Distributed Network Utility Maximization in Wireless Networks With a Bounded Number of Paths

André Schumacher*
Dept. of Information and Computer Science
Helsinki University of Technology
Andre.Schumacher@tkk.fi

Harri Haanpää
Dept. of Information and Computer Science
Helsinki University of Technology
Harri.Haanpaa@tkk.fi

ABSTRACT

We consider the fair multicommodity flow problem in multihop wireless networks. We optimize the flow between a number of source-destination pairs to achieve a fair allocation of network resources while satisfying node capacity constraints. To account for the limitations of wireless devices, we use a path-based formulation and allow only a bounded number of paths between each source-destination pair. We develop a dual decomposition based distributed algorithm for solving the problem, show the optimality of a stationary solution, and compare the performance of the algorithm with a centralized branch-and-bound method.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communications*

General Terms

Algorithms, Design, Performance

Keywords

Dual Decomposition, Multipath Routing, Congestion Control, Proportional Fairness, Multicommodity Flow

1. INTRODUCTION

The Network Utility Maximization (NUM) approach has been recently applied to various problems in the context of wired and wireless networks, and distributed systems in general. In combination with dual decomposition techniques from convex optimization theory, distributed algorithms for practical problems were obtained, such as power assignment and transmission scheduling in wireless networks [1], distributed coordination of cooperating agents [2], joint optimization of network flow and resource allocation in wireless networks [3], and multipath routing with capacity constraints [4, 5, 6, 7, 8]. NUM models are especially suitable for multihop wireless networks, as the resulting algorithms often allow efficient distributed optimization of nontrivial metrics. For an overview of dual decomposition and its applications see e.g. [9, 10] and the references therein.

We approach the fair multicommodity flow problem and propose an algorithm that only considers a bounded number of paths between each source-destination pair at a given time. The restriction on the number of paths is important in practice. For instance, connectivity typically changes often in wireless networks, and the amount of state information used for routing should therefore be kept low. We model user satisfaction with a fixed utility function and obtain a resource allocation that achieves *proportional fairness* as a special case when using a logarithmic utility function [11].

While similar problems have been considered in the literature, network utility maximization problems restricted to a bounded number of paths between each source-destination pair seem not to have received much attention. Lin and Shroff [6] assume a fixed set of paths and only discuss briefly how to incorporate finding alternate paths into their algorithm. Lestas and Vinnicombe [12] propose an algorithm that is based on the penalty function approach by Kelly et al. [11]. Their algorithm shares the basic structure of our proposed algorithm, the iterative optimization of path rates and prices with a varying set of available paths that are updated based on their price. However, as they note, the penalty function based approach tends to distribute the flow to a large number of paths, which is impractical.

We propose an algorithm based on the dual problem and investigate its performance compared to a centralized algorithm. Our algorithm provides for a distributed implementation, uses only simple arithmetic operations, and thus is suitable for implementation in multihop wireless networks. We show that if the number of paths allowed is sufficiently large, a stationary solution of our algorithm represents an optimal solution to the multicommodity problem. The rest of the paper is organized as follows. Section 2 gives a formulation of the fair multicommodity flow problem with path constraints. Section 3 develops an edge-flow formulation and presents a centralized algorithm for solving it. In Section 4, we develop a path-based formulation, apply decomposition techniques to obtain an algorithm suitable for distributed implementation, and prove the optimality of a stationary point. Section 5 contains simulation results for random network topologies and Section 6 presents our conclusions.

*Corresponding author. Research supported by the Helsinki Graduate School in Computer Science and Engineering.

PM²HW²N'08, October 31, 2008, Vancouver, BC, Canada.

2. FAIR MULTICOMMODITY FLOW

We consider the *fair multicommodity flow* problem, which is a Network Utility Maximization problem.There are N distinct communication pairs (s_n, t_n) where s_n is the source and t_n is the destination. For each pair (s_n, t_n) there is a utility function U_n which is strictly concave, differentiable, and increasing. The network utility is the sum of the utilities of each source-destination pair. We are given a directed graph $G(V, E)$ representing a wireless network, i.e., the vertices V represent network nodes and edges $(u, v) \in E$ if node u is able to transmit to node v. Note that this model does not take interference into account. Each node $v \in V$ has a node capacity c_v, which may model transmission time, battery power, etc. The goal is to maximize the sum of the utilities of the total out-flow of the source nodes such that the total amount of flow received and sent by each node v does not exceed c_v. We assume that there is at least one path of non-zero capacity from each source to its destination.

Instead of the more usual edge-based formulation of the multicommodity flow problem, we give a path-based formulation. Rather than presenting the model and algorithms in matrix notation, we use sums over variables in order to emphasize the dependency of variables for sources, intermediate nodes, paths, etc. This representation simplifies a later formulation as a distributed algorithm. Further, we limit the number of paths to be used between each source-destination pair by a constant K, resulting in the problem KNUM.

When K is small, the above problem becomes computationally complex. Wang et al. [13] show that NUM problems that optimize the selection of a single path for each communication pair are NP-hard. However, since any source-destination flow can be decomposed into at most $|E|$ paths and cycles [14], for large enough K, an optimal solution to KNUM is also an optimal solution of the multicommodity problem. In practice the minimum number of paths required to decompose a given source-destination flow might be much lower than $|E|$, but determining the minimum K is NP-hard [15]. In the following, we solve the multicommodity flow problem by solving the KNUM problem while assuming that K is large enough.

We introduce the variables and constants given in Table 1 and formulate the problem as follows.

$$\textbf{KNUM} \quad \max \quad \sum_{n'=1}^{N} U_{n'}\left(y_{n'}\right)$$

$$\text{s.t.} \quad \sum_{p \in P_n} x^p = y_n \tag{1}$$

$$\sum_{p \in P_S} 2\, r_v^p x^p \le c_v \quad \forall v \in V \setminus T \tag{2}$$

$$\sum_{p \in P_S \setminus P_n} 2\, r_v^p x^p + \sum_{p \in P_n} x^p \le c_v \quad \forall v \in \{s_n, t_n\} \tag{3}$$

$$|P_n| \le K, \quad P_n \subseteq \mathcal{P}_n \tag{4}$$

$$y_n \ge 0, \quad x^p \ge 0 \quad \forall p \in P_S,$$

where $1 \le n \le N$ and $S = (P_1, \ldots, P_N)$. Constraint (1) requires that the total outgoing flow y_n equals the sum of the path rates x^p over all paths $p \in P_n$, where $P_n \subseteq \mathcal{P}_n$ is a set of selected paths and $\mathcal{P}_n$ is the set of all paths from s_n to t_n. The binary constants r_v^p equal 1 if node v lies on path p and $r_v^p = 0$ otherwise. Constraints (2) and (3) require that the total flow processed by each node v does not exceed its capacity c_v. One needs to take into account that a node v receives and transmits traffic if it lies on a path p, unless $p \in P_n$ and $v \in \{s_n, t_n\}$, in which case v either receives or transmits. Equation (4) constraints the number of selected paths to at most K for any pair. In our model the capacity constraints are on the nodes, but it would be straightforward to use edge capacity constraints instead.

