min throughput under constraints on the maximum size of ISets: Rand50a with diverging traffic.

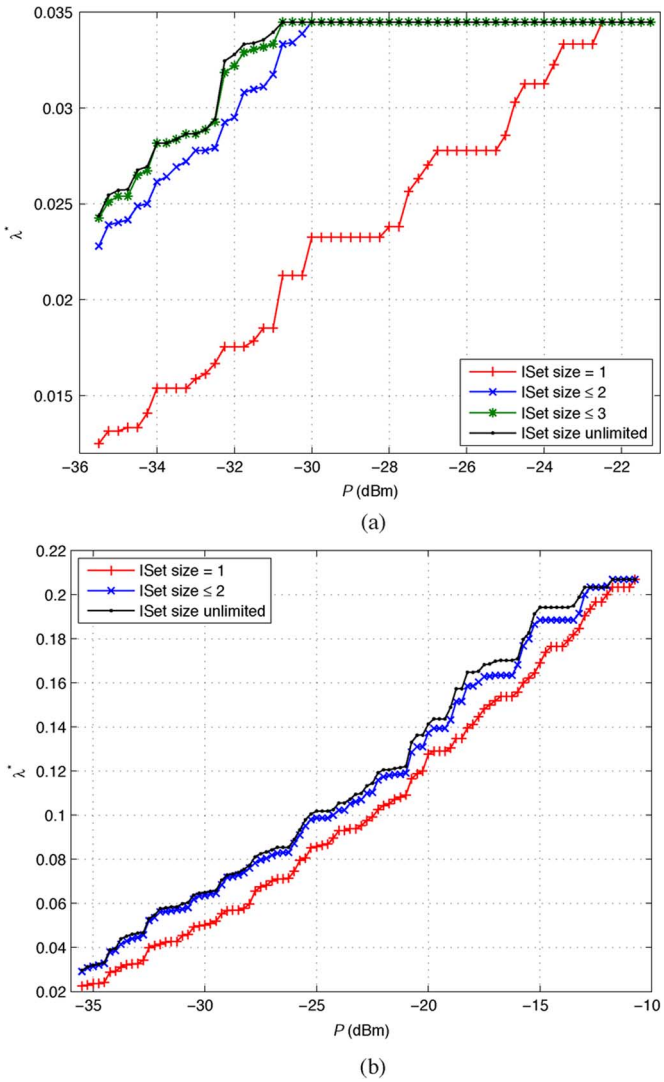

Fig. 8. Optimal max-min throughput under constraints on the maximum size of the ISets. Rand30a with converging traffic: (a) four powers and (b) five rates.

F. Revisiting Spatial Reuse

As we explained in Section V-D, the advantage of multihopping stems from spatial reuse. Therefore, it is natural to think that in an optimal configuration, spatial reuse is high, i.e., ISets of large size are scheduled for a significant fraction of the time. In this section, we want to verify if this conjecture is true. Surprisingly, our results suggest that, for the large number of randomly generated networks with diverging or converging traffic patterns that we have studied, this conjecture is **not** really true: Using only ISets of size ≤ 3, and sometimes only ISets of size ≤ 2, yields a throughput that is almost optimal. This is in spite of the fact that in the networks that we have studied there exist many ISets of size larger than 6. This indicates that spatial reuse is not a good metric to represent the efficiency of a distributed algorithm.

1) Single-Power and Single-Rate: In Fig. 7, we show the optimal max-min throughput curve as a function of P without any restrictions on the size of the ISets and the throughput curves obtained by restricting the size of the ISets that can be use to be less or equal to 1, 2, 3, and 4, respectively for the 50-node network with a diverging flow pattern (we find the same kind of results for the converging pattern).

