Non-Concave Network Utility Maximization: A Distributed Optimization Approach

Mahmoud Ashour*, Jingyao Wang†, Constantino Lagoa*, Necdet Aybat‡ and Hao Che§
* The Department of Electrical Engineering and Computer Science, The Pennsylvania State University.
† The Department of Mechanics and Engineering Science, Peking University.
‡ The Department of Industrial Engineering, The Pennsylvania State University.
§ The Department of Computer Science and Engineering, The University of Texas at Arlington.

***Abstract*—This paper proposes an algorithm for optimal decentralized traffic engineering in communication networks. We aim at distributing the traffic among the available routes such that the network utility is maximized. In some practical applications, modeling network utility using non-concave functions is of particular interest, e.g., video streaming. Therefore, we tackle the problem of optimizing a generalized class of non-concave utility functions. The approach used to solve the resulting non-convex network utility maximization (NUM) problem relies on designing a sequence of convex relaxations whose solutions converge to that of the original problem. A distributed algorithm is proposed for the solution of the convex relaxation. Each user independently controls its traffic in a way that drives the overall network traffic allocation to an optimal operating point subject to network capacity constraints. All computations required by the algorithm are performed independently and locally at each user using local information and minimal communication overhead. The only non-local information needed is binary feedback from congested links. The robustness of the algorithm is demonstrated, where the traffic is shown to be automatically rerouted in case of a link failure or having new users joining the network. Numerical simulation results are presented to validate our findings.**

***Index Terms*—Distributed optimization, non-concave utility maximization, traffic engineering.**

I. Introduction

Modern communication networks simultaneously support multiple users, services, and applications, each of which requires diverse demands. Therefore, optimum resource allocation among users and/or applications is of paramount importance to assure high quality of service (QoS). Since Kelly et al. introduced the *Network Utility Maximization* (NUM) problem in [1], the NUM framework has found many applications in the development of rate allocation algorithms and internet congestion control protocols.

This paper considers the NUM problem in a connection-oriented network where multiple paths are available for the data of each user. The utility of a user is modeled as a non-concave function and hence, the NUM is a non-convex optimization problem. The objective is to develop a distributed control protocol which steers the traffic away from congested links so that congestion is avoided and network resource utilization is maximized. In particular, the protocol runs independently in parallel at each source node using local information to allow fully distributed traffic control. The only non-local information needed is whether the forwarding path is congested or not, which is binary feedback from link nodes.

Applications including FTP and HTTP used to generate the majority of the internet traffic which is considered elastic traffic. Utility functions for elastic traffic are modeled as strictly concave functions. Resource allocation algorithms for this type of traffic have been well developed, e.g., [2]. However, modern internet flows are dominated by real-time applications, e.g., video and audio streaming, that are considered inelastic. Users' satisfaction for various inelastic applications cannot be accurately modeled using concave functions. For example, the video quality perceived by users on a mobile device is a non-decreasing and step-like function with respect to data rate, because users have almost similar quality of experience on 3 Mbps and 1 Mbps [3]. In addition, the utility for voice applications is better described as a sigmoidal function [4].

Traffic flows with non-concave utilities have received little attention although they represent important application needs in practice. Researchers usually model the user-perceived quality of experience (QoE) as a logarithmic function by adopting the proportional fairness criterion [5]–[7]. The advantage of elasticity assumptions on rate demands is that the resulting optimization problem is tractable, while the disadvantage is that the associated rate allocation may not always favor flows with small buffers [8]. When rate demands are not perfectly elastic, the utility may not be modeled as a concave function. The main challenge that faces resource allocation in networks shared by inelastic applications is that non-convex optimization problems are hard to be analyzed and solved, even by centralized computational methods. The lack of convexity due to the existence of inelastic traffic makes standard distributive algorithms, such as TCP, operate inefficiently [9].

There have been some publications on centralized algorithms [4] and distributed algorithms [10], [11] for non-concave utility maximization. Reference [4] proposes a centralized algorithm based on sum-of-squares (SoS) relaxations and positivstellensatz theorem in real algebraic geometry to calculate approximations of the optimal solution along with some performance bounds to evaluate the approximation error. This efficient but centralized numerical method is suitable for optimizing utilities that can be transformed to polynomial utilities. In [10], the authors propose distributed but suboptimal

This work was partially supported by NSF grants CNS-1329422, CMMI-1635106, FCC-1629625, and the China Scholarship Council.

978-1-5090-5336-0/17/$31.00 ©2017 IEEE

heuristics for sigmoidal utilities. Reference [11] determines the optimality conditions for the canonical distributed algorithm to converge globally for nonlinear utilities. These two approaches illustrate the choice between admission control and capacity planning to deal with non-convexity. However, neither approach provides a theoretically polynomial-time and practically efficient algorithm (centralized or distributed) for non-concave utility maximization. The lack of a comprehensive distributed algorithm that allows network optimization for inelastic applications is the main motivation behind our work.

The contributions of this work are summarized as follows.

- We propose a generic framework for the solution of the NUM problem with non-concave user utility functions. We design a sequence of convex relaxations whose solutions converge to that of the original problem.
- We develop the *distributed traffic allocation algorithm* (DTAA) that allows users to independently adjust their traffic sending rates and/or redistribute traffic load among multiple routes solely based on available local information and binary feedback from the congested link nodes.
- The DTAA is shown to be robust to link failures and it is scalable, where the traffic is automatically rerouted in case of a link failure or when new users join the network.

II. Preliminaries

For the ease of exposition, we briefly recall some mathematical results that play a key role in establishing our findings.

Definition 1. *A relation $R \subset \mathbb{R}^m \times \mathbb{R}^m$ is strongly monotone if there exists $\gamma > 0$ such that*

$$(x_1-x_2)^T(y_1-y_2) \geq \gamma \|x_1-x_2\|^2 \; \forall (x_1,y_1),(x_2,y_2) \in R. \quad (1)$$

Theorem 1. *Let f be a real-valued function, $\mathcal{F}$ be a compact set, not necessarily convex, and μ be a probability measure with support $\operatorname{supp}(\mu)$. Then,*

$$\inf_x \{f(x) : x \in \mathcal{F}\} = \inf_\mu \left\{ \int f d\mu : \operatorname{supp}(\mu) \subset \mathcal{F} \right\}. \quad (2)$$

Proof. The proof can be found in [12]. □

Theorem 1 has been used to convert polynomial optimization problems into a sequence of convex semidefinite programming problems with increasing size via optimizing over moments of probability measures [12]. The problem of moments bridges the gap between the optimization over a space of probability measures whose support is contained in a certain set and the optimization over the moments of such measures. More precisely, given a sequence of scalars $\{t_j\}_{j=1}^{\ell}$, the problem of moments is to determine whether there exists a representing Borel measure that has $\{t_j\}_{j=1}^{\ell}$ as its first ℓ moments. The following theorem provides necessary and sufficient conditions for the existence of Borel measures whose support is included in bounded symmetric intervals of the real line [13].

Theorem 2. *Given a sequence of scalars $\{t_j\}_{j=1}^{\ell}$, there exists a Borel measure $\mu(.)$ with support contained in $\mathcal{I} = [-\epsilon, \epsilon]$ such that $\mu(\mathcal{I}) = 1$ and $t_j = \int_{\mathcal{I}} y^j d\mu$ if and only if*

- *when l is even, the following holds*

$$\mathbf{M}(0,\ell) \succeq 0 \quad (3)$$
$$\epsilon^2 \mathbf{M}(1,\ell-1) \succeq \mathbf{M}(2,\ell), \quad (4)$$

- *when l is odd, the following holds*

$$\epsilon \mathbf{M}(0,\ell-1) \succeq \mathbf{M}(1,\ell) \quad (5)$$
$$\mathbf{M}(1,\ell) \succeq -\epsilon \mathbf{M}(0,\ell-1), \quad (6)$$

where $\mathbf{M}(k,k+2h) \in \mathbb{R}^{(h+1)\times(h+1)}$ is the Hankel matrix

$$\mathbf{M}(k,k+2h) = \begin{bmatrix} t_k & t_{k+1} & \dots & t_{k+h} \\ t_{k+1} & \iddots & \iddots & t_{k+h+1} \\ \vdots & \iddots & \iddots & \vdots \\ t_{k+h} & \dots & \dots & t_{k+2h} \end{bmatrix}, \quad (7)$$

and $t_0 = 1$.

