

# Mathematical Decomposition Techniques for Distributed Cross-Layer Optimization of Data Networks

Björn Johansson, *Student Member, IEEE*, Pablo Soldati, and Mikael Johansson, *Member, IEEE*

**Abstract**—Network performance can be increased if the traditionally separated network layers are jointly optimized. Recently, network utility maximization has emerged as a powerful framework for studying such cross-layer issues. In this paper, we review and explain three distinct techniques that can be used to engineer utility-maximizing protocols: primal, dual, and cross decomposition. The techniques suggest layered, but loosely coupled, network architectures and protocols where different resource allocation updates should be run at different time-scales. The decomposition methods are applied to the design of fully distributed protocols for two wireless network technologies: networks with orthogonal channels and network-wide resource constraints, as well as wireless networks where the physical layer uses spatial-reuse time-division multiple access. Numerical examples are included to demonstrate the power of the approach.

**Index Terms**—Congestion control, cross-layer protocol design, optimization, power control, scheduling, wireless networks.

## I. INTRODUCTION

**I**N ORDER TO meet the increased demand for high-bandwidth network services, it has become increasingly important to develop network control mechanisms that utilize the full capabilities of all of the layers. However, in many network technologies, including optical and wireless networks, there is an inherent coupling between different layers. Adjusting the resource allocation in the physical layer changes the average link rates, influences the optimal routing, and alters the achievable network utility. Under such coupling, optimizing only within layers will not be enough to achieve the optimal network performance, but the congestion control, routing, and physical-layer control mechanisms need to be jointly designed.

Recently, network utility maximization (NUM) has emerged as a powerful framework for studying such cross-layer issues (e.g., [1]–[5]). Although utility maximization is a mature subject in disciplines such as economics (e.g., [6]) its application to congestion control in data networks was pioneered by Kelly *et al.* [1] and by Low and Lapsley [2]. The initial work in the networking literature focused on understanding various network control schemes [e.g., transmission control protocol/active queue management (TCP/AQM) variants] in

the fixed Internet as algorithms for solving a performance optimization problem, but it has also been used to engineer new congestion control schemes, notably TCP FAST. The literature on utility maximization for networks with fixed link capacities is vast, and it is fair to say that there is now a relatively complete understanding of both equilibrium properties and of the dynamics of Internet congestion control (see, e.g., [7] for a recent survey). During the last couple of years, the basic model has been extended to include the effects of the physical layer and a number of cross-layer optimal protocols have been suggested for different wireless technologies (e.g., [4], [5], and [8]–[10]). However, one may argue that there has been limited innovation in terms of theoretical tools; almost all protocols have been designed using variations of the dual decomposition techniques employed in the initial work by Low and Lapsley.

One of the key contributions of this paper is to extend the theoretical toolbox available for studying NUM problems by giving an accessible, yet relatively comprehensive, overview of three alternative decomposition schemes from mathematical programming. We demonstrate how these schemes suggest network architectures and protocols with different properties in terms of convergence speed, coordination overhead and time-scales on which resource-updates should be carried out. Moreover, the techniques allow us to find distributed solutions to problems where the dual decomposition approach is not immediately applicable. Although these ideas have been pursued by the authors in a sequence of papers [8], [9], similar ideas have recently and independently been put forward by Palomar and Chiang [11]. The second key contribution is to show how the alternative decomposition techniques can be applied to design novel distributed protocols for two wireless network technologies: networks with orthogonal channels and network-wide resource constraints, and wireless networks, where the physical layer uses spatial-reuse time-division multiple access (TDMA).

This paper is organized as follows. In Section II, we present two networking problems that will be solved. Section III discusses how the choice of decomposition method can affect protocol properties and guide architectural considerations. Section IV provides an overview of three alternative decomposition principles from mathematical programming applied to the NUM problem. Section V reviews optimization flow control for networks with fixed link capacities. In Sections VI and VII, we show how the principles can be used to devise distributed algorithms for the two network scenarios. Considerable attention is given to the design of distributed mechanisms for allocating resources in the physical layer. Finally, Section VIII concludes the paper. Mathematical background and proofs are given in the appendix.

Manuscript received September 1, 2005; revised April 20, 2006. This work was sponsored in part by the Swedish Research Council, in part by the European Commission, and in part by the Swedish Agency for Innovation Systems.

The authors are with the Automatic Control Laboratory, School of Electrical Engineering, Royal Institute of Technology (KTH), 100 44 Stockholm, Sweden (e-mail: bjorn.johansson@ee.kth.se; pablo.soldati@ee.kth.se; mikael.johansson@ee.kth.se).

Digital Object Identifier 10.1109/JSAC.2006.879364

## II. NETWORK UTILITY MAXIMIZATION (NUM)

We consider a communication network formed by a set of nodes located at fixed positions. Each node is assumed to have infinite buffering capacity and can transmit, receive, and relay data to other nodes across communication links. The network performance then depends on the interplay between end-to-end rate selection in the transport layer, routing in the network layer, and resource allocation in the physical layer.

We model the network topology as a graph with  $L$  directed links shared by  $P$  source-destination pairs. To each source, we associate an increasing and strictly concave function  $u_p(s_p)$  which measures the utility source  $p$  has of sending at rate  $s_p$  and let  $u(s) = \sum_{p=1}^P u_p(s_p)$  denote the aggregate utility. We assume that data is routed along fixed paths, represented by a routing matrix  $R = [r_{lp}]$  with entries  $r_{lp} = 1$  if source  $p$  uses link  $l$  and 0 otherwise. Moreover, we let  $c_l$  denote the capacity of link  $l$ . The optimal network operation can be found by solving the NUM problem

$$\begin{aligned} & \text{maximize} && u(s) \\ & \text{subject to} && Rs \preceq c, \quad s \in \mathcal{S}, \quad c \in \mathcal{C} \end{aligned} \quad (1)$$

in variables  $s$  and  $c$ . In other words, the problem is to maximize aggregate utility by jointly choosing  $s$  and  $c$ , subject to the constraint that the total traffic across links must be below the offered link capacities ( $Rs \preceq c$ ) and restrictions on the admissible end-to-end rates and link capacities ( $s \in \mathcal{S}, c \in \mathcal{C}$ ). Specifically, the vector of end-to-end rates  $s$  must lie in a convex set  $\mathcal{S}$ , typically on the form  $\mathcal{S} = \{s | s_{\min} \preceq s \preceq s_{\max}\}$  or  $\mathcal{S} = \{s | s_{\min} \preceq s\}$ , while the capacity vector must lie in the (convex) multiuser capacity region  $\mathcal{C}$  of the system. Any pair  $(s, c)$  that satisfies the constraints of (1) is said to be *feasible*, and corresponds to an admissible network operation. We will make the following technical assumptions.

*Assumption A:* 1) The network is connected. 2) The utility functions,  $u_p(s_p)$ , are strictly concave, differentiable, increasing, and  $\lim_{s_p \rightarrow 0} u_p(s_p) = -\infty$ . 3) The problem is feasible and a strictly interior point exists.

The class of problems that fit into (1) is rather large, and to arrive at specific results we will focus on two (still quite broad) particular cases of (1). These problem classes are practically relevant and have an underlying structure that allows us to go all the way from new theoretical tools to novel distributed solutions to the utility maximization problem.

### A. Example: Network-Wide Resource Constraints

The model (1) is rather abstract, as it hides the complexity of optimizing the link-rate vector, e.g., allocating communication resources, such as time slots in a transmission schedule, transmission rates, powers, and bandwidths. In some cases, it is therefore more natural to be explicit about the resource dependence on the link rates, and use a model on the form

$$\begin{aligned} & \text{maximize} && u(s) \\ & \text{subject to} && Rs \preceq c(\varphi), \quad s_{\min} \preceq s \\ & && \sum_l \varphi_l \leq \varphi_{\text{tot}}, \quad 0 \preceq \varphi \end{aligned} \quad (2)$$

where  $s$  and  $\varphi$  have to be found to jointly maximize the aggregate utility, and where the link capacities are assumed to depend on a resource that has a network-wide constraint ( $\sum_l \varphi_l \leq \varphi_{\text{tot}}$ ). This formulation can model a wireless or optical network that uses orthogonal channels and supports dynamic allocation of spectrum between transmitters. The total resource constraint complicates the problem since the resource allocation has to be coordinated across the whole network. In Section VI, we will demonstrate how the tools presented in this paper allow us to develop distributed algorithms for solving the NUM problem with network-wide resource constraints.

### B. Example: Distributed Transmission Scheduling

In other cases, it is fruitful to keep the abstract formulation of (1) but to restrict the capacity region to have some specific structure. One interesting example is

$$\begin{aligned} & \text{maximize} && u(s) \\ & \text{subject to} && Rs \preceq c, \quad s_{\min} \preceq s, \quad c \in \mathcal{C}_1 \end{aligned} \quad (3)$$

where  $s$  and  $c$  have to be found to jointly maximize the aggregate utility, and where  $\mathcal{C}_1$  is a convex polytope, the convex hull of a finite set of points in  $\mathbb{R}^L$ . This seems very similar to the original problem, but the switch to a convex polytope as the feasible region, instead of only demanding the region to be convex, will prove to be crucial. This model captures networks that employ (possibly spatial-reuse) TDMA. Section VII demonstrates how the tools in this paper applied to this model suggest distributed mechanisms for constructing transmission schedules with multiple time slots.

## III. DECOMPOSITION AS GUIDING PRINCIPLE FOR DISTRIBUTED CROSS-LAYER PROTOCOL DESIGN

Modern networked systems are designed for a tradeoff between a multitude of objectives, including optimality of performance, simplicity and flexibility in implementation, operation and maintenance, as well as robustness to uncertainties and variations. Trying to master this complex tradeoff often results in highly structured designs and implementations, such as the layered architecture of the OSI reference model.