Comparing the case ISet size ≤ 2, where at most two links can be scheduled at the same time, with the case ISet size $= 1$, corresponding to no spatial reuse, it is obvious that there is a big advantage to allow some level of spatial reuse. However, the gain obtained by allowing more spatial reuse is decreasing fast, e.g., the throughput obtained with ISet size ≤ 3 is not much higher than the one obtained with ISet size ≤ 2. Moreover, ISet size ≤ 4 yields a throughput that is almost the same as that with ISet size ≤ 3 or the optimal one. This is rather surprising since it seems to indicate that even moderate spatial reuse is enough to reach high throughput. We believe that the reason is that our traffic pattern is very much gateway-centric, and hence, as discussed in [3], the performance of the network can only be improved by trying to schedule one link to or from the gateway as much as possible. This result is very important since computing the throughput by limiting the ISet size to 2 or 3 makes the computation and possibly the network operation much simpler. Finally, we note that the multihop advantage (see Section V-D) is obtained with ISet size ≤ 3 or even ISet size ≤ 2 in some cases.

2) Multipower or Multirate: Fig. 8 extends the previous results to multiple powers or multiple rates. In both cases, we see the same effect as that observed with a single power and a single rate. In particular, in the multirate case, the maximum difference between the optimal value and the throughput obtained with ISet size ≤ 2 is just 6%, and the average difference is less than 3%. An important implication, which has also been shown in Section V-C, is that when high rates are available and can be used, an optimal configuration would rather have higher rates and lower spatial reuse than higher spatial reuse and lower rates. This can be intuitively explained by the diminishing gain of spatial reuse and the relative constant gain of increasing link rates.

VI. Tools for Approximate Solutions

Although our exact algorithms are very efficient, the pricing subproblem is still NP-hard so that the computation time will

eventually become impractical for large networks where we can only expect good approximations within a reasonable time. In this section, we propose and test two approximation algorithms.

A. Approximation Tools

We have developed two approximation tools based on our column-based algorithm. The first one has already been used in Section V-F and is based on limiting the ISet size to k, which clearly has an impact on the size of the problem to solve. As a first approximation, we modify our column-generation algorithm to search for an entering column only among ISets with size smaller than k, typically $k = 1, 2$, or 3. Our second approximation, called *partial pricing* is based on the following: Whenever algorithm `greedy` fails to find an off-basis column with a strictly positive reduced cost, we stop the calculation and do not enter the enumeration algorithm `enuoracle`. As we will show below, these approximations produce objective values very close to the optimal ones.

B. Results

We now show that our two approximation techniques are quite accurate in the sense that they produce almost optimal solutions in a small computation time.

As discussed in Section V-F, we have already seen from Figs. 7 and 8 that using ISet size $\leq k$ has an excellent accuracy when compared to the exact solution even for relatively small k. The throughput computed for ISet size ≤ 3 is always within 5% of the optimal value. We obtain similar results for other networks for which the optimal solutions can be computed in a reasonable time. This is an important result in that we have now at our disposal a *fast* and quasi-optimal approximate tool that we can use to generate configurations for very large networks. We can also see in the left-hand side of Fig. 9(a) that the partial pricing algorithm also produces very accurate solutions for a network with 50 nodes. Its performance is very close to ISet size ≤ 4, as can be seen by comparing Fig. 9(a) with Fig. 7(b). However, the computation time is much shorter than that needed by the optimal solution, as shown in the right-hand side of Fig. 9(a).

Using the approximation tools, we can study larger problems for which optimal solutions are not available. The two examples we show in Fig. 9(b) and (c) are for Rand40 with three powers and three rates and for Rand80 with one power and one rate. These cannot be solved exactly because they have a very large number of logical links. Essentially, the partial pricing and ISet size ≤ 2 approximations have almost the same computation time, but partial pricing usually yields much better throughput. However, this does not mean that the partial pricing approximation is always best. The configurations produced with ISet size ≤ 2 are much simpler since at most two links need to be scheduled at a time. Therefore, partial enumeration of level k might turn out to be more useful for practical network operation. It is also worth noting that the approximation ISet size ≤ 3, while quite accurate, is much too slow, and that is why the corresponding curve in Fig. 9(c) is limited to a small power range. As for engineering insights, it seems that we have confirmed our previous result to the effect that in a mesh network where the flows are to or from the gateway, a spatial reuse of 3 seems to be close to optimal.