U_n	Utility function for pair (s_n, t_n).
$\mathcal{P}_n$	Set of all paths connecting source s_n to sink t_n.
$\mathcal{P}$	Set of all paths, i.e., $\mathcal{P} = \bigcup_n \mathcal{P}_n$; note that the $\mathcal{P}_n$ are pairwise disjoint.
S	Selection of paths, where $S = (P_1, \ldots, P_N)$ and each $P_n \subseteq \mathcal{P}_n$.
P_S	Set of selected paths, i.e., $P_S = \bigcup_n P_n$, where $S = (P_1, \ldots, P_N)$.
y_n	Total out-flow of source s_n.
x^p	Flow over path p.
r_v^p	Binary constant that set to one indicates that node v is on path p.
T	Set of terminals that includes all source and destination nodes, i.e., $T = \bigcup_n \{s_n, t_n\}$.

Table 1: Notation for problem KNUM.

In KNUM the number of possible paths from a source to a destination can be exponential in the number of nodes $|V|$. In Section 3, we develop a centralized algorithm that solves the equivalent problem KNUM-E, which is based on edge flow variables instead of path and path-rate variables. The number of variables in KNUM-E is polynomial in N, $|E|$, and K. In Section 4, we present an algorithm for KNUM that iteratively solves KNUM-P, which uses a fixed selection of paths but is otherwise identical to KNUM. To avoid oscillations in the path rates, observed earlier in e.g. [16], we apply proximal minimization techniques to KNUM-P and obtain problem KNUM-PQ, which has the same optimum.

3. CENTRALIZED ALGORITHM

We now reformulate problem KNUM so that an optimal solution can be obtained by a centralized algorithm for any K. The problem formulation and our branch-and-bound algorithm are similar to those of Caramia and Sgalambro [17], who propose a branch-and-bound algorithm for the *maximum concurrent k-splittable flow problem*, which is a variant of the *k-splittable flow problem* that was first addressed by Baier et al. [18]. In this problem, the objective is to maximize the fraction of demand that can be routed between the source-destination pairs using a bounded number of paths subject to link capacity constraints. Here, however, we maximize the total network utility, which is a concave function of the path flows subject to node capacity constraints.

3.1 Problem Formulation

Denote an edge from node u to node v either by (u, v) or e. For the edge-flow formulation of problem KNUM, we introduce additional variables δ_e^{nk}, which determine whether an edge e is available to route flow for the kth path of pair (s_n, t_n) in a given solution. Let $f_{(u,v)}^{nk}$ denote the amount of flow over edge (u, v) that originates from the kth path from source s_n to destination t_n. We again denote the total flow outgoing from s_n by y_n. Problem KNUM-E is then formulated as a convex mixed-integer nonlinear program.

$$\textbf{KNUM-E} \quad \max \quad \sum_{n'=1}^{N} U_{n'}\left(y_{n'}\right)$$

$$\text{s.t.} \quad y_n = \sum_{1\le k'\le K} \sum_{e\in O(s_n)} f_e^{nk'} - \sum_{e\in I(s_n)} f_e^{nk'} \tag{5}$$

$$-y_n = \sum_{1\le k'\le K} \sum_{e\in O(t_n)} f_e^{nk'} - \sum_{e\in I(t_n)} f_e^{nk'} \tag{6}$$

$$\sum_{e\in O(v)} f_e^{nk} - \sum_{e\in I(v)} f_e^{nk} = 0 \quad \forall v \in V \setminus \{s_n, t_n\} \tag{7}$$

$$\sum_{1\le n'\le N} \sum_{1\le k'\le K} \sum_{e\in I(v)\cup O(v)} f_e^{n'k'} \le c_v \quad \forall v \in V \tag{8}$$

$$f_e^{nk} \le \delta_e^{nk} \cdot c_v \quad \forall v \in V, \forall e \in O(v) \tag{9}$$

$$\sum_{e\in O(v)} \delta_e^{nk} \le 1, \quad \sum_{e\in I(v)} \delta_e^{nk} \le 1 \quad \forall v \in V \tag{10}$$

$$\sum_{e\in O(v)} \delta_e^{nk} - \sum_{e\in I(v)} \delta_e^{nk} = 0 \quad \forall v \in V \setminus \{s_n, t_n\} \tag{11}$$

$$\sum_{e\in I(s_n)} \delta_e^{nk} = 0, \quad \sum_{e\in O(t_n)} \delta_e^{nk} = 0 \tag{12}$$

$$y_n \ge 0,\ f_e^{nk} \ge 0,\ \delta_e^{nk} \in \{0,1\} \quad \forall e \in E,$$

where $1 \le n \le N$ and $1 \le k \le K$. Problem KNUM-E has several edge-flow variables for a single edge: each commodity corresponding to a single source-destination pair consists of K subcommodities. For each commodity n and path index k, there is a network flow *layer* f^{nk}. A value of 1 for δ_e^{nk} indicates that edge e lies on the kth path from s_n to t_n, i.e., that flow of the kth layer of (s_n, t_n) may be routed over e. Constraint (5) requires that the total outgoing flow y_n of s_n equals the sum over all K flow layers, where $I(v)$ and $O(v)$ denote incoming and outgoing edges incident to v, respectively. Equation (6) states that the total incoming flow of t_n has to equal y_n, and (7) is the *flow-balance* constraint, which has to be satisfied for each flow layer, except at source and destination nodes. Equation (8) is the capacity constraint and (9) guarantees that only selected edges carry flow.

As the δ_e^{nk} indicate edges that form the kth path of pair (s_n, t_n), they must satisfy certain constraints. Nodes on a path have at most one incoming or outgoing edge (10). Also, any node other than the source and destination must have an outgoing edge if and only if it has an incoming edge (11). Finally, sources must have no incoming edges and destinations no outgoing edges (12).