The approach advocated in this work relies on a mathematical network model that exposes the key interconnections between the network layers. Based on this model, we formulate the optimal network operation under user cooperation and cross-layer coordination as a global NUM problem. To transform the centralized optimization problem into distributed protocols, we must find efficient ways for guiding different network elements towards the common goal. Inspiration for such coordination schemes can be found in mathematical decomposition techniques. Applying decomposition techniques to the global optimization problem allows us to identify critical information that needs to be communicated between nodes and across layers, and suggests how network elements should react to this information in order to attain the global optimum. In many cases, the underlying structure is such that these optimal rate and resource allocation schemes suggested by the decomposition schemes reside in separate networking layers. The layers are only loosely coupled via a set of critical control parameters. It turns out that the basic analysis suggests that

these parameters are the Lagrange multipliers of the optimization problem.

An underlying assumption of this work is that if a solution procedure of decomposition type has mathematical convergence, then it corresponds to a possible network architecture. Inspired by [12], we can take a step further and make a conjecture that a computationally efficient solution method corresponds to a better way of organizing the networking stack than what a less computationally efficient method does. The different decomposition schemes allow us to develop network architectures and protocols with different properties in terms of convergence speed, coordination overhead, and the time-scale on which various updates should be carried out. In addition, they allow us to find distributed solutions to problems where the dual decomposition approach is not immediately applicable.

#### IV. MATHEMATICAL DECOMPOSITION TECHNIQUES

In this section, we will review three specific decomposition techniques from mathematical programming and demonstrate how they can be applied to the NUM problem. The development relies on some basic results on subgradients, which are reviewed in the appendix. Although several surveys over decomposition techniques have been written in the mathematical programming community, e.g., [12]–[14], their focus is typically on exploiting problem structure to improve computational efficiency. Our focus is different. We use mathematical decomposition techniques as guiding principle for protocol engineering. Rather than trying to subdivide the NUM into subproblems that can be solved efficiently (say, in terms of memory or CPU cycles), we use decomposition techniques to divide the optimization of a network-wide performance measure to functions that can be executed in a distributed manner, preferably using existing protocols (or slight variations thereof).

A crucial observation is that the link capacities  $c$  (or the resource allocation  $\varphi$ ) are *complicating variables* in the sense that if the link capacities are fixed, (1) is simply a optimization flow control problem which can be solved using the techniques in [1] and [2]. Below, we will review three classes of decomposition principles: primal, dual, and primal-dual. We use primal and dual in their mathematical programming meaning: primal indicates that the optimization problem is solved using the original formulation and variables, while dual indicates that the original problem has been rewritten using Lagrangian relaxation. Contrary to most literature on mathematical programming, which focuses on convex minimization problems, we present all results in the framework of concave maximization.

##### A. Dual Decomposition

Dual decomposition is sometimes referred to as *price-directed* decomposition. The name comes from the economic interpretation that the network is directed towards its optimal operation by pricing the common resources. The optimal operation of the network layer is then to maximize utility minus the total resource cost, while the physical layer should attempt to maximize total revenue. Constraints on the common resource are not

explicitly enforced, but the demand is aligned with the supply using a simple pricing strategy: increase the prices on resources that are in shortage and decrease the price of resources that are in excess.

Formally, we apply Lagrange duality to the coupling constraint of (1),  $Rs \preceq c$ , and form the partial Lagrangian

$$L(s, c, \lambda) = u(s) - \lambda^T Rs + \lambda^T c. \quad (4)$$

The *dual function* is defined as

$$g(\lambda) = \max_{s \in \mathcal{S}, c \in \mathcal{C}} L(s, c, \lambda).$$

Note that  $g(\lambda)$  is separable and can be written as

$$g(\lambda) = \underbrace{\max_{s \in \mathcal{S}} \{u(s) - \lambda^T Rs\}}_{\text{Network}} + \underbrace{\max_{c \in \mathcal{C}} \{\lambda^T c\}}_{\text{Resource Allocation}}.$$

Thus, to evaluate the dual function for a given price vector, we need to solve a network subproblem [which coincides with the end-to-end rate allocation problem in optimization flow control (14)] and a resource allocation subproblem.

If  $u^*$  denotes the optimal value of (1), then  $g(\lambda) \geq u^*$ , so we can think of  $g(\lambda)$  as an optimistic estimate of the total utility. Intuitively, the coupling constraint is not present as such, but accounted for using the pricing scheme. Suboptimal prices can be exploited by either the network layer or the physical layer to yield a value of  $g(\lambda)$  that exceeds  $u^*$ . The optimal prices are obtained by solving the *dual problem*

$$\begin{aligned} & \text{minimize} && g(\lambda) \\ & \text{subject to} && \lambda \succeq 0. \end{aligned} \quad (5)$$

The minimization tries to recreate the effect of link-rate constraints on the relaxed problem. Denote the optimal value of the dual problem by  $g^*$ . In general,  $g^* \geq u^*$ . When  $g^* = u^*$ , we say that *strong duality* holds, and the optimal value of the original problem can be computed via its dual. One condition that guarantees strong duality is Slater's condition: if there exists an interior point where strict inequalities hold, then strong duality holds. See [15] and [16] for more details on duality.

---

##### Algorithm 1: Dual

---

- 1) Let  $k = 0$  and  $\lambda^{(0)} \in \Lambda$ .
  - 2) **loop**
  - 3) Compute a subgradient at  $\lambda^{(k)}$ .
  - 4) Update  $\lambda^{(k+1)}$  via (6) and let  $k = k + 1$ .
- 

*Proposition 1:* Let  $\lambda \succeq 0$  be given, and let  $s$  and  $c$  be the associated optimal solutions to the network and resource allocation subproblems, respectively. Then, a subgradient of  $g(\lambda)$  at  $\lambda$  is given by  $c - Rs$ .

*Proof:* See [16, Sec. 8.1]. ■

Now that we have a subgradient available, we can solve the dual problem using the subgradient methods in the appendix. We update the prices according to

$$\lambda^{(k+1)} = \mathcal{P}_\Lambda \left\{ \lambda^{(k)} - \alpha^{(k)} \left( c^{(k)} - R s^{(k)} \right) \right\} \quad (6)$$

where  $\alpha^{(k)}$  is the step length and  $\mathcal{P}_\Lambda\{\cdot\}$  denotes projection on the positive orthant. From now on,  $\mathcal{P}_\mathcal{K}\{\cdot\}$  will denote projection on the set  $\mathcal{K}$ . We have the following proposition.

*Proposition 2:* Algorithm 1 converges to the optimal solution to (5) with step lengths fulfilling (32).

*Proof:* See the subgradient section in the appendix. ■

There are a number of issues in applying dual decomposition in a distributed setting. The first one is that convergence to the optimal point only holds for diminishing step-size sequences. Another, maybe more critical, issue is that the primal variables (the end-to-end rate and link-rate allocations) obtained for a given link price vector  $\lambda$  may not be feasible, even if the dual variables are set to their optimal values [12]. One of the simplest examples illustrating this is

$$\begin{aligned} & \text{maximize} && x \\ & \text{subject to} && x \leq K, \quad x \in \mathbb{R}. \end{aligned}$$

Strong duality holds and the Lagrangian is  $L(x, \lambda) = x - \lambda(x - K)$ . The optimal Lagrange multiplier is  $\lambda^* = 1$  yielding the optimal value  $g(\lambda^*) = K$ . However, the Lagrangian at  $\lambda^*$ ,  $L(x, \lambda^*) = K$ , is maximized by an arbitrary  $x$ , and depending on the implementation we can get primal solutions that are neither optimal nor feasible. Within linear programming, this property has been referred to as the *noncoordinability* phenomenon. In offline schemes, this problem can be circumvented if one can devise a method for supplying a feasible primal solution (see [3] for a simple primal heuristic for the problem at hand). The following result gives us conditions for convergence of the subproblem solutions.

*Proposition 3:* If the Lagrangian is maximized for a unique  $s$  and  $c$  for every  $\lambda \succeq 0$ , then the subproblem solutions corresponding to the optimal  $\lambda^*$  are primal optimal.

*Proof:* See the proof section in the appendix. ■

There are several approaches for attaining primal convergence in dual decomposition schemes. One approach is to add a strictly concave term to the maximization objective, as is done in *proximal point* methods (see, e.g., [17]). The original problem is then replaced by the equivalent formulation

$$\begin{aligned} & \text{maximize} && u(s) - \varepsilon \|c - \check{c}\|_2^2 \\ & \text{subject to} && R s \preceq c, \quad c \in \mathcal{C}, \quad \check{c} \in \mathbb{R}^L. \end{aligned}$$

This makes the dual function smooth, and convergence of the primal variables in the limit follows.

## B. Primal Decomposition

Primal decomposition is also called *resource-directive* decomposition. Rather than introducing a pricing scheme for the

common resources, the primal decomposition approach sequentially updates the resource allocation to maximize the total network utility. To perform the redistribution of resources, links need to estimate the marginal improvement in network performance that could be achieved if they were given an increase in their resource allocation. It turns out that this information can be obtained from the Lagrange multipliers of the optimization flow control problem: a large Lagrange multiplier indicates that a large increase in utility could be obtained by allocating more resources to the link, a small multiplier indicates the opposite.

Mathematically, primal decomposition relies on rewriting (1) in terms of the *primal function*

$$\nu(c) = \max_{s \in \mathcal{S}} \{u(s) | R s \preceq c\}.$$

The primal function is simply the stationary performance of the optimization flow control for the given resource allocation. Under the explicit model (2), it is more natural to consider the primal function as a function of  $\varphi$ , i.e.,

$$\nu(\varphi) = \max_{s \in \mathcal{S}} \{u(s) | R s \preceq c(\varphi)\}.$$

Note that the primal function is a pessimistic estimate of the achievable network utility, since the resource allocation may be fixed at suboptimal values. The optimal network utility can be found by solving the *primal problem*

$$\begin{aligned} & \text{maximize} && \nu(c) \\ & \text{subject to} && c \in \mathcal{C}. \end{aligned} \quad (7)$$

Although the primal function is potentially nonsmooth, a subgradient is given by the following proposition.