VII. Proportional Fairness

In this section, we replace the max-min objective function by proportional fairness (PF), i.e., in the problem JP given in (4)–(7), we replace (4) by (12)

$$\max_{\boldsymbol{\lambda},\mathbf{x},\boldsymbol{\alpha}} \sum_f \log \lambda_f \tag{12}$$

and the original problem becomes a nonlinear program (NLP). Note that if we can enumerate all the ISets, which of course would restrict the size of the networks that we can handle, then we can use a generic NLP solver such as MINOS [27]. In order to solve larger problems, we have developed a computational technique based on column generation that we summarize here (for more details, see [28]).

A. Nonlinear Column Generation

Our approach, sketched below, is based on the *sequential linear programming* (SLP), also known as the *Frank–Wolfe* method [29], which is designed for problems with a nonlinear objective $U(\mathbf{y})$ and linear constraints $A\mathbf{y} = \mathbf{b}$:

1) Find an initial feasible solution $\mathbf{y}_0$.
2) At iteration i, let $\mathbf{y}_i$ be the current solution. A linearized version of the problem with a linear objective function $\langle \nabla U(\mathbf{y}_i)(\mathbf{y} - \mathbf{y}_i) \rangle$ and the original constraints is solved. This produces a vector $\mathbf{y}_i^*$ that is a vertex of the domain and a direction $\mathbf{d}_i = \mathbf{y}_i^* - \mathbf{y}_i$.
3) Find the step size $\tau_i \geq 0$ in the direction $\mathbf{d}_i$ by solving the one-dimensional nonlinear optimization $\max_{\tau_i} U(\mathbf{y}_i + \tau_i \mathbf{d}_i)$.

In general, both steps are calculated to optimality at each iteration. In our case, this is not really needed, and we can use fast approximations. The line search is terminated when a sufficient decrease has been obtained as determined by the two following rules:

1) *Armijo rule*: $U(\mathbf{y}_i + \tau_i \mathbf{d}_i) - U(\mathbf{y}_i) \geq c_1 \tau_i \langle \nabla U(\mathbf{y}_i) \mathbf{d}_i \rangle$;
2) *Curvature condition*: $|\langle \nabla U(\mathbf{y}_i + \tau_i \mathbf{d}_i) \mathbf{d}_i \rangle| \leq c_2 |\langle \nabla U(\mathbf{y}_i) \mathbf{d}_i \rangle|$

with $0 < c_1 < c_2 < 1$.

To compute the direction, we need to solve an LP, and we can use the greedy pricing algorithm where we do not necessarily run the procedure to termination. The idea is that the computation-intensive enumeration step is not invoked as long as we can make a sufficiently large progress with only the greedy pricing. This is equivalent to the two conditions

$$\frac{\langle \nabla U(\mathbf{y}_i) \mathbf{d}_i \rangle}{\|\nabla U\| \|\mathbf{d}} \| \geq \delta > 0 \tag{13}$$

$$\frac{\|\mathbf{y}_i - \mathbf{y}_{i-1}\|}{\|\mathbf{y}_i\|} \geq \theta > 0 \tag{14}$$

for certain parameters δ and θ. If neither condition holds, the enumeration-based column generator is used. This insures the eventual convergence to optimality since no feasible direction with an increasing value of the objective will be left out.

B. Numerical Results

The model with proportional fairness provides some more engineering insights. We have chosen to present the results in terms of the average flow rate as opposed to the value of the