Proof. Theorems III.2.3 and III.2.4 in [14]. □

III. Problem Formulation

This section introduces the notation used throughout the paper. Furthermore, it presents the NUM problem formulation, and highlights the challenges associated with its solution.

A. Notation

Consider a communication network represented by a set $\mathcal{L} = \{1,\dots,L\}$ of directed links with finite capacities $\mathbf{c} = [c_l]_{l\in\mathcal{L}}$, shared by a set of sources $\mathcal{S} = \{1,\dots,N\}$. Each source $i \in \mathcal{S}$ transmits data at rate x_i^p along a predetermined route $p \in \mathcal{P}_i$, where each route $p \subset \mathcal{L}$ is a directed path consisting of a set of links that connect the source to its destination, and $\mathcal{P}_i$ is the set of all routes that can be used simultaneously by source i. Each source $i \in \mathcal{S}$ has a utility function $U_i : \mathbb{R}_+ \to \mathbb{R}_+$. The utility of source i, $U_i(r_i)$, is a function of the aggregate data rate transmitted over all possible routes, where $r_i = \sum_{p\in\mathcal{P}_i} x_i^p$. Let the vector of source i data rates over $\mathcal{P}_i$ be $\mathbf{x}_i = [x_i^p]_{p\in\mathcal{P}_i} \in \mathbb{R}_+^{|\mathcal{P}_i|}$, where $|\mathcal{P}_i|$ denotes the cardinality of $\mathcal{P}_i$, and $\mathbf{r} = [r_i]_{i\in\mathcal{S}} \in \mathbb{R}_+^N$. Define the matrix $\mathbf{A}_i = [a_{l,i}^p]_{l\in\mathcal{L},p\in\mathcal{P}_i} \in \mathbb{R}^{L\times|\mathcal{P}_i|}$ such that its (l,p)th entry $a_{l,i}^p = 1$ if route $p \in \mathcal{P}_i$ uses link $l \in \mathcal{L}$, and is 0 otherwise. Let $\mathcal{P}_i^l = \{p \in \mathcal{P}_i : l \in p\}$ be the set of routes for the data of source i that use link $l \in \mathcal{L}$, and $\mathcal{S}_l = \{i \in \mathcal{S} : \exists p \in \mathcal{P}_i \text{ s.t. } l \in p\}$ be the set of sources using link $l \in \mathcal{L}$.

B. Problem Statement

This work considers network utility functions that can be expressed as a sum of local user utilities, i.e., we maximize $U(\mathbf{r}) = \sum_{i\in\mathcal{S}} U_i(r_i)$ subject to network resource constraints and QoS guarantees. The traffic is allocated so that no single link in the network is congested. A link $l \in \mathcal{L}$ is said to be congested if the sum data rate of all sources using that link exceeds its capacity. The network capacity constraints are $\sum_{i\in\mathcal{S}} \sum_{p\in\mathcal{P}_i^l} x_i^p \leq c_l, \; \forall l \in \mathcal{L}$. Furthermore, minimum QoS guarantees are considered at each source $i \in \mathcal{S}$ in the form of a lower bound on its aggregate data rate. Nevertheless, we take into account the existence of an upper bound on the data rate of each source for practical reasons. Thus, $b_i \leq r_i \leq B_i$ for some $b_i, B_i \geq 0$ and all $i \in \mathcal{S}$. That said, the optimal traffic allocation that maximizes the network utility is obtained by solving the following optimization problem

$$\begin{aligned} \underset{(\mathbf{x}_i, r_i),\ i\in\mathcal{S}}{\text{maximize}} \quad & \sum_{i\in\mathcal{S}} U_i(r_i) \\ \text{subject to} \quad & \sum_{i\in\mathcal{S}} \mathbf{A}_i \mathbf{x}_i \preceq \mathbf{c}, \\ & (\mathbf{x}_i, r_i) \in \mathcal{X}_i, \quad i \in \mathcal{S}, \end{aligned} \tag{8}$$

where $\mathcal{X}_i = \left\{(\mathbf{x}_i, r_i) : r_i = \sum_{p\in\mathcal{P}_i} x_i^p,\ b_i \leq r_i \leq B_i\right\}$.

Most approaches developed in the literature for optimal decentralized traffic allocation allow only for concave diminishing reward utility functions. However, in real-time applications, concave utility functions are not the best measure of user satisfaction. This paper aims at developing a low-complexity distributed algorithm that optimizes a generalized class of non-concave user utility functions. The challenge of solving the optimization problem (8) is two-fold. First, the problem is non-convex since we aim at maximizing a non-concave objective function. Second, global network information is not available; a fact that stimulates the necessity of developing a decentralized algorithm that converges to the solution of (8).

IV. Main Results

We consider a general class of non-decreasing non-concave polynomial-like user utility functions of the form $U_i(r_i) = \sum_{j=0}^{\ell} p_{i,j} r_i^{j/\ell}$ for some $p_{i,j} \in \mathbb{R}$, and some $\ell \in \mathbb{Z}_+$. The motivation behind using this form of utility functions is three-fold: i) the particular form of this function is flexible to the extent that it can be used to approximate a wide variety of utility functions arising in real-world applications, e.g., step-like functions in the case of video streaming; ii) efficient approximation techniques can be implemented to calculate the coefficients $p_{i,j}$, e.g., regression, SoS, and Chebyshev polynomial approximation; iii) it leads to a formulation that can be efficiently solved by decentralized algorithms.

A. NUM Convex Relaxation

We propose a convex relaxation of (8) with polynomial-like utility functions by leveraging results from the moments approach to polynomial optimization. Instead of solving (8), we propose to solve the following semidefinite program:

$$\begin{aligned} \underset{(\mathbf{m}_i, \mathbf{x}_i, r_i),\ i\in\mathcal{S}}{\text{maximize}} \quad & \sum_{i\in\mathcal{S}} \mathbf{p}_i^T \mathbf{m}_i \\ \text{subject to} \quad & m_{i,0} = 1, \quad i \in \mathcal{S} \\ & \mathbf{M}_i(0,\ell) \succeq 0, \quad i \in \mathcal{S} \\ & B_i^2 \mathbf{M}_i(1, \ell-1) \succeq \mathbf{M}_i(2,\ell), \quad i \in \mathcal{S} \\ & m_{i,j} \leq r_i^{j/\ell}, \quad j \in \{1,\dots,\ell\},\ i \in \mathcal{S} \\ & \sum_{i\in\mathcal{S}} \mathbf{A}_i \mathbf{x}_i \preceq \mathbf{c}, \\ & (\mathbf{x}_i, r_i) \in \mathcal{X}_i, \quad i \in \mathcal{S}, \end{aligned} \tag{9}$$

where $\mathbf{p}_i = [p_{i,j}]_{j\in\{0,\dots,\ell\}}$, $\mathbf{m}_i = [m_{i,j}]_{j\in\{0,\dots,\ell\}}$, and $\mathbf{M}_i \in \mathbb{R}^{(h+1)\times(h+1)}$ are Hankel matrices of the form

$$\mathbf{M}_i(k, k+2h) = \begin{bmatrix} m_{i,k} & m_{i,k+1} & \dots & m_{i,k+h} \\ m_{i,k+1} & \iddots & \iddots & m_{i,k+h+1} \\ \vdots & \iddots & \iddots & \vdots \\ m_{i,k+h} & \dots & \dots & m_{i,k+2h} \end{bmatrix}. \tag{10}$$

The following proposition constitutes a main result of this paper; it states that an almost optimal traffic allocation which maximizes the sum of local non-concave user utility functions subject to network capacity constraints and QoS guarantees can be obtained by solving a convex program.