*Proposition 4:* Let  $\lambda$  be a vector of optimal dual variables for the optimization flow control problem. Assume that the allocated capacity  $c$  is such that there exist a strictly feasible flow vector  $s$ , i.e.,  $\exists s \in \mathcal{S} : R s \prec c$ . A subgradient of  $\nu(c)$  is given by  $\lambda$ .

*Proof:* See [16, Sec. 6.5.3]. ■

We can thus use the subgradient methods from the appendix to solve the primal problem, updating  $c$  using the iteration

$$c^{(k+1)} = \mathcal{P}_\mathcal{C} \left\{ c^{(k)} + \alpha^{(k)} \lambda^{(k)} \right\}. \quad (8)$$

We have the following proposition.

*Proposition 5:* Algorithm 2 converges to the optimal solution to (7) with step lengths fulfilling (32).

*Proof:* See the subgradient section in the appendix. ■

Contrary to dual decomposition, primal decomposition guarantees that the iterates remain feasible by construction.

---

## Algorithm 2: Primal

---

- 1) Let  $k = 0$  and  $c^{(0)} \in \mathcal{C}$ .
  - 2) **loop**
  - 3) Compute a subgradient at  $c^{(k)}$ .
  - 4) Update  $c^{(k+1)}$  via (8) and let  $k = k + 1$ .
-

### C. Primal-Dual Decomposition

In *primal-dual* decomposition schemes, one tries to exploit both primal and dual problem structures. One class of methods, sometimes called *mixed decomposition* applies price- and resource-directive to different components within the same system [18]. We will make use of an alternative decomposition scheme, called *cross decomposition* [19].

In essence, cross decomposition is an alternating price-directive and resource-directive decomposition approach. One alternates between the primal and dual subproblems and there is no master problem involved. In general, the pure cross decomposition approach does not converge. However, *mean value cross decomposition* (MVC), where one uses the mean value of all previous solutions, has recently been shown to convergence [20]. The MVC algorithm, see Algorithm 3, solves the following problem (presented in its original form):

$$\begin{aligned} & \text{maximize} && u(s) + v(c) \\ & \text{subject to} && A_1(s) + B_1(c) \preceq b_1 \\ & && A_2(s) + B_2(c) \preceq b_2 \\ & && s \in \mathcal{S}, \quad c \in \mathcal{C} \end{aligned} \quad (9)$$

where  $u(s)$  and  $v(c)$  are concave,  $A_1(s)$ ,  $A_2(s)$ ,  $B_1(c)$ , and  $B_2(c)$  are convex functions, and  $\mathcal{S}$  and  $\mathcal{C}$  are convex and compact sets. It is also assumed that for any  $c \in \mathcal{C}$  there exists a strictly interior point, implying that strong duality holds for the two coupling constraints. Define the partial Lagrangian as

$$L(s, c, \lambda) = u(s) + v(c) - \lambda^T (A_1(s) + B_1(c) - b_1)$$

and define  $K(\bar{c}, \bar{\lambda})$  for any  $\bar{c} \in \mathcal{C}$  and  $\bar{\lambda} \succeq 0$  as

$$K(\bar{c}, \bar{\lambda}) = \max_{s \in \mathcal{S}} \{L(s, \bar{c}, \bar{\lambda}) \mid A_2(s) + B_2(\bar{c}) \preceq b_2\}.$$

The primal subproblem is defined as

$$\begin{aligned} & \text{minimize} && K(\bar{c}, \lambda) \\ & \text{subject to} && \lambda \succeq 0 \end{aligned}$$

and the dual subproblem is defined as

$$\begin{aligned} & \text{maximize} && K(c, \bar{\lambda}) \\ & \text{subject to} && c \in \mathcal{C}. \end{aligned} \quad (10)$$

---

#### Algorithm 3: MVC

---

- 1) Let  $\bar{c}^{(0)} \in \mathcal{C}$ ,  $\bar{\lambda}^{(0)} \succeq 0$ , and  $k = 0$ .
  - 2) **loop**
  - 3) Solve the primal subproblem (11) for  $\bar{c}^{(k)}$  to get  $\lambda^{(k+1)}$ .
  - 4) Solve the dual subproblem (10) for  $\bar{\lambda}^{(k)}$  to get  $c^{(k+1)}$ .
  - 5) Update the averages with (12) and let  $k = k + 1$ .
- 

By strong duality (applicable by assumption), the primal subproblem can be rewritten as

$$\begin{aligned} & \text{maximize} && u(s) + v(\bar{c}) \\ & \text{subject to} && A_1(s) + B_1(\bar{c}) \preceq b_1, \quad s \in \mathcal{S} \\ & && A_2(s) + B_2(\bar{c}) \preceq b_2. \end{aligned} \quad (11)$$

The primal and dual subproblem are solved alternately for the mean values of all previous iterations, where the mean values are defined as

$$\bar{\lambda}^{(k)} = \sum_{i=1}^k \lambda^{(i)} / k \quad \text{and} \quad \bar{c}^{(k)} = \sum_{i=1}^k c^{(i)} / k \quad (12)$$

and we have the following proposition for these averages.

*Proposition 6:* Algorithm 3 converges to the optimal solution to (9), i.e.,  $\lim_{k \rightarrow \infty} \bar{c}^{(k)} = c^*$  and  $\lim_{k \rightarrow \infty} \bar{\lambda}^{(k)} = \lambda^*$  under the assumptions given.

*Proof:* See [20]. ■

### D. Saddle-Point Computations and Min–Max Games

An alternative path to finding primal-dual optimal solutions to the cross-layer NUM problem goes via the saddle-point characterization of optimal points. By weak duality, we have

$$\max_{s \in \mathcal{S}, c \in \mathcal{C}} \min_{\lambda \succeq 0} L(s, c, \lambda) \leq u^* \leq \min_{\lambda \succeq 0} \max_{s \in \mathcal{S}, c \in \mathcal{C}} L(s, c, \lambda).$$

This inequality, known as the *max–min inequality*, simply states that the primal problem underestimates the optimal value, while the dual problem overestimates it. Under strong duality, the inequality holds with equality as the primal and dual optimal values are equal. The optimal point can then be given the following alternative characterization:  $(s^*, c^*, \lambda^*)$  is a primal-dual optimal point to the cross-layer NUM problem if and only if  $(s^*, c^*) \in \mathcal{S} \times \mathcal{C}$ ,  $\lambda^* \succeq 0$  and  $(s^*, c^*, \lambda^*)$  forms a saddle point of the Lagrangian, in the sense that

$$L(s, c, \lambda^*) \leq L(s^*, c^*, \lambda^*) \leq L(s^*, c^*, \lambda)$$

for all  $(s, c) \in \mathcal{S} \times \mathcal{C}$ ,  $\lambda \succeq 0$  (cf. [17, Prop. 5.1.6]). In other words,  $\lambda^*$  minimizes  $L(s^*, c^*, \lambda)$ , while  $(s^*, c^*)$  maximizes  $L(s, c, \lambda^*)$ . One of the most well-known algorithms for finding saddle points is the algorithm due to Arrow–Hurwicz [21]

$$\begin{aligned} s^{(k+1)} &= \mathcal{P}_{\mathcal{S}} \left\{ s^{(k)} + \alpha^{(k)} \nabla_s L \left( s^{(k)}, c^{(k)}, \lambda^{(k)} \right) \right\} \\ c^{(k+1)} &= \mathcal{P}_{\mathcal{C}} \left\{ c^{(k)} + \alpha^{(k)} \nabla_c L \left( s^{(k)}, c^{(k)}, \lambda^{(k)} \right) \right\} \\ \lambda^{(k+1)} &= \mathcal{P}_{\Lambda} \left\{ \lambda^{(k)} - \alpha^{(k)} \nabla_{\lambda} L \left( s^{(k)}, c^{(k)}, \lambda^{(k)} \right) \right\} \end{aligned}$$

where  $\alpha^{(k)}$  is the step length. Although the iteration is not guaranteed to converge (unless one imposes the additional requirements of strict convexity-concavity [22]), it provides a unified view of the primal and dual decomposition methods. In particular, the decomposition schemes can be interpreted as methods that run the above updates on different time-scales. Primal decomposition lets the  $s$  and  $\lambda$  dynamics run on a fast time-scale (essentially, until convergence), while the resource updates are run on a slow time-scale. Similarly, dual decomposition can be seen as letting the  $s$  and  $c$  updates run on a fast time-scale, while the  $\lambda$  variables are updated slowly.

## V. OPTIMIZATION FLOW CONTROL

Since optimization flow control will be a basic building block in our novel schemes, this section contains a brief review of the work in [1] and [2]. A key assumption is that the optimal bandwidth sharing mechanism solves the NUM problem

$$\begin{aligned} & \text{maximize} && \sum_p u_p(s_p) \\ & \text{subject to} && Rs \preceq c, \quad s \in \mathcal{S} \end{aligned} \quad (13)$$

where the link capacity vector  $c$  is assumed to be fixed. A distributed solution to this problem can be derived via dual decomposition. Introducing Lagrange multipliers  $\lambda$  for the coupling constraints and forming the Lagrangian as in (4), one finds that the dual function

$$g(\lambda) = \max_{s \in \mathcal{S}} \sum_p \left\{ u_p(s_p) - s_p \sum_l r_{lp} \lambda_l \right\} + \sum_l \lambda_l c_l$$

is separable in end-to-end rates  $s_p$  and can be evaluated by letting sources optimize their rates individually based on the total congestion price along the end-to-end path, i.e., by letting

$$s_p = \arg \max_{z_p \in \mathcal{S}} u_p(z_p) - z_p \sum_l r_{lp} \lambda_l. \quad (14)$$

Moreover, the dual problem can be solved by the projected gradient iteration

$$\lambda_l^{(k+1)} = \mathcal{P}_\Lambda \left\{ \lambda_l^{(k)} + \alpha^{(k)} \left( \sum_p r_{lp} s_p^{(k)} - c_l \right) \right\}$$

where  $\alpha^{(k)}$  is the step length. Note that links can update their congestion prices based on local information: if the traffic demand across link  $l$  exceeds capacity, the congestion price increases; otherwise, it decreases. Convergence of the dual algorithm has been established in [2]. As argued in [2] and [23], the equilibrium points of a wide range of TCP protocols can be interpreted in terms of sources maximizing their marginal utilities (utilities minus resource costs). Link algorithms generate prices to align the sources' selfish strategies with the global optimum. Most of the common TCP/AQM variants can be identified with different utility functions and different laws for updating the link prices.