Problems KNUM and KNUM-E are equivalent. One can obtain from a solution to KNUM-E a solution to KNUM by following the δ_e^{nk} variables with value 1 starting at the sources and setting the path rates according to the f_e^{nk}. Although a solution to KNUM-E may contain isolated cycles, these do not affect optimality or constraint satisfiability. Conversely, a solution to KNUM-E is obtained by setting the δ_e^{nk} of edges e that are incident to consecutive nodes on the kth path from s_n to t_n to 1 (ordering the paths in P_n arbitrarily) and setting all others to 0. However, KNUM-E has $O(KN|E|)$ variables, while the number of paths selected in KNUM can be exponential in $|V|$.

3.2 Branch-and-Bound Algorithm

The formulation above lends itself to a branch-and-bound (BNB) algorithm that operates on a binary search tree containing values for the δ_e^{nk}. The algorithm employs a bounding heuristic that solves the problem with relaxed integrality constraints for the δ_e^{nk} at intermediate nodes in the search tree, i.e., with $\delta_e^{nk} \in \{0,1\}$ replaced by $0 \le \delta_e^{nk} \le 1$. To reduce the number of calls to the convex programming solver, the algorithm checks constraints (10)–(12) to prune branches leading to infeasible solutions.

Algorithm 1 is a fairly standard branch-and-bound method for finding the optimal paths. Initially, the integrality constraints for δ_e^{nk} are relaxed and the corresponding labels l_e^{nk} are set to *variable*. The optimal solution for the relaxed problem is computed, and if all δ_e^{nk} are integral, we are done; otherwise we choose a δ_e^{nk} variable to branch on, set the corresponding l_e^{nk} to *fixed*, and optimize the two subproblems where the δ_e^{nk} chosen for branching is set to 0 and 1, respectively. If at any point in the search the optimal value for a relaxed problem is worse than the best solution with all δ variables integral that has been found so far, that search branch is pruned, since introducing additional integrality constraints to the relaxed problem cannot improve the value of the optimum. For branching, we choose the δ_e^{nk} variable that is furthest away from being integral.

In Algorithm 1, `delta_feasible(`δ, l`)` checks for satisfiability of constraints (10)–(12), `compute_upper_bound(`δ, l`)` solves the relaxation of problem KNUM-E, and the check for integrality of δ_e^{nk} is performed by `is_integer_solution(`δ`)`.

3.3 Implementation

We implemented Algorithm 1 in C++ and used the Mosek solver [19] to obtain an optimal solution of the relaxation of KNUM-E. Several practical aspects needed to be considered. Firstly, we set a fractional value of δ_e^{nk} to 0, if the solver returns a zero value for f_e^{nk}. This rule can reduce the number of fractional δ_e^{nk} values for a given solution and does not break feasibility. Secondly, the values returned for

```
initially :
    for all 1 ≤ n ≤ N, 1 ≤ k ≤ K, e ∈ E
        l_e^nk ← variable
    best_value ← −∞
    current_bound ← compute_upper_bound(δ, l)
    if ( is_integer_solution (δ))
        best_value ← current_bound
        best_solution ← δ
    else
        branch_and_bound(δ, l)

function branch_and_bound(δ, l)
  choose e, n, k such that min{δ_e^nk, 1 − δ_e^nk} is maximized
  for val ∈ {0, 1}
    δ_e^nk ← val; l_e^nk ← fixed
    if ( delta_feasible (δ, l))
      current_bound ← compute_upper_bound(δ, l)
      if (current_bound > best_value)
        if ( is_integer_solution (δ))
          best_value ← current_bound
          best_solution ← δ
        else
          branch_and_bound(δ, l)
  l_e^nk ← variable
```

Algorithm 1: Branch-and-Bound for KNUM-E

the δ_e^{nk} are typically only close to optimal, depending on the convergence criterion of the solver. Consequently, one needs to tolerate a certain amount of non-integrality in the δ_e^{nk}.

4. DISTRIBUTED ALGORITHM

We next develop a distributed algorithm for KNUM using the technique of dual decomposition. We start with the case of a fixed set of paths. In Section 4.2, we present a method for maintaining a selection of paths based on their flow and dual cost, and prove optimality of a stationary solution. In Section 4.3, we propose a distributed implementation.

4.1 Fixed Selection of Paths

We use the notation in Table 1. Consider first a fixed selection of paths $S = (P_1, \ldots, P_N)$. The problem becomes

$$\textbf{KNUM-P} \quad \max \quad \sum_{n'=1}^{N} U_{n'}(y_{n'})$$

$$\text{s.t.} \quad \sum_{p\in P_n} x^p = y_n \tag{13}$$

$$\sum_{p\in P_S} 2\, r_v^p x^p \le c_v \quad \forall v \in V \setminus T \tag{14}$$

$$\sum_{p\in P_S\setminus P_n} 2\, r_v^p x^p + \sum_{p\in P_n} x^p \le c_v \quad \forall v \in \{s_n, t_n\} \tag{15}$$

$$y_n \ge 0, \quad x^p \ge 0 \quad \forall p \in P_S,$$

where $1 \le n \le N$. The difference between KNUM and KNUM-P is that in the latter the selection of paths is fixed. The objective function of KNUM-P is strictly concave in y_n but not in x^p, so the dual function may not be differentiable and a subgradient may result in oscillations [16]. To avoid this, we use proximal minimization and introduce additional variables $\bar{x}$ and a quadratic term to the objective function. This technique is discussed in [20, p. 232] and has been applied previously to NUM problems with fixed paths, e.g., by Lin and Shroff [21] and Wang et al. [16].

By introducing the $\bar{x}$, we obtain an objective function that is strictly concave in x and y for fixed $\bar{x}$ and the dual function becomes differentiable [20, p. 669]. We also replace each equality constraint by two inequality constraints.

$$\textbf{KNUM-PQ} \quad \max \quad \sum_{n'=1}^{N} U_{n'}(y_{n'}) - \frac{1}{2D}\sum_{p\in P_S}(x^p - \bar{x}^p)^2 \tag{16}$$

$$\text{s.t.} \quad \sum_{p\in P_n} x^p - y_n \le 0, \quad y_n - \sum_{p\in P_n} x^p \le 0 \tag{17}$$

$$\sum_{p\in P_S} 2\, r_v^p x^p \le c_v \quad \forall v \in V \setminus T \tag{18}$$

$$\sum_{p\in P_S\setminus P_n} 2\, r_v^p x^p + \sum_{p\in P_n} x^p \le c_v \quad \forall v \in \{s_n, t_n\} \tag{19}$$

$$y_n \ge 0, \quad x^p \ge 0, \quad \bar{x}^p \ge 0 \quad \forall p \in P_S, \tag{20}$$

where $1 \le n \le N$, and D is a positive scaling constant. KNUM-PQ is then identical to KNUM-P except for the quadratic term that equals zero at optimality. That is, if (x^*, y^*) be an optimum of KNUM-P, then $(x^*, \bar{x}^*, y^*)$ with $\bar{x}^* = x^*$ is an optimum of KNUM-PQ.