Proposition 1. *The solution of the NUM problem (8) with non-concave polynomial-like user utility functions can be approximated by solving the convex semidefinite program (9).*

Proof. See Appendix A. □

It is worth mentioning that (9) represents a relaxation of (8) when ℓ is even. Nevertheless, a similar result can be obtained when ℓ is odd by slightly modifying the constraints of (9) based on Theorem 2. In (9), a sum of linear functions is maximized subject to convex constraints including linear matrix inequalities, i.e., (9) is a convex optimization problem. Therefore, (9) can be readily solved if global network information is available using an algorithm for solving convex optimization problems, e.g., gradient-based algorithms. Nonetheless, a main objective of this paper is to develop a decentralized traffic allocation algorithm that leverages local information available at each user and minimal network information exchange.

B. Distributed Traffic Allocation Algorithm

In this section, we develop the DTAA, an iterative algorithm that converges to the solution of (9) given the absence of global network information. We move mathematical derivations to Appendix B to enhance the readability of the paper.

Among the advantages of the proposed convex formulation (9) for the NUM problem is that it is amenable to decentralization. By examining (9), we notice the following:

- The variables $\mathbf{m}_i$ and r_i are local to the ith source and need not be broadcasted to any other node.
- The objective function is a sum of local linear functions of the local variables.
- The constraints $\mathbf{M}_i(0,\ell) \succeq 0$, $B_i^2 \mathbf{M}_i(1, \ell-1) \succeq \mathbf{M}_i(2,\ell)$, $m_{i,j} \leq r_i^{j/\ell}$, and $(\mathbf{x}_i, r_i) \in \mathcal{X}_i$ are local constraints to the ith source that can be handled locally.
- The only constraint that forces interaction among the sources is the network capacity constraint.

We introduce some notations that render the formulation of (9) conveniently compact. Let

$$\begin{aligned} \mathcal{K}_i = \Big\{ & (\mathbf{m}_i, \mathbf{x}_i, r_i) : m_{i,0} = 1, m_{i,j} \leq r_i^{j/\ell}, j = 1, \dots, \ell, \\ & \mathbf{M}_i(0,\ell) \succeq 0, B_i^2 \mathbf{M}_i(1, \ell-1) \succeq \mathbf{M}_i(2,\ell), (\mathbf{x}_i, r_i) \in \mathcal{X}_i \Big\} \end{aligned} \tag{11}$$

Algorithm 1: DTAA (ρ,$\{\alpha^n\}_{n\in\mathbb{Z}_+}$,$\{\tau^k\}_{k\in\mathbb{N}}$, $\{\lambda^k\}_{k\in\mathbb{N}}$, $\mathbf{z}^0=[\mathbf{z}_i^0]_{i\in\mathcal{S}}$, $\mathbf{u}^0=[\mathbf{u}_i^0]_{i\in\mathcal{S}}$)

1 Initialize $\mathbf{z}^0$, $\mathbf{u}^0$
2 **for** $k=0,1,\dots$ **do**
3 $\quad (\mathbf{m}_i,\mathbf{x}_i,r_i)^{k+1} \leftarrow \text{argmax}\{\mathbf{p}_i^T\mathbf{m}_i - \frac{\rho}{2}\left\|\mathbf{x}_i-\mathbf{z}_i^k+\mathbf{u}_i^k\right\|^2 : (\mathbf{m}_i,\mathbf{x}_i,r_i)\in\mathcal{K}_i\}, i\in\mathcal{S}.$
4 $\quad$ Initialize $\mathbf{z}^{k,1}\leftarrow\mathbf{z}^k$
5 $\quad$ **for** $n=1,\dots,\tau^k-1$ **do**
6 $\quad\quad$ Each source $i\in\mathcal{S}$ communicates $\mathbf{z}_i^{k,n}$ to $\mathcal{P}_i$.
7 $\quad\quad$ Each link $l\in\mathcal{L}$ sends $b_l^{k,n}$ to $\mathcal{S}_l$.
8 $\quad\quad \mathbf{g}_i^{k,n}\leftarrow\mathbf{z}_i^{k,n}-\mathbf{x}_i^{k+1}-\mathbf{u}_i^k+\lambda^k\sum_{l\in\mathcal{L}}b_l^{k,n}\mathbf{a}_{l,i},\ i\in\mathcal{S}.$
9 $\quad\quad \mathbf{z}_i^{k,n+1}\leftarrow\left(\mathbf{z}_i^{k,n}-\alpha^n\mathbf{g}_i^{k,n}\right)^+,\ i\in\mathcal{S}.$
10 $\quad \mathbf{z}^{k+1}\leftarrow\mathbf{z}^{k,\tau^k}$
11 $\quad \mathbf{u}_i^{k+1}\leftarrow\mathbf{u}_i^k+\mathbf{x}_i^{k+1}-\mathbf{z}_i^{k+1},\ i\in\mathcal{S}.$

and $\mathcal{C}=\left\{[\mathbf{x}_i]_{i\in\mathcal{S}}\in\mathbb{R}_+^{\sum_{i\in\mathcal{S}}|\mathcal{P}_i|}:\sum_{i\in\mathcal{S}}\mathbf{A}_i\mathbf{x}_i\preceq\mathbf{c}\right\}$. Thus, (9) can be compactly stated as follows:

$$\begin{array}{ll}\underset{(\mathbf{m}_i,\mathbf{x}_i,r_i),\ i\in\mathcal{S}}{\text{maximize}} & \sum_{i\in\mathcal{S}}\mathbf{p}_i^T\mathbf{m}_i\\ \text{subject to} & (\mathbf{m}_i,\mathbf{x}_i,r_i)\in\mathcal{K}_i,\quad i\in\mathcal{S}\\ & \mathbf{x}\in\mathcal{C},\end{array}\tag{12}$$

where $\mathbf{x}=[\mathbf{x}_i]_{i\in\mathcal{S}}$. For a reason that will become clear in Appendix B, we introduce a new variable $\mathbf{z}=[\mathbf{z}_i]_{i\in\mathcal{S}}\in\mathbb{R}_+^{\sum_{i\in\mathcal{S}}|\mathcal{P}_i|}$ and obtain the equivalent optimization problem

$$\begin{array}{ll}\underset{(\mathbf{m}_i,\mathbf{x}_i,r_i,\mathbf{z}_i),\ i\in\mathcal{S}}{\text{maximize}} & \sum_{i\in\mathcal{S}}\mathbf{p}_i^T\mathbf{m}_i\\ \text{subject to} & (\mathbf{m}_i,\mathbf{x}_i,r_i)\in\mathcal{K}_i,\quad i\in\mathcal{S}\\ & \mathbf{z}_i=\mathbf{x}_i,\quad i\in\mathcal{S}\\ & \mathbf{z}\in\mathcal{C}.\end{array}\tag{13}$$

Algorithm 1 summarizes the proposed optimal DTAA. The vector $\mathbf{x}_i$ stores the desired transmission rates of source i over $\mathcal{P}_i$. However, source i transmits with an actual rate vector $\mathbf{z}_i$ throughout Algorithm 1's iterations. The constraint $\mathbf{z}_i=\mathbf{x}_i$ is not satisfied for every iteration. Nevertheless, both $\mathbf{z}_i$ and $\mathbf{x}_i$ eventually converge to a consensus as the algorithm keeps running. We proceed with introducing the parameters of Algorithm 1 followed by a description of how it works.