In the remaining parts of this paper, we will demonstrate how the decomposition techniques reviewed in Section IV allow us to extend the optimization flow control procedure to deal with the joint end-to-end rate and resource allocation problem for the network scenarios described in Section II.

## VI. EXAMPLE I: NETWORKS WITH ORTHOGONAL CHANNELS AND NETWORK-WIDE RESOURCE CONSTRAINT

As a first application, we consider the design of utility maximizing protocols for systems with orthogonal channels and a global resource constraint in the physical layer.

### A. Optimality Conditions and Decomposition Approaches

The problem formulation (2) can be simplified using the properties of the optimal point under the following assumptions.

*Assumption B:* 1) The channel capacities  $c(\varphi)$  are strictly concave, twice differentiable, increasing, and  $c(0) = 0$ . 2) Each row and column of the routing matrix  $R$  contains at least one positive entry. 3) The problem is feasible, i.e.,  $\sum_{l=1}^L (c_l^{-1}(\sum_{i=1}^N R_{il}(s_{\min} + \epsilon))) < \varphi_{\text{tot}}$  for some  $\epsilon > 0$ .

The routing matrix assumption implies that all sources are assumed to be sending and all links are used by at least one source. The assumption on  $s_{\min}$  and  $\varphi_{\text{tot}}$  means that  $\varphi_{\text{tot}}$  is large enough to allow all sources to send just above the minimum sending rate using the links indicated by the routing matrix. The optimal point can then be characterized as follows.

*Proposition 7:* Under assumptions A and B, the optimal point  $(s^*, \varphi^*, \lambda^*)$  to (2) is characterized by

$$\begin{aligned} \sum_l \varphi_l^* &= \varphi_{\text{tot}}, & Rs^* &= c(\varphi^*) \\ \lambda_l^* c_l'(\varphi_l^*) &= \nu^*, & \varphi_l^* &\geq \varphi_{\min}, \quad l = 1, \dots, L \\ s_p^* &\geq s_{\min}, & p &= 1, \dots, P. \end{aligned} \quad (15)$$

*Proof:* See the proof section in the appendix.  $\blacksquare$

Thus, in the optimal solution to (2), the common resource is fully utilized, all links are bottlenecks and the marginal link revenues  $\lambda_l^* c_l'(\varphi_l^*)$  are equal. One consequence of this is that it is possible to consider a simpler, but equivalent, problem

$$\begin{aligned} & \text{maximize} && u(s) \\ & \text{subject to} && Rs \preceq c(\varphi), \quad s_{\min} \preceq s \preceq s_{\max} \\ & && \sum_l \varphi_l = \varphi_{\text{tot}}, \quad \varphi_{\min} \preceq \varphi. \end{aligned} \quad (16)$$

The problems (2) and (16) are equal, in the sense that they share the same optimal solution. The crucial change is that  $\sum_l \varphi_l \leq \varphi_{\text{tot}}$  has been changed to  $\sum_l \varphi_l = \varphi_{\text{tot}}$ . Moreover, some bounds have been introduced; the upper bound,  $s_{\max}$ , on  $s$  and the lower bound,  $\varphi_{\min}$ , on  $\varphi$  are technical conditions that do not change the optimal point but make the analysis simpler. This simpler problem will be solved using two approaches, dual and primal decomposition. We also introduce the sets  $\mathcal{S} = \{s | s_{\min} \preceq s \preceq s_{\max}\}$  and  $\Phi = \{\varphi | \varphi_{\min} \preceq \varphi, \sum_l \varphi_l = \varphi_{\text{tot}}\}$ .

1) *Dual Approach:* Introducing Lagrange multipliers  $\lambda_l$ ,  $l = 1, \dots, L$ , for the capacity constraints in (16), we form the partial Lagrangian

$$L(s, \varphi, \lambda) = \sum_p u_p(s_p) - \lambda^T (Rs - c(\varphi))$$

and the associated dual function

$$g(\lambda) = \max_{s \in \mathcal{S}} \left\{ \sum_p u_p(s_p) - \lambda^T Rs \right\} + \max_{\varphi \in \Phi} \lambda^T c(\varphi).$$

Thus, the dual function decomposes into a network subproblem and a resource allocation subproblem. The network subproblem is identical to the end-to-end rate allocation problem in optimization flow control (14), while distributed solutions for the second subproblem will be developed in Section VI-B. Since

the link capacities are assumed to be strictly concave, the partial Lagrangian is strictly concave in  $(s, \varphi)$  and the dual function is differentiable [17, Prop. 6.1.1] and

$$\nabla g(\lambda) = c(\varphi^*(\lambda)) - Rs^*(\lambda).$$

Similarly to optimization flow control, the dual problem (5) can be solved using the projected (sub)gradient iteration

$$\lambda_l^{(k+1)} = \mathcal{P}_\Lambda \left\{ \lambda_l^{(k)} - \alpha^{(k)} \left( c_l(\varphi_l^{(k)}) - [Rs^{(k)}]_l \right) \right\} \quad (17)$$

where the step lengths  $\alpha^{(k)}$ , fulfill (32). This update can be carried out locally by links based on their current excess capacities.

---

#### Algorithm 4: Dual

---

- 1) Let  $k = 0$ ,  $\lambda^{(0)} \succ 0$ .
  - 2) **loop**
  - 3) Solve the network subproblem and the resource allocation subproblem for  $\lambda^{(k)}$  to get  $\varphi^{(k)}$  and  $s^{(k)}$ .
  - 4) Update  $\lambda^{(k)}$  via (17) and let  $k = k + 1$ .
- 

The optimization problem (16) can be solved with Algorithm 4, as shown in the following proposition.

*Proposition 8:* Under assumptions A and B, Algorithm (4) with step-sizes according to (32) converges to the optimal solution to (2), i.e.,  $\lim_{k \rightarrow \infty} \varphi^{(k)} = \varphi^*$ ,  $\lim_{k \rightarrow \infty} s^{(k)} = s^*$ ,  $\lim_{k \rightarrow \infty} \lambda^{(k)} = \lambda^*$ .

*Proof:* See the proof section in the appendix. ■

Note that the optimal resource allocation and source rates can be found in parallel, but the optimal solutions to both subproblems should be found before the dual variables are updated. From a practical perspective, even disregarding potential synchronization issues, this approach has the disadvantage that resource allocations have to be done at a fast time-scale and that the resource allocation algorithm (at least in the most basic analysis) has to be executed to optimality before the dual variables can be updated.

2) *Primal Approach:* As an alternative, we apply primal decomposition and rewrite (16) as

$$\begin{aligned} & \text{maximize} && \nu(\varphi) \\ & \text{subject to} && \varphi \in \Phi \end{aligned} \quad (18)$$

where we have introduced

$$\nu(\varphi) = \max_{s \in \mathcal{S}} \{u(s) | Rs \preceq c(\varphi)\}. \quad (19)$$

Note that  $\nu(\varphi)$  is simply the optimal network utility that can be achieved by optimization flow control under resource allocation  $\varphi$ . Consequently, to evaluate  $\nu(\varphi)$ , we can fix the resource allocation and execute the distributed optimization flow control until convergence.

Before attempting to solve the problem (18), we will establish some basic properties of  $\nu(\varphi)$ .

*Proposition 9:* Under assumptions A and B,  $\nu(\varphi)$  is concave and a subgradient,  $h(\varphi)$ , of  $\nu(\varphi)$  at  $\varphi$  is given by

$$h(\varphi) = (\lambda_1 c'_1(\varphi_1) \quad \dots \quad \lambda_L c'_L(\varphi_L))$$

where  $\lambda_l$  are optimal Lagrange multipliers for the capacity constraints in (19).

*Proof:* See the proof section in the appendix. ■

Since a subgradient of  $\nu$  is available, it is natural to use a projected subgradient algorithm

$$\varphi^{(k+1)} = \mathcal{P}_\Phi \left\{ \varphi^{(k)} + \alpha^{(k)} h(\varphi^{(k)}) \right\} \quad (20)$$

with diminishing step-size  $\alpha^{(k)}$  fulfilling (32). Here,  $\mathcal{P}_\Phi\{\cdot\}$  denotes distributed projection on the set  $\Phi$ , i.e., solves the following projection problem in a distributed fashion:

$$\begin{aligned} & \text{maximize} && -\|\varphi - \varphi^0\|_2^2 \\ & \text{subject to} && \varphi \in \Phi. \end{aligned} \quad (21)$$

The projection problem has a separable concave objective function since  $-\|\varphi - \varphi^0\|_2^2 = \sum_{l=1}^L -(\varphi_l - \varphi_l^0)^2$ , and can be solved using the techniques in Section VI-B.

---

#### Algorithm 5: Primal

---

- 1) Let  $\varphi^{(0)} \succ 0$  and  $k = 0$ .
  - 2) **loop**
  - 3) Solve (19) for  $\varphi^{(k)}$  to get a subgradient.
  - 4) Update  $\varphi^{(k+1)}$  via (20) and let  $k = k + 1$ .
- 

*Proposition 10:* Under assumptions A and B, Algorithm 5 with step-sizes according to (32) converges to the optimal solution to (2), i.e.,  $\lim_{k \rightarrow \infty} \varphi^{(k)} = \varphi^*$ ,  $\lim_{k \rightarrow \infty} s^{(k)} = s^*$ .

