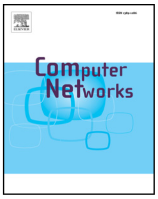
Computer Networks 106 (2016) 260–271

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Control traffic balancing in software defined networks

Shih-Chun Lin[a,1,*], Pu Wang[b,2], Min Luo[c,3]

[a] Broadband Wireless Networking Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA
[b] Department of Electrical Engineering and Computer Science, Wichita State University, Wichita, KS 67260, USA
[c] Shannon Lab, Huawei Technologies Co., Ltd. Santa Clara, USA

ARTICLE INFO

Article history:
Received 6 July 2015
Revised 13 August 2015
Accepted 15 August 2015
Available online 2 September 2015

Keywords:
Control traffic balancing
In-band control
Traffic statistic
Polynomial-time approximation algorithm
Optimal controller location
Software defined networks

ABSTRACT

To promise on-line and adaptive traffic engineering in software defined networks (SDNs), the control messages, e.g., the first packet of every new flow and network traffic statistics, should be forwarded from software defined switches to the controller(s) in a fast and robust manner. As many signaling events and control plane operations are required in SDNs, they could easily generate a significant amount of control traffic that must be addressed together with the data traffic. However, the usage of in-band control channel imposes a great challenge into timely and reliable transmissions of control traffic, while out-band control is usually cost-prohibitive. To counter this, in this paper, the control traffic balancing problem is first formulated as a nonlinear optimization framework with an objective to find the optimal control traffic forwarding paths for each switch in such a way the average control traffic delay in the whole network is minimized. This problem is extremely critical in SDNs because the timely delivery of control traffic initiated by Openflow switches directly impacts the effectiveness of the routing strategies. Specifically, the fundamental mathematical structures of the formulated nonlinear problem and solution set are provided and accordingly, an efficient algorithm, called polynomial-time approximation algorithm (PTAA), is proposed to yield the fast convergence to a near optimal solution by employing the alternating direction method of multipliers (ADMM). Furthermore, the optimal controller placement problem in in-band mode is examined, which aims to find the optimal switch location where the controller can be collocated by minimizing the control message delay. While it is not widely researched except quantitative or heuristic results, a simple and efficient algorithm is proposed to guarantee the optimum placement with regards of traffic statistics. Simulation results confirm that the proposed PTAA achieves considerable delay reduction, greatly facilitating controller's traffic engineering in large-scale SDNs.

1. Introduction

Software Defined Networks (SDNs) have been recognized as the next-generation networking paradigm with the promise to dramatically improve network resource utilization, simplify network management, reduce operating cost, and promote innovation and evolution [1–3]. One of the key ideas in SDNs is to separate the data plane from the control plane by: (i) removing control decisions from the forwarding hardware, e.g., routers or switches, (ii) enabling the forwarding hardware to be programmable through an open and standardized interface, e.g., Openflow [1], and (iii) using a network controller with the supporting management applications (such as routing and resource allocation) to define (in software) the behavior of the network forwarding infrastructure. With such decoupled networking architecture, the timely delivery of control messages for each software defined or Openflow switch largely impacts the efficiency and effectiveness of SDNs, especially when in-band mode [4] is used for control traffic that could significantly affects the system performance with the combined traffic of the original data and the control messages. Thus, it becomes a great challenge to support the load-balancing of in-band control traffic for the minimum network delay via a centralized controller in SDNs.

Furthermore, many signaling events and control plane operations are required in SDNs [2,3,5]. They could easily generate a significant amount of control traffic that must be addressed together with the data traffic. On the other hand, it would be cost-prohibitive to deploy large-scale SDN with out-band control where each switch would be directly connected to the controller with a separate control channel. We foresee that SDN technologies would

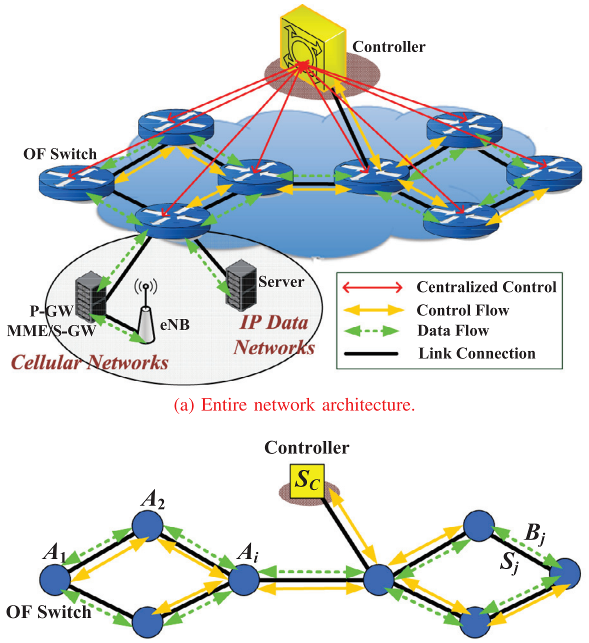
[*] Corresponding author. Tel.: +1 7654048029.
E-mail addresses: slin88@ece.gatech.edu (S.-C. Lin), pu.wang@wichita.edu (P. Wang), min.ch.luo@huawei.com (M. Luo).
[1] Student Member, IEEE
[2] Member, IEEE
[3] Senior Member,IEEE

http://dx.doi.org/10.1016/j.comnet.2015.08.004
1389-1286/© 2015 Elsevier B.V. All rights reserved.

be gradually adopted in enterprises in in-band mode, as we start to address and resolve remaining technical issues. However, existing work all focuses on balancing data traffic in data plane, such as prioritizing the interactive, elastic, and background traffic in [6]. Different from data traffic balancing which aim to evenly distribute data traffic flows among network links, control traffic balancing is much more challenging particular for in-band control [3]. It aims to find the control message forwarding paths of each switch in such a way that the control message delay can be minimized subject to acceptable performance for the original data traffic. This control traffic forwarding problem is extremely critical in SDNs because the timely delivery of control traffic initiated by Openflow switches, e.g, the first packet of every new flow and the traffic/congestion status, directly impacts the effectiveness of the routing strategies determined by the controller.

In this paper, by using queueing network theory [7], we address the control traffic forwarding issue by formulating it as *a nonlinear optimization problem*. However, the complexity of such a formulation is extremely high due to (i) its nonlinearity and (ii) massive variables of link traffic assignments for large-size networks. As a result, the conventional methods for nonlinear optimization problems, such as interior point methods [8], become impractical both in terms of computation time and storage. Therefore, the principle solving method for these large-scale nonlinear problems is to find an approximate and near optimal solution in the solution space [9]. Towards this, we first analyze the fundamental structure of control traffic balancing problem by proving its polynomial time complexity, i.e., its *polynomiality* [9]. Specifically, the optimization problem is justified as a strictly convex framework, and it is proved that the solution can be approximated by a polynomial-time fast algorithm. Furthermore, by deriving the Karush–Kuhn–Tucker (KKT) optimality conditions [8], we reveal the mathematical structure of solution set. Such polynomiality analysis along with the derived KKT conditions enable us to design a fast convergent algorithm for the control traffic balancing problem, based on the alternating direction method of multipliers (ADMM) [10], which is an emerging parallel and fast first-order method for solving large-scale convex optimization problems. In particular, we propose a polynomial-time approximation algorithm (PTAA) that applies the primal-dual update rules of ADMM approach to solve the formulated large-scale convex optimization problem. In particular, PTAA is an iterative algorithm that accurately approximates the optimal solution with fast convergence. We prove that PTAA follows the rapid convergence rate $O(1/c^m)$ with a constant $c > 1$ and iteration number m. Such fast convergence property is extremely important for SDNs because the time-varying traffic pattern in both data plane and control plane may require the fast re-planning of forwarding paths between switches and controller. Performance evaluation confirms that the proposed PTAA provides network delay for control traffic similar to the benchmarks from brute force algorithms, and outperforms the conventional single- and multi-path solutions with at least 80% delay reduction that is time-efficient and could be executed in parallel. To the best of our knowledge, this work is the first to address control traffic balancing problem in SDNs along with the provably fast-convergent algorithm to yield the near optimal solution.