We now apply the Gauss-Seidel method [20, p. 219] to KNUM-PQ. Every iteration, we first optimize KNUM-PQ with x, y variable and $\bar{x}$ fixed; second, we optimize KNUM-PQ with $\bar{x}$ variable and x and y fixed to their previous values. The second step is trivial, as (16) is maximized at $\bar{x} = x$ for fixed x and y, so it only remains to consider the first.

KNUM-PQ for fixed $\bar{x}$ has a strictly concave objective function and linear constraints. Therefore, it has a unique solution, the corresponding dual function is differentiable, and dual decomposition techniques can be readily applied. Let us formulate the (partial) Lagrangian for KNUM-PQ for fixed $\bar{x}$ by introducing multipliers λ_v for the node capacity and μ_n^+ and μ_n^- for the total out-flow constraints.

$$\begin{aligned} L(x,\bar{x},y,\lambda,\mu^+,\mu^-) &= \sum_{n=1}^{N} U_n(y_n) - \frac{1}{2D}\sum_{p\in P_S}(x^p-\bar{x}^p)^2 \\ &- \sum_{v\in V\setminus T}\lambda_v\Big(\sum_{p\in P_S} 2r_v^p x^p - c_v\Big) - \sum_{n=1}^{N}(\mu_n^+-\mu_n^-)\Big(y_n - \sum_{p\in P_n} x^p\Big) \\ &- \sum_{1\le n\le N}\sum_{v\in\{s_n,t_n\}}\lambda_v\Big(\sum_{p\in P_S\setminus P_n} 2r_v^p x^p + \sum_{p\in P_n} x^p - c_v\Big) \end{aligned} \tag{21}$$

By grouping terms, we now obtain

$$\begin{aligned} L(x,\bar{x},y,\lambda,\mu^+,\mu^-) = \sum_{n=1}^{N}\Bigg[& U_n(y_n) - (\mu_n^+-\mu_n^-)y_n \\ & - \sum_{p\in P_n}\Big(\frac{1}{2D}(x^p-\bar{x}^p)^2 - (\mu_n^+-\mu_n^-)x^p \\ & + x^p\Big(\lambda_{s_n}+\lambda_{t_n}+\sum_{v\in V\setminus\{s_n,t_n\}} 2\lambda_v r_v^p\Big)\Big)\Bigg] + \sum_{v\in V}\lambda_v c_v. \end{aligned} \tag{22}$$

The value of the dual function $g(\lambda, \mu^+, \mu^-, \bar{x})$ to problem KNUM-PQ with fixed $\bar{x}$ is obtained by maximizing L over x and y for the given dual values and $\bar{x}$. We obtain

$$g(\lambda,\mu^+,\mu^-,\bar{x}) = \sup_{x\ge 0, y\ge 0}\left\{L(x,\bar{x},y,\lambda,\mu^+,\mu^-)\right\}. \tag{23}$$

The dual problem then becomes

$$\min \quad g(\lambda,\mu^+,\mu^-,\bar{x}) \tag{24}$$

$$\text{subject to} \quad \lambda \ge 0, \quad \mu^+ \ge 0, \quad \mu^- \ge 0. \tag{25}$$

Note that $\bar{x}$ is constant. We denote by $\gamma_p(\lambda)$ the *cost* of a path $p \in P_n$ for the source-destination pair n and by $\rho_v(x)$ the *load* – sum of incoming and outgoing flow – of a node v.

$$\gamma_p(\lambda) = \lambda_{s_n} + \lambda_{t_n} + \sum_{v\in V\setminus\{s_n,t_n\}} 2\,\lambda_v r_v^p$$

$$\rho_v(x) = \begin{cases} \sum_{p\in P_S\setminus P_n} 2\, r_v^p x^p + \sum_{p\in P_n} x^p & \forall v \in \{s_n, t_n\} \\ \sum_{p\in P_S} 2\, r_v^p x^p & \forall v \in V\setminus T \end{cases}$$

As the dual function is differentiable, we obtain the following partial derivatives for the dual function $g(\lambda, \mu^+, \mu^-, \bar{x})$.

$$\frac{\partial g}{\partial \lambda_v} = c_v - \rho_v(\tilde{x}), \quad \frac{\partial g}{\partial \mu_v^+} = \sum_{p\in P_n}\tilde{x}^p - \tilde{y}_n, \quad \frac{\partial g}{\partial \mu_v^-} = \tilde{y}_n - \sum_{p\in P_n}\tilde{x}^p, \tag{26}$$

$$\text{where} \quad \tilde{x}^p = \arg\max_{x\ge 0} L(x,\bar{x},y,\lambda,\mu^+,\mu^-) \tag{27}$$

$$\tilde{y}_n = \arg\max_{y\ge 0} L(x,\bar{x},y,\lambda,\mu^+,\mu^-). \tag{28}$$

The projected gradient method and the explicit solution to the primal subproblem result in the following update equations for all $1 \leq n \leq N$ and $v \in V$. Assuming the derivative U_n' of U_n is invertible, we obtain

$$x^p(k+1) = [\bar{x}^p + D(\mu_n^+(k) - \mu_n^-(k) - \gamma_p(\lambda(k)))]^+ \quad \forall p \in P_n \tag{29}$$

$$y_n(k+1) = [{U_n'}^{-1}(\mu_n^+(k) - \mu_n^-(k))]^+ \tag{30}$$

$$\lambda_v(k+1) = [\lambda_v(k) + \alpha(\rho_v(x(k)) - c_v)]^+ \tag{31}$$

$$\mu_n^+(k+1) = \left[\mu_n^+(k) + \alpha\left(y_n - \sum_{p \in P_n} x^p(k)\right)\right]^+ \tag{32}$$

$$\mu_n^-(k+1) = \left[\mu_n^-(k) - \alpha\left(y_n - \sum_{p \in P_n} x^p(k)\right)\right]^+, \tag{33}$$

where k is the iteration number, α is a sufficiently small step size, $[\cdot]^+$ is the projection on the non-negative orthant, and ${U_n'}^{-1}$ is the inverse function of the derivative of U_n. These update equations are suitable for distributed implementation, as long as each source s_n is aware of the cost $\gamma_p(\lambda(k))$ for each path $p \in P_n$. The other dual variables, $\mu_n^+(k)$ and $\mu_n^-(k)$ can be maintained by the sources themselves.