In Algorithm 1, $\rho\in\mathbb{R}_{++}$, and $\{\alpha^n\}_{n\in\mathbb{Z}_+}\subset\mathbb{R}_{++}$ is a diminishing sequence of positive scalars that is not summable but square summable, i.e., $\sum_{n\geq1}\alpha^n=\infty$, and $\sum_{n\geq1}(\alpha^n)^2<\infty$. For instance, $\alpha^n=1/n$. The sequence $\{\tau^k\}_{k\in\mathbb{N}}\subset\mathbb{Z}_+$ is an increasing sequence of positive integers, i.e., $\tau^{k+1}\geq\tau^k$, and $\{\lambda^k\}_{k\in\mathbb{N}}\subset\mathbb{R}_{++}$ is a sequence of positive scalars. The superscripts k and n denote iteration indices. For every outer iteration k, there exist τ^k-1 inner iterations indexed by n. Each source $i\in\mathcal{S}$ keeps the vectors $\mathbf{p}_i$, $\mathbf{m}_i$, $\mathbf{x}_i$, $\mathbf{z}_i$, and $\mathbf{u}_i\in\mathbb{R}^{|\mathcal{P}_i|}$ as private information not shared with any other network entity. Furthermore, source i knows the structure of the matrix $\mathbf{A}_i$ that outlines the links used by its own routes, and need not know any other information about those links. More precisely, source i is oblivious to the capacities of the links used by its routes and does not know whether other sources are sharing those same links with it or not. In addition, the structure of the set $\mathcal{K}_i$ is known for every $i\in\mathcal{S}$, where (11) indicates that $\mathcal{K}_i$ is fully characterized by local information available at source i such as knowing the lower and upper bounds imposed on its own data rate. The kth outer iteration of Algorithm 1 consists of the following:

- In step 3, each source $i\in\mathcal{S}$ updates its desired rates $\mathbf{x}_i$ by solving a simple convex semidefinite program. This step is carried out in parallel locally at each source.
- An inner loop of τ^k-1 iterations is executed in parallel at each source. In the nth inner iteration, each source $i\in\mathcal{S}$ transmits data over its routes $\mathcal{P}_i$ at rates $\mathbf{z}_i^{k,n}$. Then, link nodes that is congested sends $b_l^{k,n}$ to the set of sources using that link, i.e., $\mathcal{S}_l$. The binary feedback bit $b_l^{k,n}$ determines the status of link $l\in\mathcal{L}$, where $b_l^{k,n}$ equals 1 if link l is congested, and is 0 otherwise. This feedback information is used by each source $i\in\mathcal{S}$ to update its actual data rate vector $\mathbf{z}_i$ as in step 9.
- Each source $i\in\mathcal{S}$ updates the vector $\mathbf{u}_i$ locally as in step 11. This step is carried out in parallel and independently across all sources.

Indeed, Algorithm 1 provides fully distributed optimal traffic allocation, where all the computations are performed in parallel independently at each source node and need not be broadcasted. Furthermore, the variable updates done locally at each source node use local information and the only non-local information needed is binary feedback from congested links; hence, the network communication overhead is low.

V. Numerical Simulations

This section presents an example application of the DTAA developed in this paper. Numerical simulations of the proposed algorithm are conducted to validate our findings. In particular, the main objectives of this section are summarized as follows.

- We numerically show that the global optimal solution of the non-convex NUM problem (8) can be approximately obtained via solving the proposed convex relaxation (9).
- We show that the DTAA presented in Algorithm 1 converges to a solution of (9).
- We demonstrate the robustness of the DTAA to link failures, and show that it automatically scales out to accommodate new users joining the network.

A. Network Topology

We adopt the network model shown in Fig. 1, which is based on the one considered in [15]. The network model allows for multiple routes to be available for the data of each source. Fig. 1 shows the topology of the network considered as well as the capacity of each link. We consider a total of $N=8$ different combination of source/destination nodes, where the intended destination of the data sent by source S_i is D_i, $i=1,\dots,N$. The routes available for the data of each source are described in Table I.

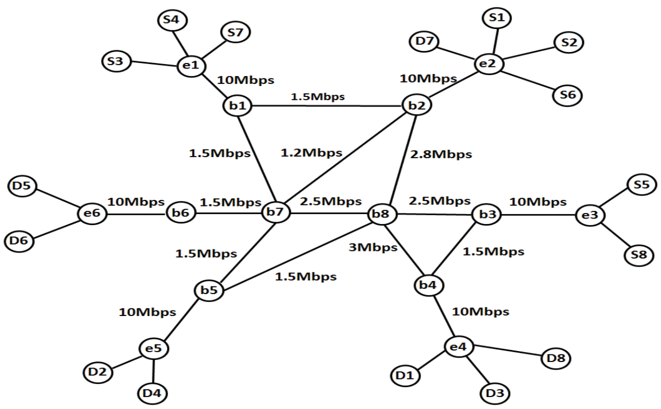

Fig. 1. The topology of the network.

TABLE I
ROUTES AVAILABLE FOR THE DATA OF EACH SOURCE

S_1 $\lvert\mathcal{P}_1\rvert=4$	x_1^1 $e_2b_2b_8b_4e_4$ x_1^2 $e_2b_2b_8b_3b_4e_4$ x_1^3 $e_2b_2b_7b_8b_3b_4e_4$ x_1^4 $e_2b_2b_8b_4e_4$	S_5 $\lvert\mathcal{P}_5\rvert=2$	x_5^1 $e_3b_3b_8b_7b_6e_6$ x_5^2 $e_3b_3b_4b_8b_5b_7b_6e_6$
S_2 $\lvert\mathcal{P}_2\rvert=3$	x_2^1 $e_2b_2b_8b_5e_5$ x_2^2 $e_2b_2b_7b_5e_5$ x_2^3 $e_2b_2b_1b_7b_5e_5$	S_6 $\lvert\mathcal{P}_6\rvert=3$	x_6^1 $e_2b_2b_1b_7b_6e_6$ x_6^2 $e_2b_2b_8b_7b_6e_6$ x_6^3 $e_2b_2b_7b_6e_6$
S_3 $\lvert\mathcal{P}_3\rvert=2$	x_3^1 $e_1b_1b_7b_8b_4e_4$ x_3^2 $e_1b_1b_2b_8b_4e_4$	S_7 $\lvert\mathcal{P}_7\rvert=3$	x_7^1 $e_1b_1b_2e_2$ x_7^2 $e_1b_1b_7b_2e_2$ x_7^3 $e_1b_1b_7b_8b_2e_2$
S_4 $\lvert\mathcal{P}_4\rvert=4$	x_4^1 $e_1b_1b_7b_5e_5$ x_4^2 $e_1b_1b_7b_8b_5e_5$ x_4^3 $e_1b_1b_2b_7b_5e_5$ x_4^4 $e_1b_1b_2b_8b_5e_5$	S_8 $\lvert\mathcal{P}_8\rvert=2$	x_8^1 $e_3b_3b_4e_4$ x_8^2 $e_3b_3b_8b_4e_4$

B. Validation of the Main Results

We propose to optimize a step-like non-decreasing utility function. As suggested by [3], step-like functions are more likely to express the video quality perceived by a user in video streaming applications. For that reason, we show our simulation results for utility functions given by

$$U_i(r_i) = \begin{cases} 0, & \text{if } 0 \le r_i < 1 \\ 1, & \text{if } 1 \le r_i < 2 \,, \quad i \in \mathcal{S}. \\ 2, & \text{if } 2 \le r_i \le 3 \end{cases} \tag{14}$$

The optimal traffic allocation is obtained through solving (8), where the matrices $\mathbf{A}_i$, $i \in \mathcal{S}$, and the vector $\mathbf{c}$ are constructed using the information shown in Fig. 1 and Table I. The lower and upper bounds imposed on the aggregate data rate of each user are $b_i = 0$ and $B_i = 10$, respectively, for all $i \in \mathcal{S}$.

Obviously, the utility functions (14) are not in polynomial-like form. Nevertheless, we approximate these utilities by polynomial-like functions, i.e., we obtain the coefficient vectors $\mathbf{p}_i$ that render the polynomial-like functions close enough to (14) according to some defined metric, where we choose to show results for $\ell = 6$. We refrain from detailing the approximation technique used for that purpose since it is not the main focus of this paper and due to space limitations.

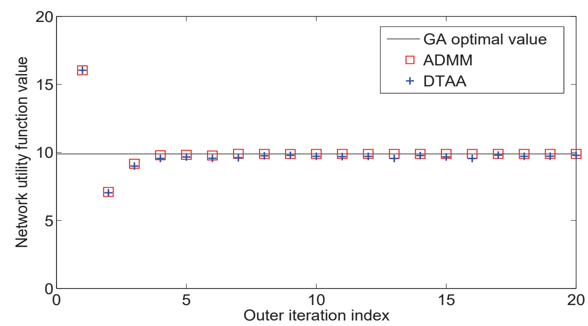

Fig. 2. Network utility function.