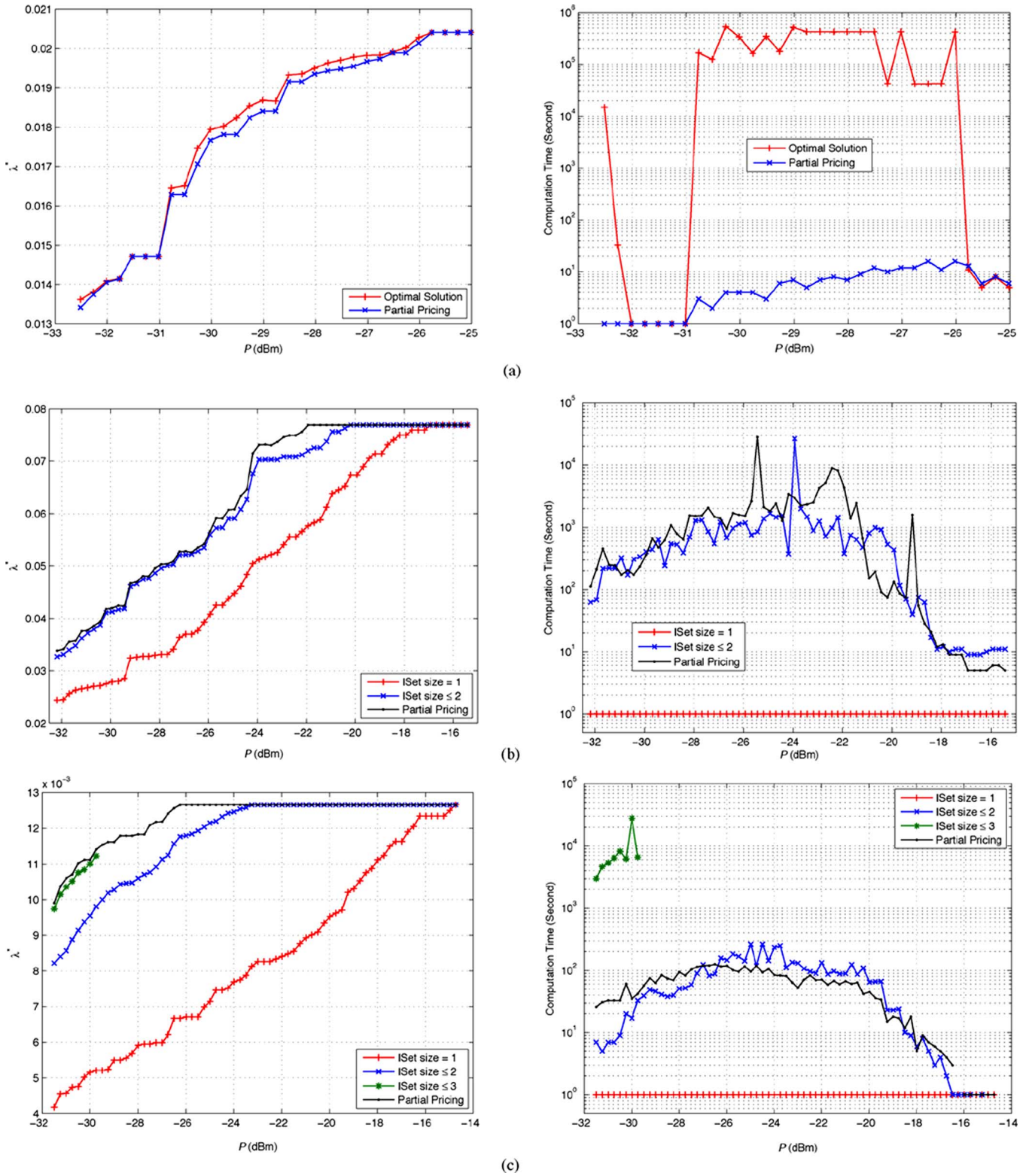

Fig. 9. Comparisons of different approximation mechanisms on different large scenarios: (a) Rand50b, one power, one rate, converging traffic; (b) Rand40, three powers, three rates, converging traffic; (c) Rand80, one power, one rate, converging traffic.

objective function since the fairness measure does not mean too much practically.

Fig. 10 shows the average rate of a flow as a function of P in the case of network Rand30a optimally configured for PF and in the case where a min-hop routing is used. The general trend of the throughput curves for PF is similar to what was found in Section V for the max-min objective both for the average rate per flow and also for the difference in the throughput produced by the optimal and min-hop routings.