In addition to the control traffic balancing problem, the controller placement problem in in-band mode is also addressed in this paper. Such problem aims to find the optimal controller location (particularly, the best attaching switch location) among all possible ones, which yields the minimum average control message delay. The controller placement problem is not widely researched in the research community to date. In particular, in [11], the distance between a controller and the switches is adopted as the performance metric and several well-known network topologies are evaluated through simulations to find the optimal controller location. In [12], the performance of four different controller placement algorithms are examined in terms of the reliability of control traffic path. Nevertheless, these efforts only look for quantitative or even heuristic results, and the qualitative analysis is still missing. Contrary to the existing solutions, we develop a simple and efficient algorithm that guarantees the optimum solution for the controller placement problem.



(a) Entire network architecture.

(b) Graph model with overlaid traffic flows.

Fig. 1. Network topology of SDNs.

The rest of the paper is organized as follows. The system model is presented in Section 2. Control traffic balancing problem as well as the fundamental problem structure are examined in Section 3. To deal with such a problem, the mathematical solution structure is analyzed and the novel PTAA is proposed through fast ADMM with the further consideration of convergence analysis and the optimal controller location in Section 4. Performance evaluation is provided in Section 5 and the paper is concluded in Section 6.

2. System model

To design load-balancing for control traffic flows, we first describe the network topology of SDNs and then provide the overlaid traffic model in the following.

2.1. Network topology

Indicated by [4], a typical SDN as shown in Fig. 1 generally consists of multiple Openflow enabled switches (i.e. OF switches), which constitutes data plane, and a centralized network controller. The SDN is modeled by a network graph $G = (V, J)$ in Fig. 1b, where V is a set of OF switches with total n switches (i.e., $|V| = n$) and J is a set of links with total $|J|$ links. Rather than exploiting costly out-band control due to a separate control channel, the in-band mode is favored and adopted gradually in practical SDN implementation [4]. In particular, in Fig. 1a, each OF switch needs to send the control purpose traffic, such as the route setup requests for new flows and real-time network congestion status, to the SDN controller. Based on the continuously received control messages, the controller optimizes the best routes for data flows according to dynamically changing traffic patterns and flow QoS requirements and sets up the routing tables of OF switches along the optimal path via certain secure protocols (e.g., Openflow [4]), thus enabling highly efficient data transmissions and superior link utilization [2,3], which is already demonstrated in practical SDNs [6,13]. Despite the promising performance of SDN, its effectiveness and scalability depends on the timely delivery of control messages from OF switches to the controller, which is the focus of this paper. Note that, in the remainder of the paper, we refer to the switches as OF switches in order to simplify the readability.

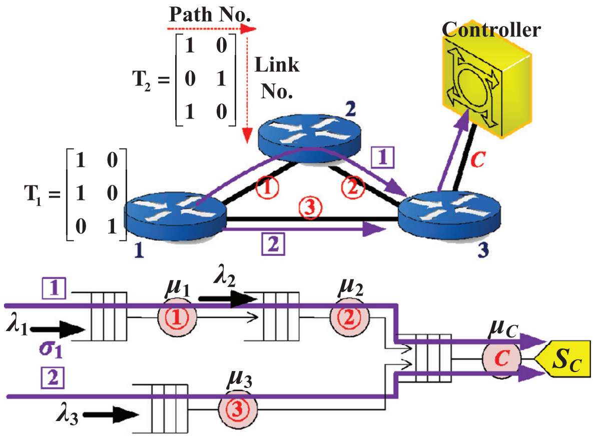

Path No.
$T_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$
Link No.
Controller
$T_1 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$
2
1
C
1
2
3
1
2
3
1
μ_1
λ_2
μ_2
λ_1
1
2
μ_C
σ_1
C
S_C
2
μ_3
λ_3
3

Fig. 2. A simple example of three switches and a controller with three links. Both switch 1 and switch 2 can utilize two respective multi-paths (e.g., the solid lines for switch 1) for their control traffic transmissions to the controller S_C.

2.2. Traffic model

A regenerative process [14] is a general class of stochastic process that certain portions of the process are statistically independent of each other, including renewal process, recurrent Markov chain, and reflected Brownian motion. Without loss of generality, both control and data flows are modeled by regenerative traffic with the regenerative service processes for both link transmission and the controller's serving capability. In particular, the control traffic of each switch i is modeled by a regenerative arrival process A_i with the mean value σ_i. For the jth link and $j \in J$, the existing data flow follows a regenerative arrival process B_j with the mean value λ_j and the link serving time S_j follows another regenerative process with the mean time $1/\mu_j$. Moreover, the optimal centralized controller i^*, whose location will be provided in Section 4.4, has the serving capability with the mean time $1/\mu_C$.

3. Control traffic balancing problem

With traffic models of data and control flows in Section 2.2, we aim to provide a load-balancing scheme that balances link traffic loads with respect to the additional control traffic for in-band control. It is assumed here that a single controller is capable to handle the entire data and control flows. Such network architecture is successfully adopted in real deployment of SDNs [6,13]. In the following, we formulate the control traffic balancing problem as an optimization framework, and provide the corresponding *polynomiality* [9].

3.1. Formulation of the balancing problem

As SDNs provide the centralized control capability with the global view of network status, we address the load-balancing of control traffic to minimize the link transmission delay via an optimization approach. Specifically, the traffic assignment matrix $\mathbf{x} = [x_{ij}]_{i \in V, j \in J}$, where x_{ij} denotes the amount of control traffic that the ith switch contributes to the jth link, is obtained with respect to minimizing the average network delay over the network. To achieve load balancing, multi-path routing is adopted, where given P_i as a set of available paths for the ith switch and $i \in V$, this switch can forward the control messages to the controller via $|P_i|$ available paths. To characterize possible multi-path routings of control flows, for the flow from the ith switch, we define a topology matrix $\mathbf{T}_i$ of size $|J| \times |P_i|$ as follows:

$$\mathbf{T}_i[j, p] = \begin{cases} 1, & \text{if the } j\text{th link lies on the } p\text{th path;} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