The non-convex NUM problem (8), with the polynomial-like approximation of (14), is solved using the genetic algorithm (GA) while assuming the availability of global network information. The traffic allocation obtained through the GA serves as a benchmark with which we compare the performance of the proposed DTAA. However, we emphasize that the GA is a centralized solution that is prohibitively expensive to be implemented in practice. All simulation results are shown for the following parameter choices: $\rho = 1$, $\lambda^k = 10\ \forall k \in \mathbb{N}$, $\tau^k = 10^3\ \forall k \in \mathbb{N}$, and $\alpha^n = 1/n$. In Fig. 2, we compare the performance of the DTAA to a centralized algorithm that solves the NUM problem based on an exact version of the alternating direction method of multipliers (ADMM) as well as the GA. The exact ADMM algorithm is presented in Appendix B-B. Fig. 2 shows that the data rate allocation obtained by the proposed DTAA results in a utility function value that is barely indistinguishable from the optimal one obtained through solving the non-convex NUM problem with the GA. Fig. 2 also shows that the performance of the DTAA summarized in Algorithm 1 closely follows that of the exact ADMM. Although the proposed algorithm is implemented in a distributed manner that requires no global network information, it attains almost the same network utility obtained by a centralized traffic allocation algorithm. Thus, Fig. 2 validates the soundness of the proposed convex relaxation of the NUM problem, and shows the convergence of the proposed DTAA to the optimal traffic allocation.

For the clarity of exposition, we choose to show the results for the data rate allocation on the routes available to the data of sources S_5 and S_8 in Fig. 3 and 4, respectively, since the set of available routes to each of them has a cardinality equal to 2, i.e., $|\mathcal{P}_5| = |\mathcal{P}_8| = 2$. The DTAA is shown to converge to the optimal data rate allocation obtained by the centralized exact ADMM.

C. Algorithm Robustness

Finally, we show that the proposed DTAA is robust to link failures and it automatically scales out to accommodate new users joining the network. This feature is attributed to the adaptive nature of updating the data rates. When a link failure is detected, the algorithm reroutes the traffic such that it avoids the routes using that link. The DTAA handles a link failure by

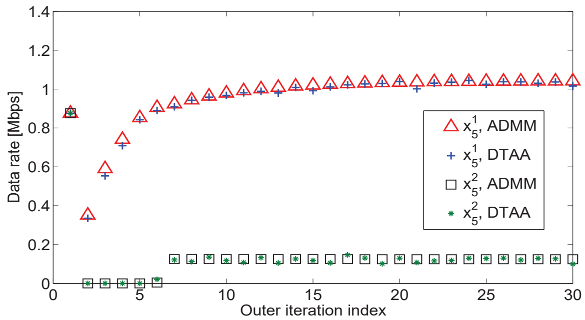

Fig. 3. Source S_5 data rate allocation.

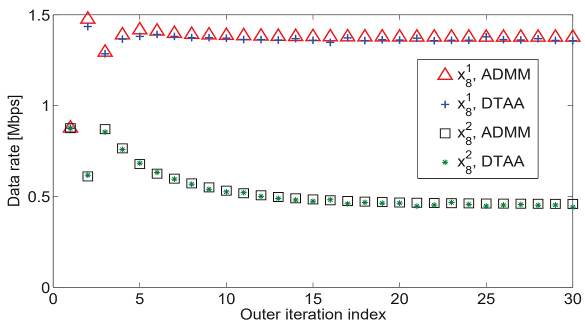

Fig. 4. Source S_8 data rate allocation.

treating it as congestion, i.e., the source nodes are oblivious to the failure and require no additional information other than the usual binary feedback. The robustness of the algorithm is demonstrated by Fig. 5, where the link connecting the nodes b_4-e_4 fails after 20 iterations of running the algorithm and recovers after 50 iterations. This particular link is chosen to fail since its failure implies that sources S_1, S_3, and S_8 are disconnected from their destinations. Thus, its failure results in a considerable degradation to the network utility, and its recovery enables us to check if the algorithm is capable of accommodating new users joining the network. The figure shows that the algorithm quickly reacts to both the failure and recovery of the link, where the utility function is shown to converge to its new optimal value. It is worth mentioning that the optimal value lines shown in this figure are obtained by a centralized solution to (9) with the assumption of the availability of global network information and the knowledge of which link failed and when it fails and recovers.

VI. Conclusion

This paper addresses the optimization of network utility functions that can be expressed as a sum of local user utilities. The utility of each user is a non-concave function of its aggregate data rate. In many practical applications, a non-concave utility function is a better model of the user-perceived quality. A convex relaxation of the non-convex NUM problem has been proposed. Furthermore, an optimal decentralized traffic allocation algorithm has been developed. All computations are performed in parallel locally at each user.

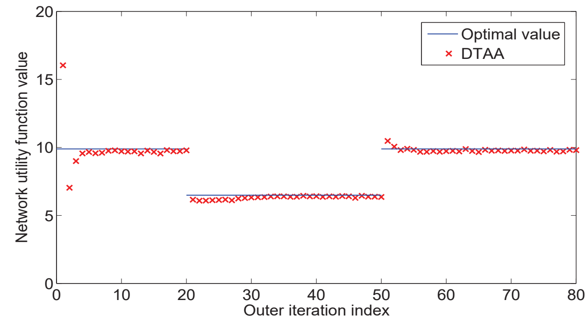

Fig. 5. Utility function value in response to link failure and recovery.

The information exchange in the network is minimal, where a binary link congestion notification bit is fed back to the source nodes. Numerical simulations demonstrated the robustness of the algorithm to sudden link failures. Moreover, the algorithm is shown to scale out automatically to accommodate new users joining the network.

Future directions include performing numerical simulation of the DTAA on large scale networks to further assess the scalability of the algorithm. Moreover, we envision developing a decentralized rate allocation algorithm that allows each node to adapt its rate among any given set of next hops solely based on immediate information from neighboring nodes.

Appendix A
Proof of Proposition 1

The NUM problem (8) with polynomial-like utility functions is stated as

$$\begin{aligned} \underset{(\mathbf{x}_i, r_i),\ i\in\mathcal{S}}{\text{maximize}} \quad & \sum_{i\in\mathcal{S}} \sum_{j=0}^{\ell} p_{i,j} r_i^{j/\ell} \\ \text{subject to} \quad & \sum_{i\in\mathcal{S}} \mathbf{A}_i \mathbf{x}_i \preceq \mathbf{c}, \\ & (\mathbf{x}_i, r_i) \in \mathcal{X}_i, \quad i \in \mathcal{S}, \end{aligned} \tag{15}$$

We note that the objective function of (15) is in polynomial form if one does a change of variables $y_i = r_i^{1/\ell}$; consequently, an equivalent formulation of (15) is

$$\begin{aligned} \underset{(\mathbf{x}_i, r_i, y_i),\ i\in\mathcal{S}}{\text{maximize}} \quad & \sum_{i\in\mathcal{S}} \sum_{j=0}^{\ell} p_{i,j} y_i^{j} \\ \text{subject to} \quad & 0 \le y_i \le r_i^{1/\ell}, \quad i \in \mathcal{S} \\ & \sum_{i\in\mathcal{S}} \mathbf{A}_i \mathbf{x}_i \preceq \mathbf{c}, \\ & (\mathbf{x}_i, r_i) \in \mathcal{X}_i, \quad i \in \mathcal{S}. \end{aligned} \tag{16}$$

This equivalent formulation is still a non-convex problem due to the non-concavity of the objective function. Nevertheless, the convexity of the feasible set is preserved. Indeed, $r_i^{1/\ell}$ is a concave function for $\ell \in \mathbb{Z}_+$; hence, the constraints $y_i \le r_i^{1/\ell}$ are convex constraints. Inspired by the results of Theorems 1 and 2, we transform (16) into an optimization problem over the space of probability measures of y_i whose support is contained in the feasible set of (16). More precisely, we denote by $m_{i,j}$ the moment of order j of y_i for some probability measure μ_i, i.e., $m_{i,j} = \int y_i^j d\mu_i$. Theorem 1 implies that the objective function becomes

$$\int \sum_{i\in\mathcal{S}} \sum_{j=0}^{\ell} p_{i,j} y_i^j d\mu_i = \sum_{i\in\mathcal{S}} \mathbf{p}_i^T \mathbf{m}_i. \tag{17}$$

Then, the result of Theorem 2 is used to construct the first 3 constraints of (9). It now remains to handle the constraint $y_i \le r_i^{1/\ell}$, i.e., represent it in terms of the moments of y_i. This constraint is approximated by the set of constraints

$$m_{i,j} \le r_i^{j/\ell},\ j \in \{1, \dots, \ell\}. \tag{18}$$

It is worth mentioning that $r_i^{j/\ell}$ is a concave function for all $j \in \{1, \dots, \ell\}$; hence, (18) is a set of convex constraints. As ℓ increases, this constraint approximation is enhanced and thus, the solution of (9) approaches that of (8).