Fig. 11 shows that the minimum throughput produced by the PF model is much lower than the minimum throughput produced by the max-min model. This is expected, but a quantitative mea-

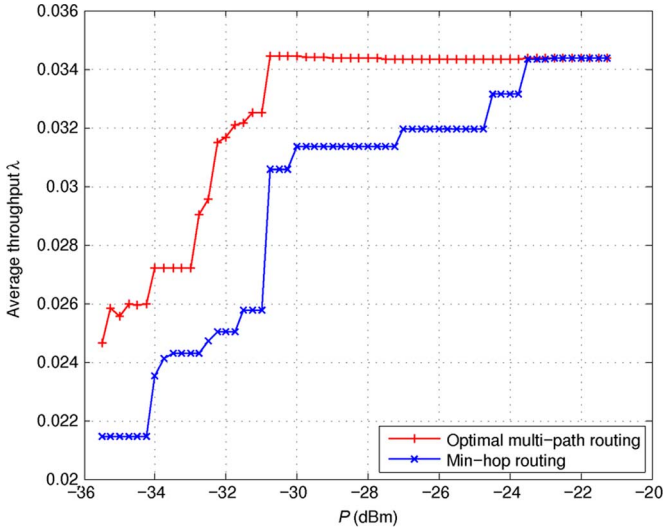

Fig. 10. Average flow rate for the case of PF for the Rand30a network.

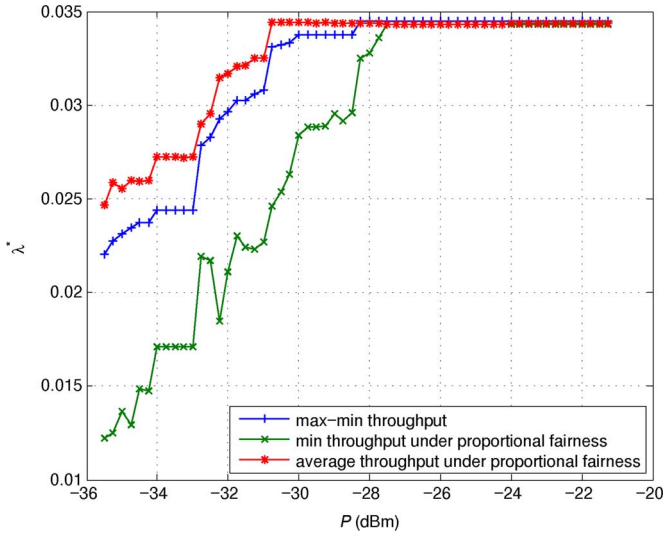

Fig. 11. Comparison of PF and max-min optimal configurations for the Rand30a network.

sure of this difference is possible only if a computation tool is available to solve the problems exactly.

A more unexpected result is provided by comparing in Fig. 11 the average flow rate obtained with PF and the max-min rate, where we see that the difference is never more than 12%. In practice, this difference could be even smaller since the max-min value is a lower bound on the average flow rate under the max-min model since there could be some flows with values larger than this. Hence, in a well-configured network, we could expect that max-min is almost as good as PF in terms of social welfare, i.e., the average flow rate.

VIII. Conclusion

This paper proposes a detailed and extensive study of the optimal configurations of fixed mesh networks using conflict-free scheduling. In the case of a max-min objective function, we confirm that power control is useful, but that the number of levels might be less important than the actual values that are used. We also quantify the advantage of multihop over single-hop, showing that multipath optimal routing is not much more efficient than single-path optimal routing and that not all min-hop routing is equally efficient. Moreover, we find that the relationship between spatial reuse and network performance is not that straightforward.

These results are obtained by developing two computational tools to solve exactly the joint routing, scheduling, power control, and rate adaptation problem. These tools allow us to calculate solutions for networks significantly larger than what is currently possible. The first one is based on linear programming and is useful when solving a set of problems with multiple input sets at the network layer. The second one is based on column generation and is much faster than the LP technique thanks to an efficient greedy pricing algorithm.

We also propose two approximation algorithms that are very fast and are shown to be nearly optimal for networks small enough that one can calculate an exact solution. They have been tested on networks up to 80 nodes, which shows that the design of WMNs of realistic sizes is now entirely feasible.