As strong duality holds, it can be shown that the solutions updated according to (29)–(33) converge to an optimal solution of KNUM-PQ with fixed $\bar{x}$ for any given selection of paths and $\bar{x}$, if the step size α is sufficiently small and the U_n are differentiable, strictly concave and increasing [22]. The optimality of a stationary point of the Gauss-Seidel method for KNUM-PQ with variable $\bar{x}$ then follows directly from the results for the Gauss-Seidel method.

4.2 Adaptive Path Selection

Until now, the path selection has been fixed in advance. Now we utilize the Lagrangian multipliers for the capacity constraints to select paths. Suppose we had chosen the set of all possible paths $S_c = (\mathcal{P}_1, \ldots, \mathcal{P}_N)$ as fixed set of paths for solving KNUM-PQ. Recall that $\mathcal{P}_n$ consists of all paths from s_n to t_n. KNUM-PQ with selection S_c is equivalent to KNUM. Although $|\mathcal{P}_n|$ can be exponential in $|V|$, from the Lagrangian (22) and the dual function (23) one can see that at an optimal solution only minimum cost paths from each s_n to t_n can have non-zero flow. Following this idea, we add a layer that iteratively updates a selection of paths by adding minimum cost paths and removing maximum cost paths if the maximum number of allowed paths K has been exceeded. The new values for the primal and dual variables are then determined by solving KNUM-PQ for the current choice of paths.

More precisely, let $S^i = (P_1^i, \ldots, P_N^i)$ be the path selection in iteration i and let $(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)$ be an optimal solution to KNUM-PQ with selection S^i. Denote by γ_p^i the cost of path p and by ρ_v^i the load of node v at the optimum. We update the selection according to the following rule. For a pair (s_n, t_n) let $p^{\min}$ be a minimum cost path from s_n to t_n under the current prices. Let $p^{\max}$ be a maximum cost path from s_n to t_n currently in P_n^i. If several paths in P_n^i have the same maximum cost, choose a path with the least flow. Now P_n^{i+1} is obtained by adding $p^{\min}$ to P_n^i, and if that would violate the maximum number of paths allowed in P_n^{i+1}, then we remove $p^{\max}$:

$$P_n^{i+1} = \begin{cases} (P_n^i \setminus \{p^{\max}\}) \cup \{p^{\min}\} & \text{if } |P_n^i \cup \{p^{\min}\}| > K \\ P_n^i \cup \{p^{\min}\} & \text{otherwise.} \end{cases} \tag{34}$$

$(P_1, \ldots, P_N) \leftarrow (p_1, \ldots, p_N)$, where p_n is the shortest hop−count path from s_n to t_n

pick some initial solution $(x(0), \bar{x}(0), y(0), \lambda(0), \mu(0))$
while true
  until convergence of $(x(t), \bar{x}(t), \lambda(t), \mu(t))$ do
    until convergence of $(\lambda(k), \mu(k))$ do
      for each pair (s_n, t_n) do
        for each path $p \in P_n^i$ do
          $x^p(k+1) \leftarrow [\bar{x}^p(t) + D(\mu_n^+(k) - \mu_n^-(k) - \gamma_p^i(\lambda(k)))]^+$
        $y_n(k+1) \leftarrow [{U_n'}^{-1}(\mu_n^+(k) - \mu_n^-(k))]^+$
        $\mu_n^+(k+1) \leftarrow [\mu_n^+(k) + \alpha(y_n - \sum_{p \in P_n} x^p(k))]^+$
        $\mu_n^-(k+1) \leftarrow [\mu_n^-(k) - \alpha(y_n - \sum_{p \in P_n} x^p(k))]^+$
      for each node $v \in V$ do
        $\lambda_v(k+1) \leftarrow [\lambda_v(k) + \alpha(\rho_v^i(x(k)) - c_v)]^+$
    let $(x(t), y(t), \lambda(t), \mu(t))$ be a stationary point of the update equations above
    $\bar{x}(t+1) \leftarrow x(t)$
  let $(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)$ be a stationary point of the update equations above
  for each pair (s_n, t_n) do
    let $p^{\min}$ be a minimum cost path from s_n to t_n
    let $p^{\max}$ be a maximum cost path of minimum flow in P_n^i
    if $|P_n^i \cup \{p^{\min}\}| > K$ then
      $P_n^{i+1} \leftarrow (P_n^i \setminus \{p^{\max}\}) \cup \{p^{\min}\}$
    else $P_n^{i+1} \leftarrow P_n^i \cup \{p^{\min}\}$

Algorithm 2: Algorithm for solving KNUM.

The complete method is given in Algorithm 2. To prove that a stationary point $(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)$, which satisfies (17)–(20), represents an optimum for KNUM-PQ with variable paths, note that conditions (35)–(39) suffice for optimality [22, p. 322] for any selection of paths $S = (P_1, \ldots, P_N)$.

$$\tilde{\lambda} \geq 0, \quad \tilde{\mu}^+ \geq 0, \quad \tilde{\mu}^- \geq 0 \tag{35}$$

$$\tilde{\mu}_n^+(\sum_{p \in P_n} \tilde{x}^p - \tilde{y}_n) = 0, \quad \tilde{\mu}_n^-(\tilde{y}_n - \sum_{p \in P_n} \tilde{x}^p) = 0 \tag{36}$$

$$\tilde{\lambda}_v(\rho_v(\tilde{x}) - c_v) = 0 \quad \forall v \in V \tag{37}$$

$$\tilde{y} = \arg\max_{y \geq 0} L(x, \bar{x}, y, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-) \tag{38}$$

$$(\tilde{x}, \tilde{\bar{x}}) = \arg\max_{(x,\bar{x}) \in \mathbb{R}_{\geq 0}^N \times \mathbb{R}_{\geq 0}^N} L(x, \bar{x}, y, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-), \tag{39}$$

where $1 \leq n \leq N$.

LEMMA 1. *For a fixed selection of paths, any stationary point $(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)$ of the algorithm described in Section 4.1 is optimal for* KNUM-PQ.