Appendix B
Derivation of the Traffic Allocation Algorithm

This section presents the detailed mathematical derivation of the DTAA developed in this paper. In other words, we derive a decentralized algorithm that solves (13). Furthermore, we provide a concise convergence proof of the algorithm for the sake of completeness.

A. The Method of Multipliers

Under some mild assumptions, such as the existence of a strictly feasible point for the convex program (13), strong duality holds. Thus, instead of solving (13), the method of multipliers suggests solving the dual problem using a gradient-based algorithm. Towards this objective, we proceed with the derivation of the dual problem corresponding to (13).

The Lagrangian function of (13), augmented with a quadratic penalty on violating the constraint $\mathbf{z} = \mathbf{x}$, is given by

$$\mathrm{L}(\mathbf{m}, \mathbf{x}, \mathbf{z}, \nu) = \sum_{i\in\mathcal{S}} \left[\mathbf{p}_i^T \mathbf{m}_i - \nu_i^T(\mathbf{x}_i - \mathbf{z}_i) - (\rho/2)\|\mathbf{x}_i - \mathbf{z}_i\|^2\right], \tag{19}$$

where $\mathbf{m} = [\mathbf{m}_i]_{i\in\mathcal{S}}$, $\nu = [\nu_i]_{i\in\mathcal{S}}$ is the vector of dual variables associated with the constraint $\mathbf{z} = \mathbf{x}$, and $\rho > 0$ is the penalty parameter. The dual function is then obtained via maximizing the Lagrangian function over all feasible primal variables $(\mathbf{m}, \mathbf{x}, \mathbf{r}, \mathbf{z})$, i.e.,

$$g(\nu) = \max_{(\mathbf{m},\mathbf{x},\mathbf{r},\mathbf{z})} \left\{\mathrm{L}(\mathbf{m}, \mathbf{x}, \mathbf{z}, \nu) : (\mathbf{m}_i, \mathbf{x}_i, r_i) \in \mathcal{K}_i, i \in \mathcal{S}, \mathbf{z} \in \mathcal{C}\right\}. \tag{20}$$

Next, we minimize the dual function, i.e.,

$$\underset{\nu}{\text{minimize}} \quad g(\nu). \tag{21}$$

The dual problem (21) is solved using a gradient descent method with a constant step size ρ. In particular, the gradient descent algorithm produces a sequence $\{\nu^k\}_{k\in\mathbb{N}}$ as follows:

$$\nu^{k+1} = \nu^k - \rho \nabla g(\nu^k), \tag{22}$$

such that $g(\nu^k) \to p^*$ as $k \to \infty$, and p^* is the optimal value of (13). For any given ν, let $(\mathbf{m}(\nu), \mathbf{x}(\nu), \mathbf{r}(\nu), \mathbf{z}(\nu))$ denote the maximizer of (20). According to this construction, the gradient of the dual function is given by

$$\nabla g(\nu) = -(\mathbf{x}(\nu) - \mathbf{z}(\nu)). \tag{23}$$

The method of multipliers can then be summarized as follows:

$$(\mathbf{m}, \mathbf{x}, \mathbf{r}, \mathbf{z})^{k+1} = \underset{(\mathbf{m},\mathbf{x},\mathbf{r},\mathbf{z})}{\text{argmax}} \Big\{ \mathrm{L}(\mathbf{m}, \mathbf{x}, \mathbf{z}, \nu^k) : \mathbf{z} \in \mathcal{C}, (\mathbf{m}_i, \mathbf{x}_i, r_i) \in \mathcal{K}_i, i \in \mathcal{S} \Big\} \tag{24}$$

$$\nu^{k+1} = \nu^k + \rho(\mathbf{x}^{k+1} - \mathbf{z}^{k+1}). \tag{25}$$

Although the convergence to the optimal traffic allocation is guaranteed through the method of multipliers, it does not count as an algorithm that can be implemented in a decentralized fashion. In fact, solving the optimization problem involved in the primal variables update rule (24) requires global information about the network due to the presence of the network capacity constraint $\mathbf{z} \in \mathcal{C}$. Furthermore, we note that the Lagrangian function is maximized jointly over the primal variables; a fact that exacerbates the difficulty of coming up with a distributed implementation of (24).

B. The Alternating Direction Method of Multipliers

Unlike the method of multipliers, ADMM updates the primal variables sequentially rather than jointly; hence, ADMM allows for decomposition when the objective function is separable. In particular, we consider two ADMM blocks, namely, $(\mathbf{m}, \mathbf{x}, \mathbf{r})$ and $\mathbf{z}$. Then, the ADMM algorithm [16] updates the primal and dual variables according to the following rules:

$$(\mathbf{m}, \mathbf{x}, \mathbf{r})^{k+1} = \underset{(\mathbf{m},\mathbf{x},\mathbf{r})}{\text{argmax}} \left\{\mathrm{L}(\mathbf{m}, \mathbf{x}, \mathbf{z}^k, \nu^k) : (\mathbf{m}_i, \mathbf{x}_i, r_i) \in \mathcal{K}_i, i \in \mathcal{S}\right\} \tag{26}$$

$$\mathbf{z}^{k+1} = \underset{\mathbf{z}}{\text{argmax}} \left\{\mathrm{L}(\mathbf{m}^{k+1}, \mathbf{x}^{k+1}, \mathbf{z}, \nu^k) : \mathbf{z} \in \mathcal{C}\right\} \tag{27}$$

$$\nu^{k+1} = \nu^k + \rho(\mathbf{x}^{k+1} - \mathbf{z}^{k+1}). \tag{28}$$

Separating the primal variables into the aforementioned two blocks and updating them sequentially brings us one step closer towards a decentralized implementation-friendly algorithm that solves (13). To show that, we exploit the separable structure of the Lagrangian function (19) to decompose the optimization problem involved in the update rule (26) into N independent optimization problems that can be solved in parallel locally at each source node. In particular, simple algebraic manipulation on (19) and its substitution into (26) and (27) render ADMM consisting of the following iterations:

$$(\mathbf{m}_i, \mathbf{x}_i, r_i)^{k+1} = \underset{(\mathbf{m}_i,\mathbf{x}_i,r_i)\in\mathcal{K}_i}{\text{argmax}} \left\{\mathbf{p}_i^T \mathbf{m}_i - (\rho/2)\left\|\mathbf{x}_i - \mathbf{z}_i^k + \mathbf{u}_i^k\right\|^2\right\} \tag{29}$$

$$\mathbf{z}^{k+1} = \Pi_{\mathcal{C}}\left\{\mathbf{x}^{k+1} + \mathbf{u}^k\right\} \tag{30}$$

$$\mathbf{u}_i^{k+1} = \mathbf{u}_i^k + \mathbf{x}_i^{k+1} - \mathbf{z}_i^{k+1} \tag{31}$$

where the variable update rules in (29) and (31) are performed in parallel for every $i \in \mathcal{S}$, $\mathbf{u}$ is a scaled version of the dual variables ν such that $\mathbf{u}_i = (1/\rho)\nu_i$, $\mathbf{u} = [\mathbf{u}_i]_{i\in\mathcal{S}}$, and $\Pi_{\mathcal{C}}\{.\}$ denotes the Euclidean projection operator over the set $\mathcal{C}$.