We then adapt our tools to the case of proportional fairness in the case of one power level and one rate and show some interesting engineering insights.

Finally, one should keep in mind that it is very hard to do routing, scheduling, and power and rate control in a real network. This requires that all the nodes be synchronized and must be done quickly in the presence of changing channel conditions. There is obviously a need for further work to check whether the engineering insights provided by our model still hold in a more dynamic situation.

Appendix
Link Weight for Greedy Pricing

We rewrite the SINR-based constraints (2) and (3) as follows:

$$\sum_{l\in\mathcal{L}_i} q_l \le 1 \qquad \forall i \in \mathcal{N} \tag{15}$$

$$\sum_{k\in\mathcal{L}} W_{kl} q_k \le \mathbf{1} \qquad \forall l \in \mathcal{L} \tag{16}$$

where W_{kl} represents the normalized interference from a link k to l, for $k \neq l$, and it has the form

$$W_{kl} = \begin{cases} \dfrac{M}{P_l G_l + M - \beta N_0}, & k = l \\ 0, & k \neq l; \exists i \in \mathcal{N}: l, k \in \mathcal{L}_i \\ \dfrac{\beta P_k G_{kl}}{P_l G_l + M - \beta N_0}, & \text{otherwise} \end{cases}$$

where M is some large constant. For constraint (15), we can construct the corresponding conflict graph as described in [7]. As a result, a greedy pricing can use the vertex degree Δ_l as a penalty factor in the link weight. For constraint (16), there is no straightforward graph representation. We instead use the *row interference index* $\sum_k W_{kl}$ and/or *column interference index* $\sum_l W_{kl}$ as penalty factors; they represent the total interference to link l and from link l (to other links), respectively. It is analogous to the vertex degree in a conflict graph. In summary, the link weight used in our greedy pricing can be expressed as $f_w(c_l\nu_l, \Delta_l, \sum_k W_{kl}, \sum_l W_{kl})$, where f_w is increasing in $c_l\nu_l$ and decreasing in Δ_l, $\sum_k W_{kl}$, and $\sum_l W_{kl}$.

REFERENCES

[1] C. Rosenberg, J. Luo, and A. Girard, "Engineering wireless mesh networks," in *Proc. of the 19th IEEE PIMRC*, 2008, pp. 1–6.

[2] *IEEE 802.16 Standard Group*, [Online]. Available: http://www.ieee802. org/16/

[3] A. Karnik, A. Iyer, and C. Rosenberg, "Throughput-optimal configuration of wireless networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 5, pp. 1161–1174, Oct. 2008.

[4] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," in *Proc. 46th Allerton Conf.*, 2008.

[5] C. Joo, X. Lin, and N. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multi-hop wireless networks," in *Proc. IEEE INFOCOM*, 2008, pp. 1103–1111.

[6] M. Johansson and L. Xiao, "Cross-layer optimization of wireless networks using nonlinear column generation," *IEEE Trans. Wireless Commun.*, vol. 5, no. 2, pp. 435–445, Feb. 2006.

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

[8] J. Zhang, H. Wu, Q. Zhang, and B. Li, "Joint routing and scheduling in multi-radio multi-channel multi-hop wireless networks," in *Proc. BroadNets*, 2005, pp. 631–640.

[9] M. Alicherry, R. Bhatia, and L. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *Proc. ACM MobiCom*, 2005, pp. 58–72.

[10] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *Proc. ACM MobiCom*, 2005, pp. 73–87.

[11] N. Garg and J. Könemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems," in *Proc. IEEE FOCS*, 1997, pp. 300–309.

[12] A. Iyer, C. Rosenberg, and A. Karnik, "What is the right model for wireless channel interference," *IEEE Trans. Wireless Commun.*, vol. 8, no. 5, pp. 2662–2671, May 2009.

[13] H. Zhai and Y. Fang, "Impact of routing metrics on path capacity in multirate and multihop wireless Ad Hoc networks," in *Proc. IEEE ICNP*, 2006, pp. 86–95.