A simple three switch scenario is illustrated in Fig. 2. The matrix $\mathbf{T}_i$ maps the traffic from pathes to links and should always be full column-rank to avoid redundant paths. Its left-inverse matrix $\mathbf{T}_i^{-1} = [t_{i1}, t_{i2}, \dots, t_{i|J|}]$ exists and has the size $|P_i| \times |J|$, where t_{ij} is the column vector that maps the jth link to all possible paths of the ith switch's flow. t_{ij} is obtained by multiplying $\mathbf{T}_i^{-1}$ with the jth standard basis e_j, i.e., $t_{ij} = \mathbf{T}_i^{-1} e_j$. While each switch i brings a control flow with the mean value σ_i, the switch i^*, where the controller is directly attached, can send its flow to controller without going through the network (i.e., $x_{i^*j} = 0,\ \forall j \in J$). We set up the equalities for the control flow conservation of switches as $\|\mathbf{T}_i^{-1}[x_{i1}, \dots, x_{i|J|}]^T\|_1 = \sigma_i,\ \forall i \in \widetilde{V} := V \setminus \{i^*\}$, where $\|\cdot\|^T$ and $\|\cdot\|_1$ denote the transpose and 1-norm of vector, respectively. Let $d_{ij} = \|\mathbf{T}_i^{-1} e_j\|_1$, such equalities can be further simplified as

$$\sum_{j \in J} d_{ij} x_{ij} = \sigma_i \quad \forall\ i \in \widetilde{V}, \tag{2}$$

which is the flow conservation constraint, implying that the control flow initiated by each switch is split into multiple outgoing flows on the selected transmission links. Furthermore, with the aid of *Little's law* [7], the average network delay D over the network for the control messages is obtained as

$$D = \frac{1}{\sum_{i \in \widetilde{V}} \sigma_i + \sum_{j \in J} \lambda_j} \sum\nolimits_{j \in J} \frac{\sum_{i \in \widetilde{V}} x_{ij} + \lambda_j}{\mu_j - \left(\sum_{i \in \widetilde{V}} x_{ij} + \lambda_j\right)}. \tag{3}$$

In particular, for link $j \in J$, new packets arrive with rate $(\sum_{i \in \widetilde{V}} x_{ij} + \lambda_j)$ and stay an average time of $1/[\mu_j - (\sum_{i \in \widetilde{V}} x_{ij} + \lambda_j)]$. Summing queue backlogs over all links, the average network delay is thus yielded, as the total external arrivals of control and data traffic into the network are $(\sum_{i \in \widetilde{V}} \sigma_i + \sum_{j \in J} \lambda_j)$. In addition, to balance the traffic loads among all links, every link should have finite transmission delay. From the formulation in (3), such finite link delay conditions are equivalent to

$$\sum_{i \in \widetilde{V}} x_{ij} < \mu_j - \lambda_j \quad \forall\ j \in J, \tag{4}$$

which ensure the incoming traffic rates are less than the link service rates and link delays remain nonnegative. Therefore, with the above accomplishments, we define the Control Traffic Load-Balancing Problem as follows.

Definition 1 (Control Traffic Load-Balancing Problem)**.** Given a SDN modeled by $G = (V, J)$ with the controller location $i^* \in V$, control traffic arrival rates σ_i, a set of topology matrices $\mathbf{T}_i$, $\forall i \in V$, data traffic rates λ_j, and link serving rates μ_j, $\forall j \in J$, the load-balancing optimization problem to be solved by the controller is

$$\begin{array}{ll} \textbf{Minimize} & D\left(x_{ij};\ \begin{array}{l} \forall\ i \in \widetilde{V} = V \setminus \{i^*\} \\ \forall\ j \in J \end{array}\right) \\ \textbf{Subject to} & \quad (2) \quad \text{and} \quad (4) \end{array} . \tag{5}$$

3.2. Polynomiality of the balancing problem

First of all, the minimization objective function in the formulated optimization problem (5) belongs to *a nonseparable nonlin-*

ear continuous function. As indicated by [9], to solve such problems, the leading methodology is to develop iterative and numerical algorithms, whose performance are characterized by the convergence rate. Moreover, different from linear problems, for the nonlinear problems, the length of the solution can be infinite (e.g., when a solution is irrational). Hence, the polynomiality of nonlinear optimization problems is determined by the existence of the polynomial-time converged algorithms which can approximate the optimal solution in the solution space.

Definition 2 (Polynomiality of Nonlinear Problem)**.** A nonlinear optimization problem is of *polynomiality*, if it has polynomial-time converged algorithm that provides optimal solutions with pre-specified accuracy in the solution space.

Theorem 1. *Control Traffic Load-Balancing Problem in* (5) *is of polynomiality*.

Proof. We first prove that the average network delay D in (3) is strictly convex. The basic form of D is provided as $BD(x) = (x + \lambda_j)/[\mu_j - (x + \lambda_j)]$, where μ_j and λ_j are the given constants and x is a single variable. Applying the second derivations to variable x, we obtain the following:

$$\frac{d^2BD(x)}{dx^2} = \frac{2\mu_j[\mu_j - (x+\lambda_j)]}{[\mu_j - (x+\lambda_j)]^4}. \tag{6}$$

The convexity of (6) is determined by the sign of numerator, specifically $\mu_j - (x + \lambda_j)$. Furthermore, as the affine mappings preserve the convexity, we replace x in (6) with $\sum_{i\in\widetilde{V}} x_{ij}$ and get the multiple-to-one strictly convex function (i.e., $(\sum_{i\in\widetilde{V}} x_{ij} + \lambda_j)/[\mu_j - (\sum_{i\in\widetilde{V}} x_{ij} + \lambda_j)]$), where (4) guarantees the strictly convexity. Finally, since the summation of strictly convex function is still strictly convex, the average delay D is a strictly convex function. Thus, (5) belongs to a strictly convex optimization framework with linear constraint functions.

While the nonseparable nonlinear optimization problem is in general hard, Control Traffic Load-Balancing Problem that belongs to a nonseparable convex continuous problem is solvable in polynomial time. In particular, based on the Ellipsoid method [15], a solution approximating the optimal objective value to the convex continuous problem is obtainable in polynomial time, provided that the gradient of the objective functions are available and that the value of the optimal solution is bounded in a certain interval [16]. In other words, any information about the behavior of the objective at the optimum can always be translated to a level of accuracy of the solution vector itself. The interest of solving such optimization problem is thus in terms of the accuracy of the solution rather than the accuracy of the optimal objective value. □

Theorem 1 motivates our following work that analyzes the mathematical structure of solution set and then approximates the objective value of the balancing problem via a polynomial-time fast algorithm.

4. Polynomial-time approximation algorithm (PTAA) for control traffic balancing

We first examine the mathematical structure of solution set to load-balancing framework in Section 3.1. The results suggest us to exploit a fast and possible parallel solving approach for such large systems of SDNs with immense variables. Along with the polynomiality analysis in Section 3.2, the PTAA is proposed through ADMM [10] (a fast first-order method), yielding fast convergence to the optimal solution.

4.1. Mathematical structure analysis of the solution set

To analyze the mathematical structure of solution set, KKT conditions [8] are widely used to examine the convex optimization problem. It provides several equations to jointly get the analytic solution. The analysis for the proposed framework is shown in the following theorem.