In pursuit of solving the NUM problem (13) in a distributed manner, we examine the update rules (29)-(31) to check the possibility of their decentralized implementation. By examining (29)-(31), we note the following:

- The variables $\mathbf{m}_i$, $\mathbf{x}_i$, $\mathbf{r}_i$, $\mathbf{z}_i$, and $\mathbf{u}_i$ are considered local variables to the ith source.
- The update rules (29) and (31) can be performed in parallel independently at each source node using local information only. Moreover, it is assumed that each source node has the computational capabilities that enable it to solve a simple convex semidefinite program as the one in (29).
- The $\mathbf{z}$-update step (30) requires knowing the values of the transmission rates $\mathbf{x}_i$ and the dual variables $\mathbf{u}_i$ for each source node $i \in \mathcal{S}$. Furthermore, it requires global information of the network represented by the requirement of knowing the structure of the set $\mathcal{C}$, specifically, it requires the knowledge of the matrices $\mathbf{A}_i$ for each $i \in \mathcal{S}$ and the capacity of every link $l \in \mathcal{L}$, i.e., $\mathbf{c}$.

Although ADMM allows the decomposition of (26) into independent optimization problems (29) solved locally at each source node, its implementation requires the presence of a central entity to perform the $\mathbf{z}$-update step (30). A two-way communication occurs between each source node and this central entity. In the first phase, each source $i \in \mathcal{S}$ sends the values of $\mathbf{x}_i$ and $\mathbf{u}_i$. After the reception of the information sent by all sources, the central entity performs the projection operation (30) and sends back the updated values of $\mathbf{z}_i$ to all sources $i \in \mathcal{S}$. Thus, it is obvious that a direct implementation of ADMM exemplifies a centralized solution to the NUM problem with considerable communication overhead. Next, we propose an inexact ADMM algorithm that can be implemented in a decentralized fashion, i.e., it resolves the problem of requiring a centralized solution for (30).

C. The Inexact Alternating Direction Method of Multipliers

The ADMM algorithm updates the primal variables through solving an optimization problem per ADMM block. Indeed, $(\mathbf{m}, \mathbf{x}, \mathbf{r})$ and $\mathbf{z}$ are updated by solving the maximization problems (26) and (27), respectively. Eckstein shows in [17] that it is possible to obtain a variant of the ADMM algorithm in which at least one of the optimization problems involved in the update rules of ADMM blocks merits an iterative solution. In other words, under some conditions, an approximate inexact solution of any of the optimization problems in the update rules of ADMM suffices to retain the overall convergence of the algorithm. Inspired by this insight, we propose an iterative solution to (30) that opens room for a distributed implementation of the optimal traffic allocation algorithm. More precisely, the proposed iterative solution to (30) requires neither global information of the network nor high communication overhead among source nodes.

The projection operation in (30) entails solving the following quadratic program

$$\begin{aligned} \underset{\mathbf{z}_i,\ i\in\mathcal{S}}{\text{minimize}} \quad & \frac{1}{2}\sum_{i\in\mathcal{S}} \left\|\mathbf{z}_i - \left(\mathbf{x}_i^{k+1} + \mathbf{u}_i^k\right)\right\|^2 \\ \text{subject to} \quad & \sum_{i\in\mathcal{S}} \mathbf{a}_{l,i}^T \mathbf{z}_i - c_l \le 0,\ l \in \mathcal{L} \\ & \mathbf{z}_i \succeq 0,\ i \in \mathcal{S}, \end{aligned} \tag{32}$$

where $\mathbf{a}_{l,i}^T$ denotes the lth row of the matrix $\mathbf{A}_i$. We propose an alternative formulation of (32) that imposes a penalty on violating the capacity constraint of any link $l \in \mathcal{L}$. More precisely, for each iteration k, instead of updating $\mathbf{z}$ through solving (32), we solve

$$\begin{aligned} \underset{\mathbf{z}}{\text{minimize}} \quad & f^k(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{z} \succeq 0, \end{aligned} \tag{33}$$

where the minimand of (33) is defined as

$$f^k(\mathbf{z}) = \frac{1}{2}\sum_{i\in\mathcal{S}} \left\|\mathbf{z}_i - \mathbf{x}_i^{k+1} - \mathbf{u}_i^k\right\|^2 + \lambda^k \sum_{l\in\mathcal{L}} \left(\sum_{i\in\mathcal{S}} \mathbf{a}_{l,i}^T \mathbf{z}_i - c_l\right)^+ \tag{34}$$

and $(.)^+ = \max(., 0)$. Problems (32) and (33) are equivalent for a high enough value of $\lambda^k \in \mathbb{R}_{++}$. Theoretically, it is not easy to choose λ^k that guarantees the equivalence of (32) and (33). Thus, we envision investigating this technical issue in a future work. For every ADMM iteration k, an inner iterative subgradient descent algorithm is employed to solve (33) through generating a sequence $\{\mathbf{z}^{k,n}\}_{n\in\mathbb{Z}_+}$ such that

$$\mathbf{z}_i^{k,n+1} = \left(\mathbf{z}_i^{k,n} - \alpha^n \mathbf{g}_i^{k,n}\right)^+,\ i \in \mathcal{S}, \tag{35}$$

with an appropriately chosen diminishing step size sequence $\{\alpha^n\}_{n\in\mathbb{Z}_+}$, and $\mathbf{g}^{k,n} = [\mathbf{g}_i^{k,n}]_{i\in\mathcal{S}} \in \partial f^k(\mathbf{z}^{k,n})$. It follows from (34) that the pth entry of $\mathbf{g}_i^{k,n}$ can be chosen as follows:

$$g_i^{p,k,n} \in \left\{z_i^{p,k,n} - x_i^{p,k+1} - u_i^{p,k}\right\} + \sum_{l\in\mathcal{L}} \beta_l^{k,n},\quad p = 1,\ldots,|\mathcal{P}_i|, \tag{36}$$

with the set $\beta_l^{k,n}$ defined as

$$\beta_l^{k,n} = \begin{cases} \{0\}, & \text{if } \sum_{i\in\mathcal{S}} \mathbf{a}_{l,i}^T \mathbf{z}_i^{k,n} < c_l \\ [0, \lambda^k a_{l,i}^p], & \text{if } \sum_{i\in\mathcal{S}} \mathbf{a}_{l,i}^T \mathbf{z}_i^{k,n} = c_l \\ \{\lambda^k a_{l,i}^p\}, & \text{if } \sum_{i\in\mathcal{S}} \mathbf{a}_{l,i}^T \mathbf{z}_i^{k,n} > c_l, \end{cases} \tag{37}$$

where $z_i^{p,k,n}$, $x_i^{p,k+1}$, and $u_i^{p,k}$ denote the pth entry of $\mathbf{z}_i^{k,n}$, $\mathbf{x}_i^{k+1}$, and $\mathbf{u}_i^k$, respectively. For instance, an obvious construction of $\mathbf{g}_i^{k,n}$ is given by

$$\mathbf{g}_i^{k,n} = \mathbf{z}_i^{k,n} - \mathbf{x}_i^{k+1} - \mathbf{u}_i^k + \lambda^k \sum_{l\in\mathcal{L}} b_l^{k,n} \mathbf{a}_{l,i}, \tag{38}$$

where $b_l^{k,n} \in \{0, 1\}$ is a binary bit that equals 1 if $\mathbf{z}^{k,n}$ causes link l to be congested, and is 0 otherwise.