PROOF. First observe that every stationary point is primal feasible, i.e., satisfies constraints (17)–(20), because otherwise $(\tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)$ would not be stationary according to update rules (31)–(32). The solution is also dual feasible due

to the projection rule. From (31) we have

$$\rho_v(\tilde{x}(k)) - c_v \leq 0, \quad \rho_v(\tilde{x}(k)) < c_v \Rightarrow \tilde{\lambda}_v = 0 \quad \forall v \in V,$$

so the complementary slackness condition (37) is satisfied and similarly for $\tilde{\mu}^+$ and $\tilde{\mu}^-$ due to (32) and (33), respectively. What remains to be shown is that $\tilde{x}$, $\tilde{\bar{x}}$, and $\tilde{y}$ maximize the Lagrangian with respect to $\tilde{\lambda}$, $\tilde{\mu}^+$, and $\tilde{\mu}^-$. Since the Lagrangian (22) is strictly concave in y, we have

$$\frac{\partial L(x, \bar{x}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)}{\partial y_n} = 0$$

only at the unique optimum; this is equivalent to equality holding in (30). Also note that at convergence we need to have $\tilde{x} = \tilde{\bar{x}}$. For $\tilde{x}^p > 0$ it thus follows from (29) that

$$\tilde{\mu}_n^+ - \tilde{\mu}_n^- - \gamma_p(\tilde{\lambda}) = 0 \tag{40}$$

and therefore

$$\frac{\partial L(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)}{\partial x^p} = 0, \qquad \frac{\partial L(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)}{\partial \bar{x}^p} = 0.$$

For $\tilde{x}^p = 0$ it follows from (29) that $\tilde{\mu}_n^+ - \tilde{\mu}_n^- - \gamma_p(\tilde{\lambda}) \leq 0$ and therefore

$$\frac{\partial L(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)}{\partial x^p} \leq 0, \quad \frac{\partial L(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)}{\partial \bar{x}^p} = 0.$$

Thus, conditions (38) and (39) are also satisfied. □

Consider now the general case where the path update is performed according to (34).

LEMMA 2. *Upon convergence, for each source-destination pair (s_n, t_n), all paths $p \in P_n$ with non-zero flow have cost $\gamma_p(\tilde{\lambda}) = \tilde{\mu}_n^+ - \tilde{\mu}_n^-$, which is minimum among s_n–t_n paths.*

PROOF. Claim follows from (34) and (40). □

PROPOSITION 1. *If Algorithm 2 converges, it converges to an optimal path selection and rate allocation for* KNUM. *The resulting source-destination flows are optimal for the multicommodity flow formulation without path constraints.*

PROOF. Consider two path selections S_A and S_B, where $S_A = (P_1, \ldots, P_N)$ results from Algorithm 2 and $S_B = (\mathcal{P}_1, \ldots, \mathcal{P}_N)$ contains all possible paths between each pair. Problem KNUM-PQ with selection S_B is obviously equivalent to KNUM. Let $(\tilde{x}, \tilde{\bar{x}}, \tilde{y}, \tilde{\lambda}, \tilde{\mu}^+, \tilde{\mu}^-)$ be the optimal solution with selection S_A. We extend that solution to a solution for KNUM-PQ with path selection S_B by adding new variables: for each path in $S_B \setminus S_A$, we let $\tilde{x}^p = \tilde{\bar{x}}^p = 0$. We show that the new solution is stationary, so that its optimality follows from Lemma 1.

Consider any pair (s_n, t_n) and a path p from s_n to t_n in $S_B \setminus S_A$. In our construction, $\tilde{x}^p = \tilde{\bar{x}}^p = 0$. From Lemma 2 it follows that there cannot be any paths that have a lower cost than the paths in P_n, that is, $\gamma_p(\tilde{\lambda}) \geq \tilde{\mu}_n^+ - \tilde{\mu}_n^-$ for all $p \in S_B \setminus S_A$. From the update rule (29) it then follows that $\tilde{x}^p = \tilde{\bar{x}}^p = 0$ is stationary. Also, adding new path variables x^p and $\bar{x}^p$ does not affect the update rules for the other variables as long as the new path variables have value 0, which they do. Therefore, any solution resulting from the algorithm is an optimal solution for KNUM-PQ with path selection S_B and, equivalently, for KNUM. □

COROLLARY 1. *If the parameter K is too small to admit a solution to* KNUM *with the same objective value as an optimal solution for the multicommodity flow formulation, then Algorithm 2 does not converge.*

4.3 Distributed Implementation

The operations nodes perform in Algorithm 2 are purely local and only require information on the total outgoing path flow at each source node and the load ρ_v at each node v. If the traffic originated from sources can be approximated by a continuous network flow, then one can replace explicit message passing by passive measurements in order to estimate the load ρ. The path update according to (34) can be implemented efficiently as a distributed algorithm (see e.g. [23]).

In the algorithm as described, however, the primal variables x and y and the dual variables λ, μ^+, and μ^- must converge before updating $\bar{x}$. This would require careful synchronization between all source-destination pairs. To circumvent this problem, one can consider a modified version of the algorithm that updates $\bar{x}$ at the same time scale as λ, μ^+, and μ^- with a different step size β. In the limit, for $\beta/\alpha \to 0$, the two versions are then equivalent. For a similar algorithm it was shown in [6] that convergence for a fixed set of paths is guaranteed for sufficiently small β. Also, our proof of optimality of a stationary solution still applies to the modified algorithm.

The path updates depend on the node prices λ and should be executed only when the prices have converged. In principle, one could include a convergence test in the shortest-path method. Instead, we choose a low path-update frequency so that the λ have time to converge. To account for the possibly numerous nodes with zero price, we add a small constant so that among paths of equal cost shortest ones are preferred.

Algorithm 3 is executed by each source node s_n continuously. The functions `prim_local_conv` and `dual_local_conv` evaluate the convergence criterion of the primal and dual variables of source s_n, respectively. The convergence criterion for the node prices is evaluated by calling `lambda_conv` but could also be, in principle, integrated in the shortest-path algorithm. Additionally, in order to avoid race conditions that could lead to unfairness among the sources in terms of path-update intervals, a minimum number of iter-

for each path $p \in P_n$ do
  $x^p(t+1) = [\bar{x}^p(t) + D(\mu_n^+(t) - \mu_n^-(t) - \gamma_p(\lambda(t)))]^+$
  $\bar{x}^p(t+1) = [(1 - \frac{\beta}{D})\bar{x}^p(t+1) + \frac{\beta}{D}x^p(t+1)]^+$