Theorem 2. *Given Control Traffic Load-Balancing Problem in Definition 1, the corresponding KKT conditions are*

$$\begin{cases} (2) \text{ and } (4); \\ k_2^j \geq 0 \qquad\qquad\qquad\qquad \forall\ j \in J; \\ k_2^j\left(\sum_{i\in\widetilde{V}} x_{ij} + \lambda_j - \mu_j\right) = 0 \quad \forall\ j \in J; \\ \frac{\mu_j}{\left(\sum_{i\in\widetilde{V}} \sigma_i + \sum_{j\in J} \lambda_j\right)\left[\mu_j - \left(\sum_{i\in\widetilde{V}} x_{ij} + \lambda_j\right)\right]^2} + k_1^i d_{ij} + k_2^j = 0 \\ \qquad\qquad\qquad\qquad\qquad \forall\ i \in \widetilde{V},\ j \in J. \end{cases}$$

Proof. First, the *Lagrangean* of the problem is derived as

$$\frac{1}{\sum_{i\in\widetilde{V}} \sigma_i + \sum_{j\in J} \lambda_j} \sum\nolimits_{j\in J} \frac{\sum_{i\in\widetilde{V}} x_{ij} + \lambda_j}{\mu_j - \left(\sum_{i\in\widetilde{V}} x_{ij} + \lambda_j\right)}$$
$$+ \sum_{i\in\widetilde{V}} k_1^i \left(\sum_{j\in J} d_{ij} x_{ij} - \sigma_i\right) + \sum_{j\in J} k_2^j \left(\sum_{i\in\widetilde{V}} x_{ij} + \lambda_j - \mu_j\right),$$

where k_1^i and k_2^j, $\forall i \in \widetilde{V}, j \in J$ are the lagrangean multipliers. Then, the derivation of obtaining KKT conditions from *Lagrangean* is standard and thus omitted here. □

The following corollary provides the structure of the possible solution sets.

Corollary 1. *The solution of load-balancing problem obeys:*

$$\begin{aligned} x_{ij} &= \frac{\sigma_i - \sum_{l\in J, l\neq j} d_{il} x_{il}}{d_{ij}} && \forall i \in \widetilde{V}; \\ x_{ij} &< \mu_j - \left(\sum_{l\in\widetilde{V}, l\neq i} x_{lj} + \lambda_j\right) && \forall j \in J. \end{aligned} \tag{7}$$

Proof. The results are obtained by solving the simultaneous equations from the KKT conditions in Theorem 2. □

(7) determines the optimal traffic matrix $\mathbf{x}^*$ by iteratively calculating each element x_{ij} with respect to a set of constraint functions. With this understanding and the motivation of polynomial-time solving algorithm from the *polynomiality* of the balancing problem in Section 3.2, we thus propose a fast and parallel polynomial-time approximation algorithm (PTAA) in Section 4.2 to employ iterative update rules of ADMM for the optimal traffic matrix.

4.2. Polynomial-time approximation algorithm (PTAA)

To exploit ADMM [10] for the proposed optimization problem, there are two steps as follows. We first formulate the dual problem from the given primal problem. We then alternatively solve both problems for the optimal solution.

Theorem 3. *The dual problem of (5) is as follows:*

$$\begin{aligned}
&\textbf{Find}: && x_{ij} \text{ and } \beta_{ij} \quad \forall\, i \in \widetilde{V}, j \in J\\
&\textbf{Maximize} && \frac{-1}{\sum_{i\in\widetilde{V}} \sigma_i + \sum_{j\in J} \lambda_j} \sum\nolimits_{j\in J} \frac{\sum_{i\in\widetilde{V}} \beta_{ij} + \lambda_j}{\mu_j - \left(\sum_{i\in\widetilde{V}} \beta_{ij} + \lambda_j\right)}\\
&\textbf{Subject to} && \begin{cases} \beta_{ij} = x_{ij} & \forall\, i \in \widetilde{V}, j \in J\\ \sum\limits_{j\in J} d_{ij}\beta_{ij} = \sigma_i & \forall\, i \in \widetilde{V}\\ (4) \end{cases}
\end{aligned} \tag{8}$$

Proof. A set of auxiliary variables is first introduced as $\boldsymbol{\beta} = [\beta_{ij}]_{i\in\widetilde{V}, j\in J}$ and $\beta_{ij} = x_{ij}$, $\forall i \in \widetilde{V}, j \in J$. Then, the dual problem is obtained from the standard derivations and thus omits here. □

Given (8) and the *penalty parameter* $\rho > 0$ for the *augmented Lagrangian* [10], we consider the update rules for primal variables x_{ij}, β_{ij} and dual variables γ_{ij}, $\forall i \in \widetilde{V}, j \in J$. For x-update, the following iteration is obtained:

$$\mathbf{x}^{(m+1)} := \underset{(4)}{\arg\min}\ \frac{\rho}{2} \sum_{i\in\widetilde{V}} \sum_{j\in J} \left(x_{ij} - \beta_{ij}^{(m)} + \gamma_{ij}^{(m)}\right)^2. \tag{9}$$

To simplify (9), let $\hat{x}_j = \sum_{i\in\widetilde{V}} x_{ij}/(n-1)$, $\hat{\beta}_j^{(m)} = \sum_{i\in\widetilde{V}} \beta_{ij}^{(m)}/(n-1)$, and $\hat{\gamma}_j^{(m)} = \sum_{i\in\widetilde{V}} \gamma_{ij}^{(m)}/(n-1)$. Then, $x_{ij} = \beta_{ij}^{(m)} - \gamma_{ij}^{(m)} + \hat{x}_j - \hat{\beta}_j^{(m)} + \hat{\gamma}_j^{(m)}$ and the x-**update** of ADMM for the primal problem (5) and the corresponding dual problem (8) is

$$\begin{aligned}
&\textbf{Find}: && \hat{x}_j \quad \forall\, j \in J\\
&\textbf{Minimize} && \frac{(n-1)\rho}{2} \sum_{j\in J} \left(\hat{x}_j - \hat{\beta}_j^{(m)} + \hat{\gamma}_j^{(m)}\right)^2.\\
&\textbf{Subject to} && (n-1)\hat{x}_j < \mu_j - \lambda_j \quad \forall\, j \in J
\end{aligned} \tag{10}$$

(10) has $|J|$ single-variable problems and can be independently implemented in parallel for each link j, greatly decreasing the computation complexity.

For β-update, the iteration of $\boldsymbol{\beta}^{(m+1)}$ is

$$\begin{aligned}
&\underset{\sum\limits_{j\in J} d_{ij}\beta_{ij} = \sigma_i\ \forall\ i\in\widetilde{V}}{\arg\min} \frac{1}{\sum_{i\in\widetilde{V}} \sigma_i + \sum_{j\in J} \lambda_j} \sum\nolimits_{j\in J} \frac{\sum_{i\in\widetilde{V}} \beta_{ij} + \lambda_j}{\mu_j - \left(\sum_{i\in\widetilde{V}} \beta_{ij} + \lambda_j\right)}\\
&\quad + \frac{\rho}{2} \sum_{i\in\widetilde{V}} \sum_{j\in J} \left(\beta_{ij} - x_{ij}^{(m+1)} - \gamma_{ij}^{(m)}\right)^2,
\end{aligned} \tag{11}$$

and $\beta_{ij} = x_{ij}^{(m+1)} + \gamma_{ij}^{(m)} + \hat{\beta}_j - \hat{x}_j^{(m+1)} - \hat{\gamma}_j^{(m)}$. To rewrite (11) in terms of $\hat{\beta}_j$, $\forall j \in J$ as usual, we deal with the constraint functions by matrix operation and parameter rearrangement:

$$\begin{aligned}
&\sum_{j\in J} d_{ij}\beta_{ij} = \sigma_i \Rightarrow (n-1) \sum_{j\in J} d_{ij}\hat{\beta}_j = \sigma_i + \sum_{j\in J} \left(d_{ij} \sum_{l\in\widetilde{V}, l\neq i} \beta_{lj} \right)\\
&\quad \Rightarrow (n-1) \sum_{j\in J} d_{ij}\hat{\beta}_j = \sigma_i + \sum_{l\in\widetilde{V}, l\neq i} f_i^l \sigma_l \quad \forall\, i \in \widetilde{V},
\end{aligned}$$

where $f_i^l = \begin{pmatrix} d_{i1} & \cdots & d_{i|J|} \end{pmatrix} \begin{pmatrix} d_{l1} & \cdots & d_{l|J|} \end{pmatrix}^\dagger$ and $(\ \cdot\ )^\dagger$ denotes the pseudo-inverse of matrix. The β-**update** of ADMM is then obtained by

$$\begin{aligned}
&\textbf{Find}: && \hat{\beta}_j \quad \forall\, j \in J\\
&\textbf{Minimize} && \frac{1}{\sum_{i\in\widetilde{V}} \sigma_i + \sum_{j\in J} \lambda_j} \sum\nolimits_{j\in J} \frac{(n-1)\hat{\beta}_j + \lambda_j}{\mu_j - ((n-1)\hat{\beta}_j + \lambda_j)}\\
& && + \frac{(n-1)\rho}{2} \sum\nolimits_{j\in J} \left(\hat{\beta}_j - \hat{x}_j^{(m+1)} - \hat{\gamma}_j^{(m)}\right)^2 .\\
&\textbf{Subject to} && \sum_{j\in J} d_{ij}\hat{\beta}_j = \frac{\sigma_i + \sum_{l\in\widetilde{V}, l\neq i} f_i^l \sigma_l}{n-1} \quad \forall\, i \in \widetilde{V}
\end{aligned} \tag{12}$$

Instead of having multiple single-variable problems, (12) is a $|J|$-variables problem due to the coupled constraint function among $\hat{\beta}_j$, $\forall j \in J$. However, such a constraint function is simply a linear combination of $|J|$ variables and can be easily solved by the powerful SDN controller. Finally, the iteration of **dual-update** of ADMM is obtained:

$$\begin{aligned}
\gamma_{ij}^{(m+1)} &:= \gamma_{ij}^{(m)} + x_{ij}^{(m+1)} - \beta_{ij}^{(m+1)}\\
\Rightarrow \hat{\gamma}_j^{(m+1)} &= \hat{\gamma}_j^{(m)} + \hat{x}_j^{(m+1)} - \hat{\beta}_j^{(m+1)} \ \forall\, i \in \widetilde{V}, j \in J
\end{aligned} \tag{13}$$

With the above accomplishments, we propose PTAA in Algorithm 1 through update rules of ADMM to solve the control traffic balancing problem.

```
Algorithm 1: Polynomial-time Approx. Algo. (PTAA)
  Input  : Primal (5) and dual (8) problems.
  Output: $x_{ij}, \forall\ i \in \widetilde{V}, j \in J$
  Set $\hat{x}_j^{(0)} = 0, \hat{\beta}_j^{(0)} = 0, \hat{\gamma}_j^{(0)} = 0, \forall\ j \in J$
  for $m = 0, 1, \ldots$ do
      Compute $\hat{x}_j^{(m+1)}, \forall\ j \in J$ according to (10)
      Compute $\hat{\beta}_j^{(m+1)}, \forall\ j \in J$ according to (12)
      Compute $\hat{\gamma}_j^{(m+1)}, \forall\ j \in J$ according to (13)
      Set $x_{ij}^{(m+1)}$ from $\hat{x}_j^{(m+1)}, \forall\ i \in \widetilde{V}, j \in J$
  end
```

The convergence analysis and rate of proposed PTAA are provided in the following Section 4.3, given that the objective function of (5) is strictly convex as proved previously.

4.3. Convergence analysis of Algorithm 1: PTAA

We first define the optimal solution for the dual problem of Control Traffic Load-Balancing Problem in the following.

Definition 3. Consider the problem (8), there exists a saddle point $(\{x_{ij}^*\}, \{\beta_{ij}^*\}, \{\gamma_{ij}^*\})$ that satisfies KKT conditions:

$$\{\gamma_{ij}^*\} \in \nabla D(\{x_{ij}^*\}); \tag{14}$$

$$x_{ij}^* - \beta_{ij}^* = 0 \quad \forall\, i \in \widetilde{V}, j \in J \tag{15}$$

where ∇ denotes the gradient operation.

The existence of optimal solution in Definition 3 follows from the strong duality theorem [8]. However, when it fails to hold, PTAA has either unsolvable or unbounded subproblems, or a diverging sequence of $\gamma_{ij}^{(m)}$. In that case, the optimality condition of the subproblems becomes

$$\{\gamma_{ij}^{(m+1)}\} \in \nabla D\big(\{x_{ij}^{(m+1)}\}\big). \tag{16}$$

Moreover,

$$\gamma_{ij}^{(m+1)} = \gamma_{ij}^{(m)} - \rho\beta_{ij}^{(m+1)} \quad \forall\, i \in \widetilde{V}, j \in J. \tag{17}$$

Two crucial properties of objective function in the problem (8) are further examined in the following.

Lemma 1. *The objective function $D(\{x_{ij}\})$ in (5), namely $-D(\{\beta_{ij}\})$ in (8), is strongly convex and it has a Lipschitz continuous gradient. In particular, there exist two constants $\nu_D > 0$ and $L_D > 0$ that satisfy the conditions:*

(i) ***Strong convexity.*** *Given $x := \{x_{ij}\}$ in the domain of D, $D(x) - \frac{\nu_D}{2}\|x\|_2^2$ is convex where $\|\cdot\|_2$ denotes the ℓ_2-norm. In other words $\forall x_1, x_2$ in the domain of D, and $s_1 \in \nabla D(x_1)$ and $s_2 \in \nabla D(x_2)$,*

$$\langle s_1 - s_2, x_1 - x_2 \rangle \geq \nu_D \|x_1 - x_2\|_2^2, \tag{18}$$

where $\langle\ \cdot, \cdot \rangle$ denotes the inner product.

(ii) ***Lipschitz continuous gradient.*** *For all $x := \{x_{ij}\}$, $y := \{y_{ij}\}$ in the domain of D,*

$$\|\nabla D(x) - \nabla D(y)\|_2 \leq L_D \|x - y\|_2. \tag{19}$$

Proof. (i) For the strong convexity, first, D is proved previously as a strictly convex function. Furthermore, D has bounded domain from its definition in (3). As D is strictly convex with a bounded domain, it is strongly convex. That is, for all x_1, x_2 in the domain of D and $t \in [0,\ 1]$, $D(tx_1 + (1-t)x_2) \leq tD(x_1) + (1-t)D(x_2) - \frac{1}{2}\nu_D t(1-t)\|x_1 - x_2\|_2^2$, which implies (18).