Lemma 1. *The sequence $\{\mathbf{z}^{k,n}\}$ generated by (35) converges to the optimal solution of (33) $\mathbf{z}^{k,*}$, i.e., $\mathbf{z}^{k,n} \rightarrow \mathbf{z}^{k,*}$ as $n \rightarrow \infty$.*

Algorithm 2: Approximate ADMM $(\rho,\{\alpha^n\}_{n\in\mathbb{Z}_+},\{\epsilon^k\}_{k\in\mathbb{Z}_+},\{\lambda^k\}_{k\in\mathbb{N}},\mathbf{z}^0=[\mathbf{z}_i^0]_{i\in\mathcal{S}},\mathbf{u}^0=[\mathbf{u}_i^0]_{i\in\mathcal{S}})$

Initialize $\mathbf{z}^0$, $\mathbf{u}^0$
repeat{for $k=0,1,\ldots$}
  $(\mathbf{m}_i,\mathbf{x}_i,r_i)^{k+1} \leftarrow \text{argmax}\{\mathbf{p}_i^T\mathbf{m}_i - \frac{\rho}{2}\left\|\mathbf{x}_i - \mathbf{z}_i^k + \mathbf{u}_i^k\right\|^2 : (\mathbf{m}_i,\mathbf{x}_i,r_i)\in\mathcal{K}_i\}, i\in\mathcal{S}.$
  Initialize $\mathbf{z}^{k,1}\leftarrow\mathbf{z}^k$
  repeat{for $n=1,2,\ldots$}
    Each source $i\in\mathcal{S}$ communicates $\mathbf{z}_i^{k,n}$ to $\mathcal{P}_i$.
    Each link $l\in\mathcal{L}$ sends $b_l^{k,n}$ to $\mathcal{S}_l$.
    $\mathbf{g}_i^{k,n}\leftarrow\mathbf{z}_i^{k,n}-\mathbf{x}_i^{k+1}-\mathbf{u}_i^k+\lambda^k\sum_{l\in\mathcal{L}}b_l^{k,n}\mathbf{a}_{l,i},\ i\in\mathcal{S}.$
    $\mathbf{z}_i^{k,n+1}\leftarrow\left(\mathbf{z}_i^{k,n}-\alpha^n\mathbf{g}_i^{k,n}\right)^+,\ i\in\mathcal{S}.$
  until $\|\mathbf{g}^{k,n}\|\le\epsilon^{k+1}$
  $\mathbf{z}^{k+1}\leftarrow\mathbf{z}^{k,n}$
  $\mathbf{u}_i^{k+1}\leftarrow\mathbf{u}_i^k+\mathbf{x}_i^{k+1}-\mathbf{z}_i^{k+1},\ i\in\mathcal{S}.$
until *Overall convergence*

Proof. This result follows from the convergence of subgradient methods; see for instance [18]. □

Let $\{\epsilon^k\}_{k\in\mathbb{Z}_+}\subset\mathbb{R}_{++}$ be a summable sequence of positive scalars, i.e., $\sum_{k\ge1}\epsilon^k<\infty$. Then, Algorithm 2 provides an optimal decentralized traffic allocation algorithm.

Proposition 2. *The sequence* $\left\{(\mathbf{m}_i,\mathbf{x}_i,r_i,\mathbf{z}_i,\rho\mathbf{u}_i)^k\right\}$ *generated by Algorithm 2 converges to a KKT point of (13).*

Proof. The proof of this form of approximate version of ADMM is presented in [17]. Indeed, Algorithm 2 satisfy the conditions required for [17, Proposition 7] to hold. We briefly mention the main idea of the proof for the convenience of the reader. For each iteration k, $f^k(\mathbf{z})$ defined in (34) is a strongly convex function of $\mathbf{z}$ with modulus 1. Thus, the subdifferential map ∂f^k is strongly monotone with modulus $\gamma=1$. We have $0\in\partial f^k(\mathbf{z}^{k,*})$ since $\mathbf{z}^{k,*}$ is the optimal solution of (33), and $\mathbf{g}^{k,n}\in\partial f^k(\mathbf{z}^{k,n})$ by construction. Then, the Cauchy-Schwarz inequality and the strong monotonicity of ∂f^k imply that

$$\|\mathbf{g}^{k,n}\|\|\mathbf{z}^{k,n}-\mathbf{z}^{k,*}\|\ge(\mathbf{z}^{k,n}-\mathbf{z}^{k,*})^T\mathbf{g}^{k,n}\ge\|\mathbf{z}^{k,n}-\mathbf{z}^{k,*}\|^2. \tag{39}$$

Therefore, the termination criterion of the inner loop of Algorithm 2 implies that the distance between the approximate solution of (33) $\mathbf{z}^{k+1}$ and the exact minimizer $\mathbf{z}^{k,*}$ is upper bounded by a summable sequence, i.e.,

$$\|\mathbf{z}^{k+1}-\mathbf{z}^{k,*}\|\le\epsilon^{k+1}, \tag{40}$$

and hence, Proposition 2 follows from [19, Theorem 8]. □

The proposed inexact ADMM algorithm overcomes the two main obstacles encountered by the exact ADMM algorithm; specifically, the need for a centralized solution to (30) and the high communication overhead. A distributed implementation of the traffic allocation algorithm is now possible since the variables $\mathbf{z}_i$ are updated in parallel locally at each source $i\in\mathcal{S}$. In a real-world implementation, Algorithm 2 runs continuously without being terminated. Therefore, Algorithm 1 represents the continuously running version of Algorithm 2. In Algorithm 1, the termination criterion of the inner loop of Algorithm 2 is replaced with executing the inner loop of Algorithm 1 for an increasing number of steps as k increases and hence, ensuring that the approximate solution obtained for (33) is enhanced as k increases.

References

[1] F. P. Kelly, A. K. Maulloo, and D. K. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research society*, vol. 49, no. 3, pp. 237–252, 1998.

[2] A. Beck, A. Nedic, A. Ozdaglar, and M. Teboulle, "Optimal distributed gradient methods for network resource allocation problems." [Online]. Available: http://asu.mit.edu/sites/default/files/documents/publications/NUM-FGM.pdf

[3] X. Q. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," in *SIGCOMM*, 2015, pp. 325–338.

[4] M. Fazel and M. Chiang, "Network utility maximization with nonconcave utilities using sum-of-squares method," in *Proceedings of the 44th IEEE Conference on Decision and Control*, 2005, pp. 1867–1874.

[5] J. Chen, M. Chiang, J. Erman, G. Li, K. Ramakrishnan, and R. K. Sinha, "Fair and optimal resource allocation for LTE multicast (eMBMS): Group partitioning and dynamics," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 1266–1274.

[6] H. Zhou, Y. Ji, X. Wang, and B. Zhao, "ADMM based algorithm for eICIC configuration in heterogeneous cellular networks," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 343–351.

[7] J. Liu, A. Eryilmaz, N. B. Shroff, and E. S. Bentley, "Heavy-ball: A new approach to tame delay and convergence in wireless network optimization," in *2016 IEEE Conference on Computer Communications (INFOCOM)*, pp. 1–9.

[8] F. Kelly, G. Raina, and T. Voice, "Stability and fairness of explicit congestion control with small buffers," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 3, pp. 51–62, 2008.

[9] G. Tychogiorgos, A. Gkelias, and K. K. Leung, "A non-convex distributed optimization framework and its application to wireless ad-hoc networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4286–4296, 2013.

[10] J.-W. Lee, R. R. Mazumdar, and N. B. Shroff, "Non-convex optimization and rate control for multi-class services in the internet," *IEEE/ACM transactions on networking*, vol. 13, no. 4, pp. 827–840, 2005.

[11] P. Hande, S. Zhang, and M. Chiang, "Distributed rate allocation for inelastic flows," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1240–1253, 2007.

[12] J. B. Lasserre, "Global optimization with polynomials and the problem of moments," *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 796–817, 2001.

[13] N. Ozay, C. Lagoa, and M. Sznaier, "Set membership identification of switched linear systems with known number of subsystems," *Automatica*, vol. 51, pp. 180–191, 2015.

[14] M. Krein and A. Nudelman, "The markov moment problem and extremal problems, volume 50 of translations of mathematical monographs," *American Mathematical Society, Providence, Rhode Island*, 1977.

[15] C. M. Lagoa, H. Che, B. A, and Movsichoff, "Adaptive control algorithm for dencentralized optimal traffic engineering in the internet," *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, pp. 415–428, June 2004.

[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[17] J. Eckstein and W. Yao, "Approximate versions of the alternating direction method of multipliers," Tech. Rep., 2016.

[18] S. Boyd, L. Xiao, and A. Mutapcic, "Subgradient methods," *lecture notes of EE392, Stanford University*, 2003.

[19] J. Eckstein and D. P. Bertsekas, "On the Douglas Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, no. 1-3, pp. 293–318, 1992.