*Proof:* See the proof section in the appendix. ■

The primal method relies on solving the optimization flow problem on a fast time-scale and incremental resource updates in an ascent direction of the total network utility on a slower time-scale. The source rate and link price updates are carried out in a distributed way, similarly to optimization flow control. As we will show next, the resource update can be performed in a distributed manner that only relies on communication and resource exchanges between direct neighbors. Put together, this results in a completely distributed algorithm for the NUM problem.

#### B. Solving the Resource Allocation Subproblem

The simple resource allocation problem

$$\begin{aligned} & \text{maximize} && \sum_{l=1}^L f_l(\varphi_l) \\ & \text{subject to} && \sum_l \varphi_l = \varphi_{\text{tot}}, \quad \varphi_l \geq 0 \end{aligned} \quad (22)$$

is a standard problem in economics, and several solution approaches exist when  $f_l(\varphi_l)$  is strictly concave and twice differentiable. The problem is central to the primal and dual approach, since both the resource allocation subproblem in the dual approach and the distributed projection in the primal approach are of this type. The optimal point of (22) can be characterized by the Karush–Kuhn–Tucker (KKT) conditions [15]

$$\begin{aligned} f'_l(\varphi_l^*) &= \psi^* \text{ if } \varphi_l^* > 0 \text{ and } f'_l(\varphi_l^*) \leq \psi^* \text{ if } \varphi_l^* = 0 \\ \sum_l \varphi_l^* &= \varphi_{\text{tot}}. \end{aligned} \quad (23)$$

These properties can be exploited in the distributed search for an optimal point. We will present two algorithms that solve the resource allocation problem: the weighted gradient approach and the direct negotiation approach.

1) *Weighted Gradient Approach*: The algorithms in [24] and [25] solve (22) under the assumptions that  $f_l$  are concave, twice continuously differentiable, with the second derivative bounded below and above,  $m_l \leq f_l''(\varphi_l) < 0$ , with  $m_l$  known. The algorithm presented in [24] can also handle non-negativity constraints on resources by identifying the  $\varphi$ 's that will be zero at optimality and finding an appropriate starting point. The resource updates rely on nearest neighbor communication only, and can be written in vector form as

$$\varphi^{(k+1)} = \varphi^{(k)} + W \nabla f(\varphi^{(k)}). \quad (24)$$

We require that  $\mathbf{1}^T W = 0$ , which implies that new resource allocations will remain feasible (since  $\mathbf{1}^T \varphi^{(k+1)} = \mathbf{1}^T \varphi^{(k)} + \mathbf{1}^T W \nabla f(\varphi^{(k)}) = \mathbf{1}^T \varphi^{(k)}$ ). A simple way of guaranteeing convergence is that  $W$  should satisfy the following conditions (the *Metropolis* weight scheme from [25]):

$$W_{ij} = -\min \left\{ \frac{1}{|\mathcal{N}(i)|m_i}, \frac{1}{|\mathcal{N}(j)|m_j} \right\} + \epsilon, \quad j \in \mathcal{N}(i)$$

$$W_{ii} = -\sum_{j \in \mathcal{N}(i)} W_{ij}$$

$$W_{ij} = 0, \text{ otherwise}$$

where  $\epsilon$  is a small positive constant and  $\mathcal{N}(i)$  is the set of neighboring links to  $i$ . Note that the limitation that links should only be allowed to communicate and exchange resources with its neighbors turns up as a structural constraint on  $W$ .

The idea that nodes should give away resources to neighbors that have better use for them is very intuitive. In e.g., [26] a similar scheme, based on heuristics, is suggested to be used in dynamic synchronous transfer mode optical networks. In this scheme, the nodes have tokens that give them right to a certain bandwidth. The nodes are suggested to transfer tokens to a neighbor that have more use of the bandwidth; more precisely, a token is transferred if the expression  $|(priority\ of\ node\ i) \cdot (free\ channels\ of\ node\ i + 1) - (priority\ of\ node\ i + 1) \cdot (free\ channels\ of\ node\ i)|$  decreases by a token transfer. The weighted gradient algorithm gives a justification for this heuristic, and provides a precise definition of the priorities.

2) *Direct Negotiation Approach*: As an alternative, the resource allocation subproblem can be solved via direct negotiation. This scheme requires the network to be ordered in a ring structure (or, in fact, any other structure providing order). This is not a major restriction, since a similar structure is also needed for determining a starting point that guarantees non-negativity of the iterates of the weighted gradient method [24].

The approach is based on the so-called water-filling method, see, e.g., [15]. We define

$$h_i(\psi) = \begin{cases} \varphi_{tot}, & \text{if } \psi < f_i'(\varphi_{tot}) \\ (f_i')^{-1}(\psi), & \text{if } f_i'(\varphi_{tot}) \leq \psi < f_i(0) \\ 0, & \text{if } f_i(0) \leq \psi \end{cases}$$

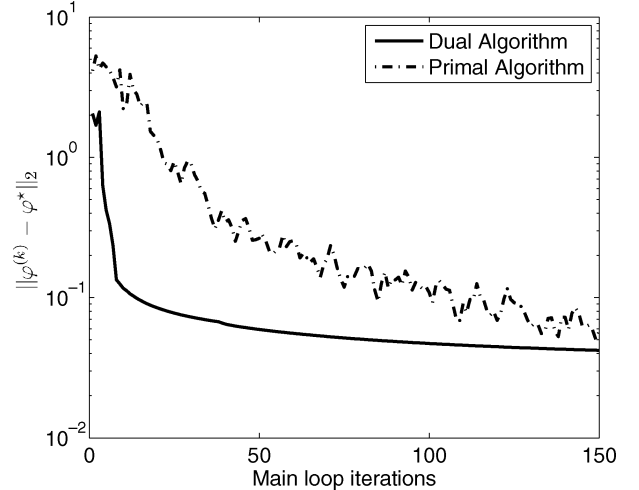


Fig. 1. The norm of the resource allocation minus the optimal resource allocation versus main loop iterations for the dual and primal algorithms.

which is the optimal resource allocation for the resource price  $\psi$ . Note that  $h_i$  is a continuous function, decreasing in  $\psi$  and that the inverse of  $f_i'(\varphi_i)$  is well-defined since  $f_i(\varphi_i)$  is strictly concave. Also, introduce  $h(\psi) = \sum_i h_i(\psi)$ , which is a sum of decreasing continuous functions, hence itself continuous and decreasing. Now, the water-filling method finds  $\psi^*$  such that

$$h(\psi^*) = \varphi_{tot}. \quad (25)$$

A lower bound for  $\psi^*$  is  $\underline{\psi} = \min_i f_i'(\varphi_{tot})$ , and an upper bound is  $\overline{\psi} = \max_i f_i'(0)$ , so  $\underline{\psi} \leq \psi^* \leq \overline{\psi}$ . Since  $h(\psi)$  is continuous and decreasing in  $\psi$ , we can use binary search to find  $\psi^*$  that fulfills (25). Start with taking  $\psi = (\overline{\psi} + \underline{\psi})/2$ . If  $h(\psi)$  is below  $\varphi_{tot}$ , then set the upper bound to  $\psi$ , i.e.,  $\overline{\psi} = \psi$ , but if  $h(\psi)$  is above  $\varphi_{tot}$ , then set the lower bound to  $\psi$ , i.e.,  $\underline{\psi} = \psi$ . Repeat until (25) is satisfied with desired accuracy.

The algorithm can be executed in a distributed way: One node takes the lead and announces the initial value of  $\psi$ . The node forwards the current value of  $\psi$  and the sum of claimed resources to the next node in the structure. After a complete cycle, the total sum of requested resources is available and the search interval for  $\psi^*$  can be cut in half as above. The process is repeated until the desired accuracy is achieved.

### C. Numerical Results

To illustrate the performance of the approaches, we apply the dual and the primal algorithms to a sample eight-node network used in [8]. The channel capacities are chosen to be the Shannon capacities, the resource is bandwidth and the utility functions are logarithmic  $u_p(s_p) = \log(s_p)$ . The routing matrix is not of full rank, but each row and column contain at least one positive entry.

Solving the optimization problem with the primal and the dual algorithms yields the results in Fig. 1. Both algorithms are converging to the optimal point. The dual algorithm descends in a smoother fashion, which can be explained by that the dual function is differentiable. However, the dual algorithm seems to be sensitive to the starting point and the initial step-size (this

cannot be seen in the current plot): if either of these is badly adjusted the convergence rate can be significantly reduced. Moreover, the iterates are only primal feasible in the limit. The primal algorithm exhibits the typical subgradient method behavior, i.e., the descent is not monotone. However, it seems to be more robust with respect to the starting point and initial step-size, and the iterates are always primal feasible.

## VII. EXAMPLE II: CROSS-LAYER OPTIMIZED SCHEDULING IN S-TDMA WIRELESS NETWORKS

Our second application considers NUM of wireless networks that employ spatial-reuse TDMA (S-TDMA). S-TDMA is a collision-free access scheme that allows spatially separated radio terminals to transmit simultaneously when the interference they incur on each other is not too severe. We will consider a particular instance of S-TDMA networks that offers a single communication rate,  $c_{\text{tgt}}$ , to all links that obey both primary and secondary interference constraints. The *primary interference* constraints require that a node communicates with at most one other node at a time (these constraints are typically imposed by the underlying technology, e.g., nodes equipped with omnidirectional antennas and no multiuser detectors). The *secondary interference* constraints require that the signal-to-interference-and-noise ratios (SINRs) at the receivers of active links exceed a target value

$$G_{ll}P_l / \left( \sigma_l + \sum_{j \neq l} G_{lj}P_j \right) \geq \gamma_{\text{tgt}}.$$

Here,  $P_l$  is the transmit power of link  $l$ ,  $\sigma_l$  is the thermal noise power at the receiver of link  $l$ , and  $G_{lj}$  denotes the effective power gain from the transmitter of link  $j$  to the receiver of link  $l$ . We say that a set of links is a *feasible transmission group* if there is a power allocation such that all links in the set obey the primary and secondary interference constraints. Associated to each feasible transmission group is a feasible link capacity vector, where  $c_l = c_{\text{tgt}}$  for all links  $l$  in the group and  $c_l = 0$  otherwise. By time-sharing over a large number of time slots, we can achieve any average link capacity vector in the convex hull of the feasible link capacity vectors.