(ii) For the Lipschitz continuous gradient, given D as a convex function, it is equivalent to prove the following is convex: $\frac{L_D}{2}\|x\|_2^2 - D(x)$. Since D is twice differentiable, it implies $\nabla^2 D(x) \preceq L_D I$ for all x, where I denotes the identity matrix. That is, the largest eigenvalue of D's Hessian matrix should be no larger than L_D. Without loss the generality and ease the analysis meanwhile, we first consider a simple case:

$$\acute{D}(x_1, x_2, x_3) = K\left[\frac{x_1 + x_2 + \lambda_1}{\mu_1 - (x_1 + x_2 + \lambda_1)} + \frac{x_3 + \lambda_2}{\mu_2 - (x_3 + \lambda_2)}\right]$$

where K is a constant and there are three variables according to general form D in (3). Then $\acute{D}$'s Hessian matrix is

$$\begin{bmatrix} K_1(x_1, x_2) & K_1(x_1, x_2) & 0 \\ K_1(x_1, x_2) & K_1(x_1, x_2) & 0 \\ 0 & 0 & K_2(x_3) \end{bmatrix}$$

where $K_1(x_1, x_2) = \frac{2K\mu_1[\mu_1 - (x_1 + x_2 - \lambda_1)]}{[\mu_1 - (x_1 + x_2 - \lambda_1)]^4}$ and $K_2(x_3) = \frac{2K\mu_2[\mu_2 - (x_3 - \lambda_2)]}{[\mu_2 - (x_3 - \lambda_2)]^4}$. The corresponding eigenvalues are

$$\iota_1 = 0,\ \iota_2 = 2K_1(x_1, x_2),\ \iota_3 = K_2(x_3).$$

While K_1 and K_2 are upper bounded by certain constants respectively, it is thus sufficient to have $L_{\acute{D}} = \max\{2K_1(x_1, x_2), K_2(x_3)\}$. Similarly, as there are at most $|n-1| \times |J|$ eigenvalues from D's general form and each eigenvalue is upper bounded by a certain constant (as suggested by the above example), we can have $L_D = \max_{1 \leq k \leq |n-1||J|} \iota_k$ for the general objective function D and it ends the proof. □

With Lemma 1 in hands, we are able to analyze the global convergence of the proposed PTAA in Algorithm 1. We first introduce two sequences as follows.

Definition 4. Let $(\{x_{ij}^{(m)}\}, \{\beta_{ij}^{(m)}\}, \{\gamma_{ij}^{(m)}\})$ be any sequence generated by Algorithm 1 starting from any initial point $(\{x_{ij}^{(0)}\}, \{\beta_{ij}^{(0)}\}, \{\gamma_{ij}^{(0)}\})$. We have two series as

$$U^{(m)} = \sum_{i \in \tilde{V}} \sum_{j \in J} \left[\left(\beta_{ij}^{(m)} - \beta_{ij}^*\right)^2 + \frac{1}{\rho^2}\left(\gamma_{ij}^{(m)} - \gamma_{ij}^*\right)^2\right];$$

$$W^{(m)} = \sum_{i \in \tilde{V}} \sum_{j \in J} \left[\left(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right)^2 + \frac{1}{\rho^2}\left(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right)^2\right].$$

Motivated by [17], the following convergence analysis is based on bounding the error $U^{(m)}$ and estimate its decrease.

Lemma 2. *Given Definition 3 and the strong convexity in Lemma 1, the sequence $\{U^{(m)}\}$ obeys*

$$U^{(m)} - U^{(m+1)} \geq H\left(\left\{\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right\}, \left\{\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right\}\right) + \frac{2\nu_D}{\rho} \sum_{i \in \tilde{V}} \sum_{j \in J} \left(\beta_{ij}^{(m+1)} - \beta_{ij}^*\right)^2 \tag{20}$$

where $H(\cdot, \cdot) = W^{(m)} + \frac{2}{\rho} \sum_{i \in \tilde{V}} \sum_{j \in J} (\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)})(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)})$.

Proof. By the convexity of D and the optimality conditions ((14)) and (16), we have $\sum_{i \in \tilde{V}} \sum_{j \in J} (\beta_{ij}^{(m+1)} - \beta_{ij}^*)(\gamma_{ij}^{(m+1)} - \gamma_{ij}^*) \geq \nu_D \sum_{i \in \tilde{V}} \sum_{j \in J} (\beta_{ij}^{(m)} - \beta_{ij}^*)^2$. In addition, it follows from (15) and (17) that $\rho(\beta_{ij}^{(m+1)} - \beta_{ij}^*) = \gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}$ and thus

$$\frac{1}{\rho} \sum_{i \in \tilde{V}} \sum_{j \in J} \left(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right)\left(\gamma_{ij}^{(m+1)} - \gamma_{ij}^*\right) \geq \nu_D \sum_{i \in \tilde{V}} \sum_{j \in J} \left(\beta_{ij}^{(m)} - \beta_{ij}^*\right)^2. \tag{21}$$

From the definition of $U^{(m)}$ and the identity that $(a-c)^2 - (b-c)^2 = (a-b)^2 + 2(b-c)(a-b)$, it follows that

$$\begin{aligned} U^{(m)} - U^{(m+1)} &= \sum_{i \in \tilde{V}} \sum_{j \in J} \Bigg[\left(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right)^2 \\ &\quad + 2\left(\beta_{ij}^{(m+1)} - \beta_{ij}^*\right)\left(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right) \\ &\quad + \frac{1}{\rho^2}\left(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right)^2 \\ &\quad + \frac{2}{\rho^2}\left(\gamma_{ij}^{(m+1)} - \gamma_{ij}^*\right)\left(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right)\Bigg] \\ &\geq W^{(m)} + \frac{2\nu_D}{\rho} \sum_{i \in \tilde{V}} \sum_{j \in J} \left(\beta_{ij}^{(m)} - \beta_{ij}^*\right)^2 \\ &\quad + \frac{2}{\rho} \sum_{i \in \tilde{V}} \sum_{j \in J} \left(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right)\left(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right). \end{aligned}$$

The last inequality comes from (21) and it ends the proof. □

In the following theorem, a more refined bound is obtained by applying the strong convexity of D to the cross term of $H(\ \cdot, \cdot)$, i.e., $2/\rho \sum_{i \in \tilde{V}} \sum_{j \in J} (\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)})(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)})$.

Theorem 4. *Given Definition 4, we have*

$$U^{(m)} - U^{(m+1)} \geq W^{(m)} + \frac{2\nu_D}{\rho}\sum_{i\in\widetilde{V}}\sum_{j\in J}\left(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right)^2 + \frac{2\nu_D}{\rho}\sum_{i\in\widetilde{V}}\sum_{j\in J}\left(\beta_{ij}^{(m+1)} - \beta_{ij}^{*}\right)^2. \tag{22}$$

Proof. Applying (16) at two consecutive iterations, we have the following:

$$\begin{cases} \{\gamma_{ij}^{(m-1)} - \rho\beta_{ij}^{(m)}\} \in \nabla D\big(\{\beta_{ij}^{(m)}\}\big); \\ \quad \{\gamma_{ij}^{(m+1)}\} \in \nabla D\big(\{\beta_{ij}^{(m+1)}\}\big). \end{cases} \tag{23}$$

The difference of the two terms on the left in (23) is

$$\gamma_{ij}^{(m-1)} - \rho\beta_{ij}^{(m)} - \gamma_{ij}^{(m+1)} = \gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)} \tag{24}$$

for all $i \in \widetilde{V}, j \in J$ where the equality comes from (17). By (23), (24), and (18), it follows that $\sum_{i\in\widetilde{V}}\sum_{j\in J}(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)})(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}) \geq \nu_D \sum_{i\in\widetilde{V}}\sum_{j\in J}(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)})$. Substituting into (20) ends the proof. □

Now the bound in Theorem 4 is used to give the global convergence of the proposed PTAA.