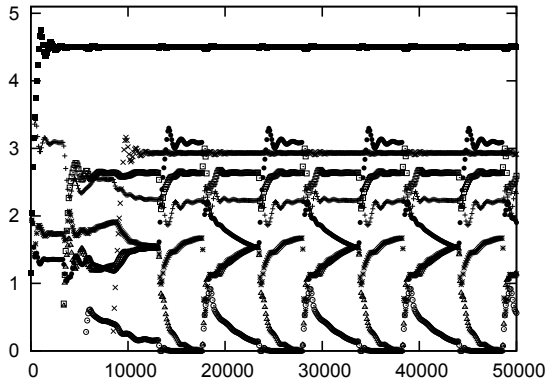
$y_n(t+1) = [{U'_n}^{-1}(\mu_n^+(t) - \mu_n^-(t))]^+$
$\mu_n^+(t+1) = [\mu_n^+(t) + \alpha(y_n - \sum_{p \in P_n} x^p(t))]^+$
$\mu_n^-(t+1) = [\mu_n^-(t) - \alpha(y_n - \sum_{p \in P_n} x^p(t))]^+$

if [prim_local_conv$(x_n(t), \bar{x}_n(t), y_n(t))$
  and dual_local_conv$(\mu_n^+(t), \mu_n^-(t))$ and lambda_conv$(\lambda(t))$
  and it_since_path_update $\geq$ min_it_before_path_update)]
  let $p^{\min}$ be a minimum cost path from s_n to t_n
  let $p^{\max}$ be a maximum cost path
    of minimum flow in P_n^i
  if $|P_n^i \cup \{p^{\min}\}| > K$ then
    $P_n^{i+1} \leftarrow (P_n^i \setminus \{p^{\max}\}) \cup \{p^{\min}\}$
  else $P_n^{i+1} \leftarrow P_n^i \cup \{p^{\min}\}$

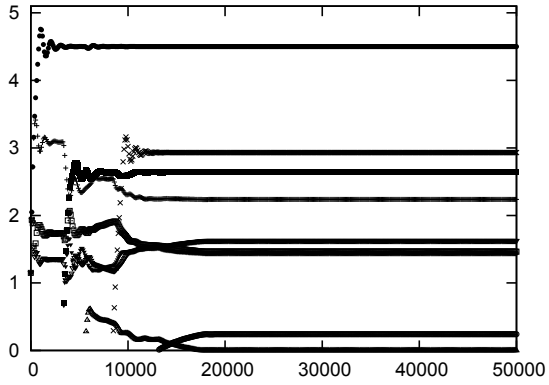
Algorithm 3: Algorithm running at the sources.

$\lambda_v(t+1) = \lambda_v(t+1) = [\lambda_v(t) + \alpha(\rho_v(x(t)) - c_v)]^+$

Algorithm 4: Algorithm running at all nodes.

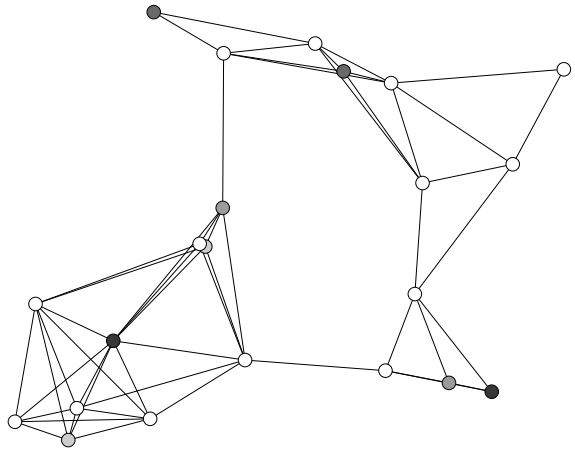


(a) $K = 2$

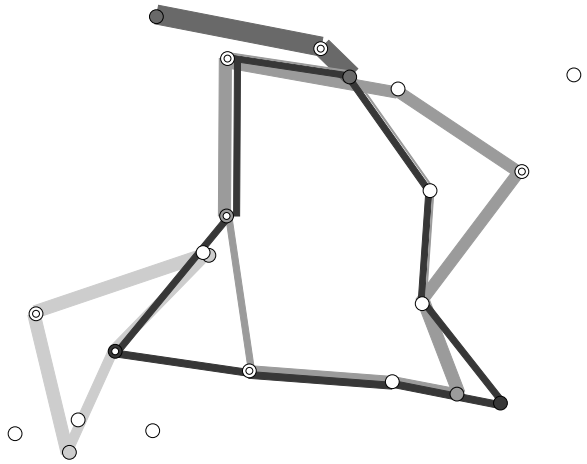


(b) $K = 3$

Figure 1: Evolution of path rates for each source-destination pair for the network in Figure 2.



(a) Input Graph; pairs are colored in different shades of gray; edges are drawn without direction for simplicity.



(b) Results for $K = 3$; saturated nodes are marked with inner circles; thickness of lines is proportional to the amount of flow colored according to its pair. One pair uses one, two pairs use two, and one pair uses three paths.

Figure 2: Disk graph with 22 nodes and 4 pairs.

ations is required to pass before a new path update. The update rule that affects all nodes, including the sources, is given in Algorithm 4. It only requires that each node measures the load it experiences in the network.

5. SIMULATIONS

In a distributed implementation of the methods described in the previous section, Algorithms 3 and 4 run concurrently. As a first step to validate convergence and estimate the impact of parameter K on the algorithm, we implemented a combined version of the updates performed by the source and intermediate nodes using Matlab. We leave the distributed implementation of the algorithm using network simulations and practical implementations for future work.

Consider the following simulation setup. A random disk graph $G(V, E)$ is generated by scattering $|V|$ nodes in the plane of unit dimensions. Initially $V = \{1, \ldots, |V|\}$, and E will contain (u, v) for any two nodes u and v whose distance is not larger than the transmission range, which is chosen to be close to 0.31. The node capacities c_v are drawn independently from the interval $[5, 10]$ uniformly at random. We choose $U_n = \log(y_n)$, therefore seeking a proportionally fair allocation of total outgoing-flow rates for the sources. We run the algorithm for various values of K and observe the evolution of path rate x^p for each path $p \in P_n$ and pair (s_n, t_n). We also calculate an optimal solution using Algorithm 1. If the graph is unconnected or if the Algorithm 1 does not terminate within four hours computing time on a 2GHz machine, the graph is discarded and the process is repeated. We let $\alpha = 10^{-3}$, $\beta = 10^{-2}$, and $D = 2^{-1}$.

Figure 1 shows the evolution of path rates for each of the 4 pairs in the network of Figure 2 for $K = 2$ and $K = 3$. For $K = 2$ one can see that the solution does not converge within $5 \cdot 10^4$ iterations and path rates oscillate. For all choices $K \geq 3$, however, the path rates x^p converge and also the y_n converge to the optimal total outgoing flow allocation. Note that not all of the paths for $K = 3$ actually carry flow.