[14] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length based scheduling and congestion control," in *Proc. IEEE INFOCOM*, 2005, vol. 3, pp. 1794–1803.

[15] A. L. Stolyar, "Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm," *Queue. Syst.*, vol. 50, no. 4, pp. 401–457, 2005.

[16] L. Georgiadis, M. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, no. 1, pp. 1–144, 2006.

[17] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer rate control in wireless networks," in *Proc. IEEE INFOCOM*, 2005, vol. 3, pp. 1804–1814.

[18] ILOG CPLEX 11.0. [Online]. Available: http://www.ilog.com/products/cplex/ Online. Available

[19] P. Björklund, P. Värbrand, and D. Yuan, "Resource optimization of spatial TDMA in Ad Hoc radio networks: A column generation approach," in *Proc. IEEE INFOCOM*, 2003, vol. 2, pp. 818–824.

[20] A. Capone and G. Carello, "Scheduling optimization in wireless MESH networks with power control and rate adaptation," in *Proc. IEEE SECON*, 2006, pp. 138–147.

[21] M. Cao, X. Wang, S.-J. Kim, and M. Madihian, "Multi-hop wireless backhaul networks: A cross-layer design paradigm," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 4, pp. 738–748, May 2007.

[22] M. Cao, V. Raghunathan, S. Hanly, V. Sharma, and P. Kumar, "Power control and transmission scheduling for network utility maximization in wireless networks," in *Proc. IEEE CDC*, 2007, pp. 5215–5221.

[23] V. Erceg, L. Greenstein, Y. Tjandra, S. Parkoff, A. Gupta, B. Kulic, A. Julius, and R. Bianchi, "An empirically-based path loss model for wireless channels in suburban environments," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 7, pp. 1205–1211, Jul. 1999.

[24] P. Gupta and P. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[25] L. Lasdon, *Optimization Theory for Large Systems*. New York: Macmillan, 1970.

[26] GLPK: The Gnu Linear Programming Kit. [Online]. Available: http://www.gnu.org/software/glpk/

[27] B. A. Murtaugh and M. A. Saunders, "MINOS 5.4 users guide," Dept. Operations Research, Stanford University, Stanford, CA, Tech. Rep. SOL 83-20R, Dec. 1993.

[28] J. Luo, A. Girard, and C. Rosenberg, "Efficient algorithms to solve a class of resource allocation problems in large wireless networks," in *Proc. WiOpt*, 2009, pp. 1–9.

[29] D. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA: Athena Scientific, 1999.

Jun Luo (M'09) received the Ph.D. in computer science from the Swiss Federal Institute of Technology in Lausanne (EPFL), Switzerland, in 2006.

From 2006 to 2008, he has worked as a Post-Doctoral Research Fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. In 2008, he joined the faculty of the School of Computer Engineering, Nanyang Technological University, Singapore, where he is currently an Assistant Professor. His research interests include wireless networking, distributed systems, multimedia protocols, network modeling and performance analysis, applied operations research, as well as network security. For more information, check http://www3.ntu.edu.sg/home/junluo/.

Dr. Luo is a Member of the Association for Computing Machinery (ACM).

Catherine Rosenberg received the Doctorat en Sciences degree in computer science from the University of Paris, Orsay, France, in 1986.

She is a faculty member with the University of Waterloo, Waterloo, ON, Canada, where she holds a University Research Chair. She has authored over 100 papers and has been awarded eight patents in the USA. Her research interests are broadly in networking, currently with an emphasis in wireless networking and in traffic engineering (quality of service, network design, and routing).

André Girard received the Ph.D. degree in physics from the University of Pennsylvania, Philadelphia, in 1971.

He is an Honorary Professor with INRS-EMT and an Adjunct Professor with Ecole Polytechnique of Montreal, QC, Canada. His research interests all have to do with the optimization of telecommunication networks, and in particular with performance evaluation, routing, dimensioning, and reliability. He has made numerous theoretical and algorithmic contributions to the design of telephone, ATM, and IP networks.