Theorem 5 (Global Convergence)**.** *Consider the sequence $\{(\{x_{ij}^{(m)}\}, \{\beta_{ij}^{(m)}\}, \{\gamma_{ij}^{(m)}\})\}_m$ generated by the proposed PTAA in Algorithm 1. Under the assumption that such sequence is bounded, $\{(\{x_{ij}^{(m)}\}, \{\beta_{ij}^{(m)}\}, \{\gamma_{ij}^{(m)}\})\}_m$ converges to a KKT point $(\{x_{ij}^{*}\}, \{\beta_{ij}^{*}\}, \{\gamma_{ij}^{*}\})$ of the problem (8), namely, $U^{(m)} \to 0$.*

Proof. Being bounded, $\{(\{x_{ij}^{(m)}\}, \{\beta_{ij}^{(m)}\}, \{\gamma_{ij}^{(m)}\})\}_m$ has a converging subsequence $\{(\{x_{ij}^{(m_k)}\}, \{\beta_{ij}^{(m_k)}\}, \{\gamma_{ij}^{(m_k)}\})\}_{m_k}$. And let $\lim_{k\to\infty}\{(\{x_{ij}^{(m_k)}\}, \{\beta_{ij}^{(m_k)}\}, \{\gamma_{ij}^{(m_k)}\})\}_{m_k} = (\{\bar{x}_{ij}\}, \{\bar{\beta}_{ij}\}, \{\bar{\gamma}_{ij}\})$. We will show $(\{\bar{x}_{ij}\}, \{\bar{\beta}_{ij}\}, \{\bar{\gamma}_{ij}\})$ is a KKT point in the following. First, from (22), we conclude that $U^{(m)}$ is monotonically non-increasing and thus converging. It means $W^{(m)} \to 0$, so $\forall \;\; i \in \widetilde{V}, j \in J$, $\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)} \to 0$ and $\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)} \to 0$, or equivalently $r^{(m)} := x_{ij}^{(m+1)} - \beta_{ij}^{(m+1)} \to 0$ as $m \to \infty$. By passing limit over the subsequence, we have: $\bar{x}_{ij} - \bar{\beta}_{ij} = 0, \forall \;\; i \in \widetilde{V}, j \in J$. Moreover, taking limit over the subsequence on both sides of (16), we obtain: $\{\bar{\gamma}_{ij}\} \in \nabla D(\{\bar{x}_{ij}\})$. Therefore, $(\{\bar{x}_{ij}\}, \{\bar{\beta}_{ij}\}, \{\bar{\gamma}_{ij}\})$ satisfies the KKT condition of (8). Since $(\{\bar{x}_{ij}\}, \{\bar{\beta}_{ij}\}, \{\bar{\gamma}_{ij}\})$ is a KKT point, we can now let $(\{x_{ij}^{*}\}, \{\beta_{ij}^{*}\}, \{\gamma_{ij}^{*}\}) = (\{\bar{x}_{ij}\}, \{\bar{\beta}_{ij}\}, \{\bar{\gamma}_{ij}\})$. From $(\{x_{ij}^{(m_k)}\}, \{\beta_{ij}^{(m_k)}\}, \{\gamma_{ij}^{(m_k)}\}) \to (\{\bar{x}_{ij}\}, \{\bar{\beta}_{ij}\}, \{\bar{\gamma}_{ij}\})$ in k and the convergence of $U^{(m)}$, it follows $U^{(m)} \to 0$ in m and ends the proof. □

To further consider the precise convergence behavior, we establish the global linear convergence results for our PTAA solution in the following. We first derive Lemma 3 to bound $\sum_{i\in\widetilde{V}}\sum_{j\in J}(\gamma_{ij}^{(m+1)} - \gamma_{ij}^{*})^2$.

Lemma 3. *While ∇D is Lipschitz continuous with constant L_D from Lemma 1, we have $\sum_{i\in\widetilde{V}}\sum_{j\in J}(\gamma_{ij}^{(m+1)} - \gamma_{ij}^{*})^2 \leq L_D^2 \sum_{i\in\widetilde{V}}\sum_{j\in J}(\beta_{ij}^{(m+1)} - \beta_{ij}^{*})^2$.*

Proof. By the optimality conditions ((14)) and (16) together with the Lipschitz continuity of ∇D, it follows that $\sum_{i\in\widetilde{V}}\sum_{j\in J}(\gamma_{ij}^{(m+1)} - \gamma_{ij}^{*})^2 = \|\nabla D(\{\beta_{ij}^{(m+1)}\}) - \nabla D(\{\beta_{ij}^{*}\})\|_2^2 \leq L_D^2 \sum_{i\in\widetilde{V}}\sum_{j\in J}(\beta_{ij}^{(m+1)} - \beta_{ij}^{*})^2$. □

Note that the inequality (22) has the form

$$U^{(m)} - U^{(m+1)} \geq C, \tag{25}$$

where C stands for its right-hand side. From Lemma 3, we can further obtain

$$C \geq c_1\sum_{i\in\widetilde{V}}\sum_{j\in J}\left(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right)^2 + c_2\sum_{i\in\widetilde{V}}\sum_{j\in J}\left(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right)^2 + c_3\sum_{i\in\widetilde{V}}\sum_{j\in J}\left(\beta_{ij}^{(m+1)} - \beta_{ij}^{*}\right)^2 + c_4\sum_{i\in\widetilde{V}}\sum_{j\in J}\left(\gamma_{ij}^{(m+1)} - \gamma_{ij}^{*}\right)^2 \tag{26}$$

where $c_1 = 1 + 2\nu_D/\rho > 0$, $c_2 = 1/\rho^2 > 0$, and $c_3, c_4 > 0$. Thus, we are able to analyze the global linear convergence of the proposed PTAA in the following theorem.

Theorem 6 (Global Linear Convergence)**.** *Under the same assumption of Theorem 5, the proposed PTAA in Algorithm 1 converges to the optimal solutions with rate $O(1/c^m)$, where $c > 1$ is a constant and m is the number of iterations.*

Proof. From the definition of $U^{(m+1)}$ and (26), it follows that $C \geq \delta U^{(m+1)}$ by letting $\delta := \min\{c_3, \; c_4\rho^2\}$. Combining with (25), we further obtain the inequality: $U^{(m)} \geq (1 + \delta)U^{(m+1)}$. Next, we apply this inequality for $U^{(0)}$ up to $U^{(m-1)}$ as follows: $U^{(0)} \geq (1+\delta)U^{(1)} \geq (1+\delta)^2U^{(2)} \geq \cdots \geq (1+\delta)^mU^{(m)} \Rightarrow U^{(m)} \leq (1+\delta)^{-m}U^{(0)}$ where the last inequality holds for every iteration (i.e., $\forall m$). By setting $c = 1 + \delta$ ends the proof. □

To keep the proof of Theorem 6 easy to follow, we have avoided giving the explicit formulas of c_i's and thus also of δ. On the other hand, we provide the idea what quantities affect δ by discussing the value of δ in the following.