For small enough network instances, we can run Algorithm 1 with various values for K starting at $K = 1$ to find K_m, the least K such that the optimal objective value of KNUM-E is equal to the optimal objective value of problem KNUM, which lacks constraints on the number of paths with non-zero flow. We continue increasing K to find K_{conv}, the smallest K for which our algorithm achieves convergence for a given network. The value $K_{conv} - K_m$ equals to the number of additional paths our algorithm requires to con-

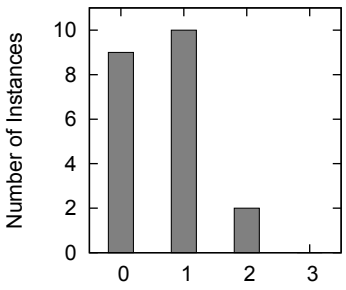


Figure 3: Number of additional paths $K_{conv} - K_m$ Algorithm 2 requires to achieve objective value U_{K_m} over a set of 21 random graph instances.

verge to the optimal utility for the multicommodity problem. Recall that $K_{conv} \geq K_m$ follows from Corollary 1. Figure 3 shows a histogram for $K_{conv} - K_m$ over a set of 21 networks generated randomly as described above with $|V| = 22$. One can see that the number of additional paths seems generally small and does not exceed 2 in any of our instances.

6. CONCLUSIONS

We studied the fair multicommodity flow problem and applied techniques from Network Utility Maximization theory to obtain an algorithm that maintains a path selection and updates the flow rate for each selected path. The number of selected paths is bounded by a constant K. If K is sufficiently large, we show that a stationary point of the algorithm corresponds to the multicommodity flow maximizing global network utility. This problem, contrary to the formulation with given paths where one optimizes only path rates, has only been rarely addressed in the literature. Our algorithm provides for a distributed implementation and therefore is suitable for implementation, e.g., in multihop wireless networks. We also present a centralized branch-and-bound algorithm, which we use for performance comparison.

7. REFERENCES

[1] Soldati, P., Johansson, B., Johansson, M.: Proportionally fair allocation of end-to-end bandwidth in STDMA wireless networks. In: Proc. 7th ACM Int. Symp. on Mobile ad hoc netw. and comp. (MobiHoc), New York, NY, USA, ACM Press (2006) 286–297

[2] Raffard, R., Tomlin, C., Boyd, S.: Distributed optimization for cooperative agents: application to formation flight. In: Proc. 43rd IEEE Conf. on Decision and Control. (2004) 2453–2459

[3] Xiao, L., Johansson, M., Boyd, S.P.: Simultaneous routing and resource allocation via dual decomposition. IEEE Trans. on Comm. **52** (2004) 1136–1144

[4] Presti, F.L.: Joint congestion control: routing and media access control optimization via dual decomposition for ad hoc wireless networks. In: Proc. 8th ACM Int. Symp. on Modeling, analysis and sim. of wireless and mobile sys. New York, NY, USA, ACM Press (2005) 298–306

[5] Low, S.: Multipath optimization flow control. In: Proc. 8th IEEE Int. Conf. on Networks, Washington, DC, USA, IEEE Comp. Soc. (2000) 39–43

[6] Lin, X., Shroff, N.B.: An optimization-based approach for QoS routing in high-bandwidth networks. IEEE/ACM Trans. Netw. **14**(6) (2006) 1348–1361

[7] He, J., Bresler, M., Chiang, M., Rexford, J.: Towards robust multi-layer traffic engineering: Optimization of congestion control and routing. IEEE J. Sel. Areas in Comm. **25** (2007) 868–880

[8] Paganini, F.: Congestion control with adaptive multipath routing based on optimization. In: Proc. 40th Ann. Conf. on Inf. Sci. and Sys. (2006) 333–338

[9] Chiang, M., Low, S.H., Calderbank, A.R., Doyle, J.C.: Layering as optimization decomposition: A mathematical theory of network architectures. Proc. IEEE **95** (2007) 255–312

[10] Johansson, B., Soldati, P., Johansson, M.: Mathematical decomposition techniques for distributed cross-layer optimization of data networks. IEEE J. Sel. Areas in Comm. **24**(8) (2006) 1535–1547

[11] Kelly, F., Maulloo, A., Tan, D.: Rate control in communication networks: shadow prices, proportional fairness and stability. J. Op. Res. Soc. **49** (1998) 237–252

[12] Lestas, I., Vinnicombe, G.: Combined control of routing and flow: a multipath routing approach. In: Proc. 43rd IEEE Conf. on Decision and Control. (2004) 2390–2395

[13] Wang, J., Li, L., Low, S., Doyle, J.: Can shortest-path routing and TCP maximize utility. In: Proc. 22nd Ann. Joint Conf. IEEE Comp. and Comm. Soc. (INFOCOM). (2003) 2049–2056

[14] Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Network Flows: Theory, Algorithms, and Applications. Prentice Hall, Englewood Cliffs, NJ (1993)

[15] B. Vatinlen, F. Chauvet, P.C., Mahey, P.: Simple bounds and greedy algorithms for decomposing a flow into a minimal set of paths. Europ. J. Operational Research **185** (2008) 1390–1401

[16] Wang, W.H., Palaniswami, M., Low, S.H.: Optimal flow control and routing in multi-path networks. Perform. Eval. **52**(2-3) (2003) 119–132

[17] Caramia, M., Sgalambro, A.: An exact approach for the maximum concurrent k-splittable flow problem. Optimization Letters (2007)

[18] Baier, G., Köhler, E., Skutella, M.: On the k-splittable flow problem. In: Proc. 10th Ann. Europ. Symp. on Algorithms, London, UK, Springer-Verlag (2002) 101–113

[19] MOSEK ApS: The MOSEK optimization tools manual. Version 5.0. (2008) Avail. for download at `http://www.mosek.com`.

[20] Bertsekas, D., Tsitsiklis, J.: Parallel and distributed computation: Numerical methods. Prentice Hall (1989)

[21] Lin, X., Shroff, N.B.: Utility maximization for communication networks with multipath routing. IEEE Trans. Automatic Control **51** (2006) 766–781

[22] Bertsekas, D.P.: Nonlinear Programming, second edn. Athena Scientific (1999)

[23] Haldar, S.: An 'all pairs shortest paths' distributed algorithm using $2n^2$ messages. J. Algorithms **24**(1) (1997) 20–36