### A. Decomposition Approach

It is important to understand that the classical approach of dual decomposition cannot be used for computing a schedule. The main reason is that the dual function

$$g(\lambda) = \max_{s \in \mathcal{S}} \{u(s) - \lambda^T R s\} + \max_{c \in \mathcal{C}_1} \{\lambda^T c\}$$

is linear in  $c_l$  and will return a single transmission group in each iteration. Since each transmission group only activates a few links, many links will have zero rates until the next iteration of the algorithm can be carried out by the system and a new transmission group activated. If the communication overhead for solving the scheduling subproblem is negligible, it may be viable to perform the scheduling computations in each time slot. However, as we will see below, negotiating for transmission rights and adjusting transmission powers may require substantial overhead in interference-limited systems. It is then more attractive to maintain a transmission schedule with multiple time

slots and update the schedule less frequently. As it turns out, a distributed algorithm for computing an asymptotically optimal schedule can be derived via mean value cross decomposition.

For technical reasons (compactness), we have to add the requirement that  $c \succeq c_{\min}$  to (3). If  $c_{\min}$  is chosen sufficiently small (this requires that  $s_{\min}$  is sufficiently small as well), then the modified problem will have the same optimal solution as the original one. Simulations indicate that if  $c_{\min}$  is small, it can in fact be neglected. Now, we use a mean value cross decomposition approach to solve this modified problem (an alternative one-sided approach is described in [9]). Recall that the mean value cross decomposition alternates between the primal and dual subproblems using the average values of the computed iterates as inputs. The primal subproblem

$$\begin{aligned} & \text{maximize} && u(s) \\ & \text{subject to} && R s \preceq \bar{c}^{(k)}, \quad s \in \mathcal{S} \end{aligned} \quad (26)$$

gives the optimal Lagrange multipliers  $\lambda^{(k)}$  for the capacity constraints, while the (relevant part of the) dual subproblem

$$\begin{aligned} & \text{maximize} && c^T \bar{\lambda}^{(k)} \\ & \text{subject to} && c \in \mathcal{C}_1, \quad c_{\min} \preceq c \end{aligned} \quad (27)$$

yields  $c^{(k)}$ . Since the primal subproblem is an instance of optimization flow control, mean value cross-decomposition suggests the following approach for solving the NUM problem (3): based on an initial schedule, we run the TCP/AQM scheme until convergence (this may require us to apply the schedule repeatedly). We refer to this phase as the data transmission phase. Nodes then record the associated equilibrium link prices for their transmitter queues and maintain their average values in memory. During the subsequent negotiation phase, we try to find the transmission group with largest average price-weighted throughput, and augment the schedule with the corresponding link capacity vector (effectively increasing the number of time slots in the schedule by one). If the time slots are of equal length, the offered link capacities of the resulting schedule will equal the average of all computed transmission groups. The procedure is then repeated with the revised schedule. Our algorithm is summarized in Algorithm 6.

---

### Algorithm 6: MVC

---

- 1) Let  $k = k_0$  and  $\bar{c}^{(k_0)} \succ c_{\min}$ .
  - 2) **loop**
  - 3) Solve (26) for  $\bar{c}^{(k-1)}$  to obtain  $\lambda^{(k)}$  and compute  $\bar{\lambda}^{(k)}$ .
  - 4) Solve (27) for  $\bar{\lambda}^{(k-1)}$  to obtain  $c^{(k)}$ .
  - 5) Augment the schedule, compute  $\bar{c}^{(k)}$ , and let  $k = k + 1$ .
- 

*Proposition 11:* Under assumption A, Algorithm 6 converges to the optimal solution to (3), i.e.,  $\lim_{k \rightarrow \infty} \bar{c}^{(k)} = c^*$ .

*Proof:* See the proof section in the appendix. ■

Note that an initial schedule can be constructed by letting  $k_0 = L$  and using a pure TDMA schedule. Our theoretical analysis applies to the case when we augment the schedule indefinitely, while in practice one would like to use schedules with limited frame length. Some suggestions for how the basic scheme can be adopted to this case can be found in [9].

### B. Solving the Scheduling Subproblem

The final component of a distributed solution to the NUM problem for S-TDMA networks is a distributed mechanism for solving the scheduling subproblem (27) in each negotiation phase. Although a fully distributed scheme that solves the subproblem to optimality appears out of reach, a suite of suboptimal schemes have been proposed and investigated in [9]. We will outline one of these approaches below.

Since the scheduling subproblem (27) is linear in  $c_l$ , an optimal solution can always be found at a vertex of the capacity region, i.e., among one of the feasible transmission groups. We will consider a distributed solution that is based on two logical steps: first, a candidate transmission group is formed by trying to maximize the objective function subject to primary interference constraints only; then, transmitters adjust their powers to allow the most advantageous subset of candidate links to satisfy the secondary interference constraints. Clearly, some links may need to drop out of the candidate set during the power negotiation phase, and the resulting transmission group may be suboptimal.

The candidate group formation is based on the observation that the primary constraints are satisfied if only one link in each two-hop neighborhood is activated. In an attempt to maximize the objective of the dual subproblem, the link with the highest average link price in a two-hop neighborhood will assign itself membership to the candidate set. To allow links to make this decision, we assume that the transmitters of each link forwards information about its link price to the receiving node. By collecting the maximum link prices from its neighbors, each node can decide if one of its own transmitters should enter the candidate set or remain silent.

Once a link has identified itself as a member of the candidate set, it will start contending for transmission rights. In our approach, links enter the transmission group one-by-one, adjusting their transmit powers to maximize the number of links in the transmission group. The approach exploits the properties of distributed power control with active link protection (DPC/ALP) [27]. The DPC/ALP algorithm is an extension of the classical distributed power control algorithms (e.g., [28]) which maintains the quality-of-service of operational links (link protection), while allowing inactive links to gradually power up in order to try to enter the transmission group. As interference builds up, active links sustain their quality while new ones may be blocked and denied channel access. The DPC/ALP algorithm exploits local measurements of SINRs at the receivers and runs iteratively. To describe the algorithm, we introduce  $\mathcal{A}$  and  $\mathcal{I}$  as the set of active and inactive links, and let  $\gamma_l$  be the measured SINR on link  $l$ . The DPC/ALP algorithm operates by updating the transmit powers  $P_l$  according to

$$P_l^+ = \begin{cases} \delta P_l \gamma_l^{\text{tgt}} / \gamma_l, & \text{if } l \in \mathcal{A} \\ \delta P_l, & \text{if } l \in \mathcal{I} \end{cases} \quad (28)$$

where  $\delta > 1$  is a control parameter and  $P_l^+$  denotes the next iterate of  $P_l$ . Links change status from inactive to active when their measured SINR exceeds the target. Inactive nodes that consistently fail to observe any SINR improvement enters a voluntary dropout phase and go silent (see [27] for details).

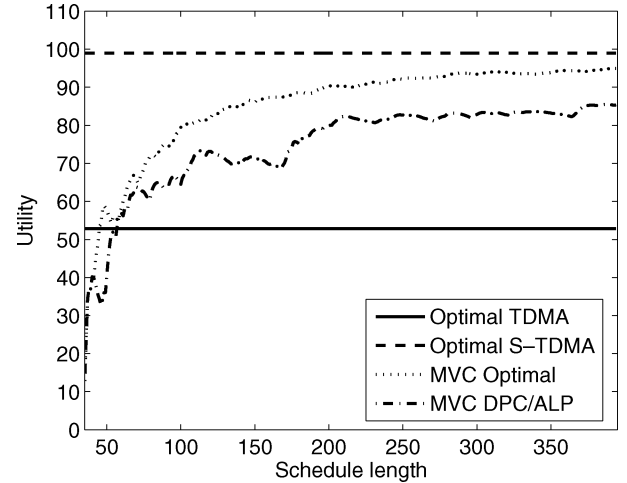


Fig. 2. Network utility as function of schedule length for a number of alternative schemes.

The negotiation phase is initialized by letting  $\mathcal{I}$  equal the candidate set and  $\mathcal{A}$  be empty. Links then execute the DPC/ALP algorithm. To increase the likelihood of forming a transmission group with high price-weighted throughput, we let links wait a random time before starting to execute the DPC/ALP algorithm. The waiting probability is a decreasing function of the link price, so that highly loaded links will tend to start ramping up transmit powers before lightly loaded ones (and thus, have increased chances to obtain transmission rights). At the end of the negotiation phase,  $\mathcal{A}$  constitutes a feasible transmission group.

### C. Numerical Results

We now demonstrate the typical performance of Algorithm 6 by applying it to the hypothetical indoor wireless local area network (LAN) scenario described in [29]. Fig. 2 shows the objective value versus schedule length. The MVC algorithms start with an initial schedule consisting of an equal time-slot length TDMA schedule. The solid line is the optimal (variable time-slot length) TDMA schedule, the dashed line is the optimal performance computed using the offline approach described in [29], the dotted line is the MVC algorithm with the subproblem solved to optimality, and finally, the dash-dotted line is the MVC algorithm with the DPC/ALP approach to solve the resource allocation. The MVC-based algorithms perform significantly better than the optimal (variable time-slot length) TDMA schedule.

## VIII. CONCLUSION

This paper has presented three distinct techniques that can be used to engineer utility-maximizing protocols: primal, dual, and cross decomposition. These techniques extend the theoretical toolbox available for studying NUM problems, motivate alternative network architectures, and suggest protocols where resource allocation updates should be run at different time-scales. In addition, we have detailed how these techniques can be used to design utility-maximizing mechanisms for two different networking technologies.

Although the theory for NUM is evolving quickly, much work remains to be done. This includes developing better tools for

analyzing protocol dynamics to guarantee stable and efficient system behavior, and improved support for understanding the dependencies that the protocols introduce between networking layers. On the practical side, complete implementations of the NUM-based protocols should be developed and their performance should be assessed in network simulators under realistic traffic and radio models, bringing the theory one step closer to real-world deployments.

## APPENDIX

### A. Gradient and Subgradient Methods

A more thorough background on the methods presented here are given in [15]–[17]. In this section, we focus on

$$\begin{aligned} & \text{maximize} && f(s) \\ & \text{subject to} && s \in S \end{aligned} \quad (29)$$

where  $f(s)$  is concave and  $S$  is convex and closed. One of the simplest methods for solving (29) is the gradient ascent algorithm, which attempts to maximize the increase of the objective function in each iteration by updating the current iterate in the direction of the gradient of  $f(s)$ . In many cases, however,  $f$  is not differentiable. A natural extension of gradients for non-differentiable functions are *subgradients*. Any vector  $\mu$  which satisfies

$$f(s) \leq f(y) + \mu^T(s - y), \quad \forall s \in S \quad (30)$$

qualifies as a subgradient of  $f$  at  $y$ . All subgradients are overestimators for concave functions. If there is only one unique subgradient, then the function is differentiable. A simple example of a concave nondifferentiable function is  $f(s) = -|s|$ . This function is nondifferentiable at  $s = 0$ , and the set of all subgradients at  $s = 0$  is  $\{a \mid -1 \leq a \leq 1\}$ .

Subgradients can be used instead of the gradient to find the maximum, and these subgradient methods take steps in the direction of a subgradient, i.e., proceed according to

$$s^{(k+1)} = \mathcal{P}_S \left\{ s^{(k)} + \alpha^{(k)} \mu^{(k)}(s^{(k)}) \right\}. \quad (31)$$

The subgradient method is outlined in Algorithm 7. Contrary to the gradient method, the objective value does not necessarily increase in each iteration of the subgradient method. Rather, the distance between the iterate and the optimal solution will decrease. To show asymptotic convergence, it is enough to use diminishing step-sizes and that the subgradients are bounded. The step-sizes that we will use fulfill

$$\sum_{k=1}^{\infty} \alpha^{(k)} = \infty, \quad \sum_{k=1}^{\infty} (\alpha^{(k)})^2 < \infty. \quad (32)$$

This is satisfied with, e.g.,  $\alpha^{(k)} = 1/k$ . The following proposition formalizes the convergence using such step-sizes.

*Proposition 12:* If a solution to problem (29) exists, the subgradients are bounded by some constant  $C$ , and the step-sizes are chosen to fulfill (32), then Algorithm 7 converges  $\lim_{k \rightarrow \infty} s^{(k)} = s^*$ .

*Proof:* See [16, Prop. 8.2.6]. ■

If a fixed step-size is used, then the best value so far will converge to an area around the optimal point as pointed out in the following proposition.

*Proposition 13:* If a solution to problem (29) exists, the subgradients are bounded by some constant  $C$ , and the step-sizes are set to a constant  $\alpha^{(k)} = \alpha$ , then the best value of Algorithm 7 converges to a ball around the optimal point  $\liminf_{k \rightarrow \infty} f(s^{(k)}) \leq f(s^*) + (\alpha C^2/2)$ .

*Proof:* See [16, Prop. 8.2.2]. ■

If the objective function is differentiable, then ordinary projected gradient methods can be used. If the gradient also is Lipschitz, i.e., if there exists  $K \geq 0$  such that  $\|\nabla f(x) - \nabla f(y)\| \leq K\|x - y\|$ ,  $\forall x, y \in \mathcal{S}$ , then a projected gradient method with fixed step-size can be used. In this method, the subgradient is replaced by the gradient and the diminishing step-size is replaced with a fixed step-size. Convergence of this method is shown in the following proposition.

*Proposition 14:* If a solution to problem (29) exists, the gradient is Lipschitz with Lipschitz constant  $K$ , and the step-size  $\alpha$  fulfills  $0 < \alpha \leq (K/2)$ , then Algorithm 7 converges  $\lim_{k \rightarrow \infty} s^{(k)} = s^*$ .

*Proof:* See [17]. ■

If the Lipschitz property does not hold, then a search can be done to find a step length that increases utility in each iteration. However, this requires global coordination and does not appear to be a viable alternative in our search for distributed algorithms.

---

### Algorithm 7: Subgradient Method

---

- 1) Let  $k = 0$  and  $s^{(0)} \in \mathcal{S}$ .
  - 2) **loop**
  - 3) Compute a subgradient at  $s^{(k)}$ .
  - 4) Update  $s^{(k+1)}$  via (31) and let  $k = k + 1$ .
- 

### B. Proofs

*Proof:* [Proof of primal convergence in the dual problem, Proposition 3]. There exist a dual optimal solution  $\lambda^*$ , strong duality is assumed to hold, and a primal feasible optimal solution exist. Therefore ([16, Prop. 6.1.1]), any optimal primal-dual solution must fulfill  $L(s^*, c^*, \lambda^*) = \min_{s \in \mathcal{S}, c \in \mathcal{C}} L(s, c, \lambda^*)$ . Since this is fulfilled for a unique pair  $(s^*, c^*)$  they must be primal optimal. ■

*Proof:* [Proof of optimal properties, Proposition 7]. Use the KKT conditions (see, e.g. [15]) to characterize the optimal point. The Lagrangian is

$$\begin{aligned} L = & \sum_p u_p(s_p) + \sum_i \lambda_i \left( c_i(\varphi_i) - \sum_j R_{ij}s_j \right) \\ & + \nu \left( \varphi_{\text{tot}} - \sum_l \varphi_l \right) + \sum_k (s_k - s_{\min})m_k + \sum_l \varphi_l m_l. \end{aligned}$$

The KKT conditions for the optimal point is

$$\begin{cases} \frac{\partial L}{\partial s_p} = u'_p(s_p^*) \\ -\sum_i \lambda_i^* R_{ip} + m_p^* = 0 \\ \frac{\partial L}{\partial \varphi_l} = \lambda_l^* c'_l(\varphi_l^*) \\ -\nu^* + n_l^* = 0 \\ \sum_l \varphi_l^* - \varphi_{\text{tot}} \leq 0, & (\sum_l \varphi_l^* - \varphi_{\text{tot}}) \nu^* = 0 \\ \nu^* \geq 0, & \sum_j R_{ij} s_j^* - c_i(\varphi_i^*) \leq 0 \\ \lambda_i^* \geq 0, & (\sum_j R_{ij} s_j^* - c_i(\varphi_i^*)) \lambda_i^* = 0 \\ s_k^* \geq s_{\min}, & (s_{\min} - s_k^*) m_k^* = 0 \\ m_k^* \geq 0, & \varphi_l^* \geq 0 \\ \varphi_l^* n_l = 0, & n_l^* \geq 0. \end{cases}$$

Eliminate  $m_s^*$  and  $n_l^*$

$$\begin{cases} \sum_l \varphi_l^* - \varphi_{\text{tot}} \leq 0, & (\sum_l \varphi_l^* - \varphi_{\text{tot}}) \nu^* = 0 \\ \nu^* \geq 0, & \sum_j R_{ij} s_j^* - c_i(\varphi_i^*) \leq 0 \\ \left( \sum_j R_{ij} s_j^* \right. \\ \quad \left. - c_i(\varphi_i^*) \right) \lambda_i^* = 0, & \lambda_i^* \geq 0 \\ \sum_i \lambda_i^* R_{ik} - u'_k(s_k^*) \geq 0, & s_k^* \geq s_{\min} \\ \nu^* - \lambda_l^* c'_l(\varphi_l^*) \geq 0, & \varphi_l^* (\nu^* - \lambda_l^* c'_l(\varphi_l^*)) = 0 \\ \varphi_l^* \geq 0, & (s_{\min} - s_k^*) \\ & \times (\sum_i \lambda_i^* R_{ik} - u'_k(s_k^*)) = 0. \end{cases}$$

Note that each row in  $R$  has positive entries by assumption. Combining this with  $\sum_j R_{ij} s_j^* - c_i(\varphi_i^*) \leq 0$  and  $s_j \geq s_{\min}$  yields  $\varphi_i^* > 0 \forall i$ . Similarly, combining the assumption that the columns in  $R$  have positive entries with  $\sum_i \lambda_i^* R_{ik} - u'_k(s_k^*) \geq 0$  and  $u'_k(s_k^*) > 0$ , gives that at least one  $\lambda_i^*$  is positive. Hence,  $\nu^* - \lambda_l^* c'_l(\varphi_l^*) \geq 0$  and  $c'_l(\varphi_l^*) > 0$  give that  $\nu^* > 0$ . Now,  $\varphi_l^* > 0$  and  $\varphi_l^* (\nu^* - \lambda_l^* c'_l(\varphi_l^*)) = 0$  imply that  $\lambda_l^* > 0$  and  $\nu^* = \lambda_l^* c'_l(\varphi_l^*) \forall l$ . Further,  $\nu^* > 0$  and  $\lambda_l^* > 0 \forall l$  give that  $\sum_l \varphi_l^* = \varphi_{\text{tot}}$  and  $R s^* = c(\varphi^*)$ . Finally, since  $\sum_p R_{lp} s_p \geq \sum_p R_{lp} s_{\min}$ , then  $\varphi_l \geq \varphi_{\min} = c_l^{-1}(\sum_p R_{lp} s_{\min})$ . ■

*Proof:* [Convergence proof of the dual algorithm, Proposition 8]. The subgradient (which also is the gradient)

$$\nabla g(\lambda) = c(\varphi^*(\lambda)) - R s^*(\lambda)$$

where  $s$  is in the interval  $s_{\min} \preceq s \preceq c(\varphi_{\text{tot}})$  and  $\varphi$  is in the interval  $\varphi_{\min} \preceq \varphi \preceq \varphi_{\text{tot}}$ . This implies that the subgradient is bounded as follows:

$$\|\nabla g(\lambda)\|_2 \leq \|c(\varphi_{\text{tot}}) + R c(\varphi_{\text{tot}})\|_2.$$

With the assumptions A and B, convergence now follows from [16, Prop. 8.2.6]. ■

*Proof:* [Proof of concavity and subgradient, Proposition 9]. By strong duality

$$\begin{aligned} \nu(\varphi) &= \min_{\lambda \succeq 0} \max_{s_{\min} \preceq s} \sum_{p=1}^P (u_p(s_p) - s_p q_p) + \sum_{l=1}^L \lambda_l c_l(\varphi_l) \\ &= \min_{\lambda \succeq 0} \tilde{g}(s(\lambda)) + \sum_{l=1}^L \lambda_l c_l(\varphi_l) \end{aligned}$$

with  $q_p = \sum_{l=1}^L r_{lp} \lambda_l$ . Thus, since  $\nu(\varphi)$  is the pointwise infimum of concave functions, it is concave. Let  $\lambda^*$  be the optimal Lagrange multipliers for a resource allocation vector  $\varphi$ . For any other resource allocation  $\tilde{\varphi}$ , it holds that

$$\begin{aligned} \nu(\tilde{\varphi}) &\leq \max_{s_{\min} \preceq s} \left\{ \sum_p u_p(s_p) - s_p q_p^* + \sum_{l=1}^L \lambda_l^* c_l(\tilde{\varphi}_l) \right\} \\ &\leq \nu(\varphi) + \sum_{l=1}^L \lambda_l^* c'_l(\varphi_l) (\tilde{\varphi}_l - \varphi_l) \end{aligned}$$

with  $q_p^* = \sum_{l=1}^L r_{lp} \lambda_l^*$ . This, by the definition of a subgradient, concludes the proof. ■

*Proof:* [Convergence proof of the primal algorithm, Proposition 10]. The subgradient is given by  $h(\varphi) = (\lambda_1 c'_1(\varphi_1) \cdots \lambda_L c'_L(\varphi_L))$ . Since  $h(\varphi)$  is continuous in  $\varphi$ , bounded for every  $\varphi$ , and  $\varphi$  lies in a compact set, the subgradient is bounded by (the finite value)  $D = \max_{\varphi_{\min} \preceq \varphi \preceq \varphi_{\text{tot}}} \|h(\varphi)\|_2$ . Together with the assumptions A and B, convergence now follows from [16, Prop. 8.2.6]. ■

*Proof:* [Convergence proof of the cross algorithm, Proposition 11]. The key is to identify the algorithm to be a mean value cross decomposition. We do the following identifications (with MVC notation to the left and S-TDMA algorithm notation on the right):

$$\begin{aligned} u(s) &= u(s) & v(c) &= 0 & A_1(s) &= R s & B_1(c) &= -c \\ b_1 &= 0 & A_2(s) &= 0 & B_2(c) &= 0 & b_2 &= 0. \end{aligned}$$

The MVC primal subproblem (11) is identified with the S-TDMA primal subproblem (26). The MVC dual subproblem (10) is identified with the S-TDMA dual subproblem (27). Hence, the S-TDMA algorithm is an MVC algorithm and convergence follows from [20]. ■

## REFERENCES

- [1] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, pp. 237–252, 1998.
- [2] S. H. Low and D. E. Lapsley, "Optimization flow control-I: Basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861–874, 1999.
- [3] L. Xiao, M. Johansson, and S. Boyd, "Simultaneous routing and resource allocation in wireless networks," *IEEE Trans. Commun.*, vol. 52, no. 7, pp. 1136–1144, Jul. 2004.
- [4] M. Chiang, "Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 1, pp. 104–116, Jan. 2005.
- [5] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2005, pp. 1804–1814.
- [6] K. J. Arrow and L. Hurwicz, *Essays in Economics and Econometrics*. Chapel Hill, NC: Univ. North Carolina Press, 1960, ch. Decentralization and Computation in Resource Allocation, pp. 34–104.
- [7] R. Srikant, *The Mathematics of Internet Congestion Control*. Cambridge, MA: Birkhäuser, 2004.
- [8] B. Johansson and M. Johansson, "Primal and dual approaches to distributed cross-layer optimization," in *Proc. 16th IFAC World Congr.*, Prague, Czech Republic, Jul. 2005, CD-ROM.
- [9] P. Soldati, B. Johansson, and M. Johansson, "Proportionally fair allocation of end-to-end bandwidth in STDMA networks," in *Proc. ACM MobiHoc*, Florence, Italy, May 2006, pp. 286–297.
- [10] J. Wang, L. Li, S. H. Low, and J. C. Doyle, "Cross-layer optimization in TCP/IP networks," *IEEE/ACM Trans. Netw.*, vol. 3, no. 13, pp. 582–595, Jun. 2005.

- [11] D. Palomar and M. Chiang, "Alternative decompositions and distributed algorithms for network utility maximization," in *Proc. IEEE GLOBECOM*, St. Louis, MO, Nov. 2005, pp. 2563–2568.
- [12] K. Holmberg, Primal and dual decomposition as organizational design: Price and/or resource directive decomposition, Dept. Math., Linköping Inst. Technol., Linköping, Sweden, LiTH-MAT-R-94-03, Dec. 1994.
- [13] L. Lasdon, *Optimization Theory for Large Systems*. New York: Macmillan, 1970.
- [14] O. E. Flippo and A. H. G. Rinnooy Kan, "Decomposition in general mathematical programming," *Math. Program.*, vol. 60, pp. 361–382, 1993.
- [15] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
- [16] D. Bertsekas, A. Nedić, and A. Ozdaglar, *Convex Analysis and Optimization*. Belmont, MA: Athena Scientific, 2003.
- [17] D. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 1999.
- [18] B. Obel, "A note on mixed procedures for decomposing linear programming problems," *Mathematische Operationsforschung und Statistik, Series Optimization*, vol. 9, pp. 537–544, 1978.
- [19] T. J. V. Roy, "Cross decomposition for mixed integer linear programming," *Math. Program.*, vol. 25, pp. 46–63, 1983.
- [20] K. Holmberg and K. Kiwiel, "Mean value cross decomposition for non-linear convex problems," *Optim. Methods Softw.*, vol. 21, no. 3, pp. 401–407, 2006.
- [21] K. J. Arrow, L. Hurwicz, and H. Uzawa, *Studies in Linear and Non-linear Programming*. Stanford, CA: Stanford Univ. Press, 1958.
- [22] A. S. Nemirovski and D. B. Judin, "Cesari convergence of the gradient method of approximating saddle points of convex-concave functions," *Soviet Math. Dokl.*, vol. 19, no. 2, pp. 482–486, 1978.
- [23] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Trans. Netw.*, vol. 4, no. 11, pp. 525–536, Aug. 2003.
- [24] Y. C. Ho, L. Servi, and R. Suri, "A class of center-free resource allocation algorithms," *Large Scale Syst.*, vol. 1, pp. 51–62, 1980.
- [25] L. Xiao and S. Boyd, "Optimal scaling of a gradient method for distributed resource allocation," *J. Optim. Theory Appl.*, vol. 129, no. 3, 2006.
- [26] C. Antal, J. Molnár, S. Molnár, and G. Szabó, "Performance study of distributed channel allocation techniques for a fast circuit switched network," *Comput. Commun.*, vol. 21, no. 17, pp. 1597–1609, 1998.
- [27] N. Bambos, S. C. Chen, and G. Pottie, "Channel access algorithms with active link protection for wireless communication networks with power control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 583–597, Oct. 2000.
- [28] G. Foschini and Z. Miljanic, "A simple distributed autonomous power control algorithm and its convergence," *IEEE Trans. Veh. Technol.*, vol. 42, no. 4, pp. 641–646, Nov. 1993.

- [29] M. Johansson and L. Xiao, "Cross-layer optimization of wireless networks using nonlinear column generation," *IEEE Trans. Wireless Commun.*, vol. 5, pp. 435–445, Feb. 2006.



**Björn Johansson** (S'04) received the M.Sc. degree in engineering physics from the Lund Institute of Technology, Lund, Sweden, in 2004. He is currently working towards the Ph.D. degree in telecommunications at the Automatic Control Laboratory, School of Electrical Engineering, Royal Institute of Technology (KTH), Stockholm, Sweden.

His research is focused on distributed resource allocation in networked systems.



**Pablo Soldati** received the M.Sc. degree in telecommunications engineering from the University of Siena, Siena, Italy, in 2004. He is currently working towards the Ph.D. degree in telecommunications at the Automatic Control Laboratory, School of Electrical Engineering, Royal Institute of Technology (KTH), Stockholm, Sweden.

His research interests are in the area of wireless networks and networked embedded systems.



**Mikael Johansson** (M'02) received the Ph.D. degree in automatic control from Lund University, Lund, Sweden, in 1999.

From 1999 to 2002, he was a Consulting Professor and Postdoctoral Scholar at Stanford University, Stanford, CA, and the University of California, Berkeley. He is currently an Associate Professor at the School of Electrical Engineering, Royal Institute of Technology (KTH), Stockholm, Sweden. His research interests include wireless systems, data networking, optimization, and hybrid and embedded

control. He has authored one book and numerous journal and conference publications in these areas, and is contributing to several national and international research projects in communications, computing and control.