Corollary 2. *As mentioned in the proof of Theorem 6, it follows that $\delta = 2\nu_D/(\rho + \frac{L_D^2}{\rho})$. Furthermore, since it is better to have larger δ, we can choose $\rho = L_D$ and obtain $\delta_{\max} = 1/\kappa_D$ where $\kappa_D = L_D/\nu_D$ is the condition number of function D.*

Proof. From Lemma 3, we obtain the following for δ:

$$\frac{2\nu_D}{\rho} = \delta\left(1 + \frac{L_D^2}{\rho^2}\right)$$

by comparing the definition of $U^{(m+1)}$ with (26). Moreover, $\delta_{\max}$ is yielded by $\frac{d}{d\rho}\delta|_{\delta=\delta_{\max}} = 0$. □

Not surprisingly, the convergence rate is negatively affected by the condition number of D.

In the next corollary, we consider some convergence properties of the sequence $\{W^{(m)}\}$, which facilitates the stopping rule design for the proposed PTAA.

Corollary 3. *Given $\{W^{(m)}\}$ from Definition 4, $W^{(m)}$ is non-increasing and converges at rate $O(1/c^m)$.*

Proof. We first consider the non-increasing property. Following the same procedure in the proof of Lemma 2,

$$\begin{aligned} W^{(m)} - W^{(m+1)} = \sum_{i\in\widetilde{V}}\sum_{j\in J}\Big[&\left(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right)^2 \\ &+ 2\left(\beta_{ij}^{(m+1)} - \beta_{ij}^{(m+2)}\right)\left(\beta_{ij}^{(m)} - \beta_{ij}^{(m+1)}\right) \\ &+ \frac{1}{\rho^2}\left(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right)^2 \\ &+ \frac{2}{\rho^2}\left(\gamma_{ij}^{(m+1)} - \gamma_{ij}^{(m+2)}\right) \\ &\times\left(\gamma_{ij}^{(m)} - \gamma_{ij}^{(m+1)}\right)\Big] \geq W^{(m)} - W^{(m)} = 0 \end{aligned}$$

where the last inequality comes from the fact that $2ab \geq -(a-b)^2$. We next consider the convergence rate of $W^{(m)}$. As indicated by (22), $U^{(m)} \geq W^{(m)}$ for all iterations. Furthermore, from the results of Theorem 6, it follows that $W^{(m)} \leq U^{(0)}/(1+\delta)^m$ and ends the proof. □

Remark 1. Rather than based on heuristic principles, the above corollary suggests that the sequence $\{W^{(m)}\}$ can be used as a natural stopping rule for the proposed update rules in Section 4.2, which decreases at rate $1/c^m$. In particular, our stopping-rule sequence enjoys the non-increasing property without fluctuating over iterations. Meanwhile, as expected from its definition, $W^{(m)}$ scales linearly with the size of system parameters, i.e., $|n-1| \times |J|$. This implies that $W^{(m)}$ is an ideal candidate for the stopping rule: PTAA in Algorithm 1 can be terminated when $W^{(m)}/(|n-1||J|)$ is below a certain threshold.

Upon this stage, we completely solve the load-balancing problem via the proposed fast and parallel PTAA.

4.4. Optimal location of SDN controller

Through the proposed load-balancing solution, the optimal controller location problem is investigated, which aims to find the best switch location where the controller is directly attached and to yield the minimum average control message delay. In particular, we evaluate the average delay among all possible attaching locations for the controller and select the one with the minimum delay for the optimal controller location. Algorithm 2 presents a space-search approach to obtain such optimal location, whose complexity is $O(n/c^m)$ from n times of the proposed Algorithm 1: PTAA. In each of location search cycle, we calculate the topology matrices and the corresponding parameters d_{ij}, $\forall i \in V, j \in J$ with respect to the given candidate of controller attaching location. We then formulate the load-balancing problem in Section 3.1 and solve the problem by the proposed PTAA in Section 4.2 for this specific candidate. After evaluating all candidates, the one provides the least average network delay is selected for the optimal location. This simple but time-efficient algorithm indeed facilitates optimal controller location in SDNs.

Algorithm 2: Controller Location Searching

```
Input  : $\sigma_i$, $\lambda_j$, $\mu_j$, $\forall i \in V$, $j \in J$
Output: $i^*$   % Optimal attaching location for controller
$D_{min} = \infty$   % Initialization
for $loc = 1 : |V|$ do    % Location search cycle
    Get $\mathbf{T}_i(loc)$ via (1) and $d_{ij}(loc) = \|\mathbf{T}_i^{-1}(loc)e_j\|_1$,
        $\forall i \in V \setminus \{loc\}$, $j \in J$
    Formulate balancing problem (5) w.r.t. $d_{ij}(loc)$
    $D \leftarrow$ Algorithm 1$(d_{ij}(loc), (5))$
    if $D < D_{min}$ then
        $D_{min} = D$ and $i^* = loc$
    end
end
```

5. Performance evaluation

We evaluate the proposed PTAA and compare it with single- and multi-path forwarding as well as the benchmark. Two practical scenarios are established: Internet2 OS3E network [18] and Sprint GIP backbone network [19]. Average network delay and optimal controller location are obtained with respect to control and data traffic statistics. Simulation results confirm that the proposed PTAA achieves remarkable delay reduction via a fast and parallel computation approach, thus favored by practical implementation in large-scale SDNs. In particular, we set the penalty parameter $\rho = 10^{-3}$ in PTAA after an empirical examination. Also, the rapid convergence rate of PTAA follows our analytical derivation as $O(1/c^m)$ shown in Fig. 3, where y-axis is average network delay as our targeted objective. It provides the satisfactory values after 300 iterations which serves as a desired stopping point. For both single- and multi-path solutions, the number of hops is selected as routing metric in [11], and shortest path strategy is adopted to guide control traffic from a switch to controller. Specifically, multi-path forwarding equally splits control traffic among all available next hops and applies the shortest path strategy to the corresponding routes. The benchmark is further generated from the exhaustive searching via brute force.

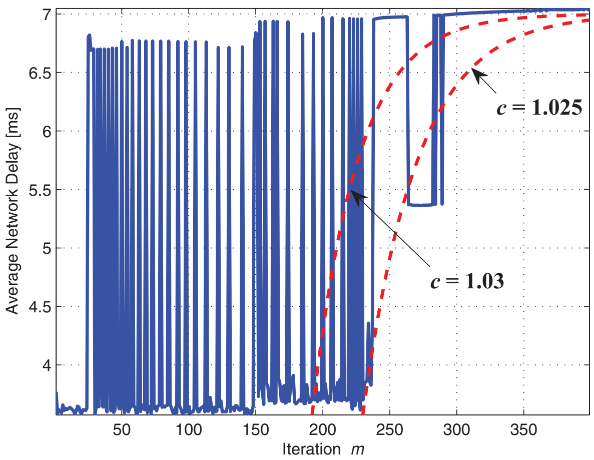
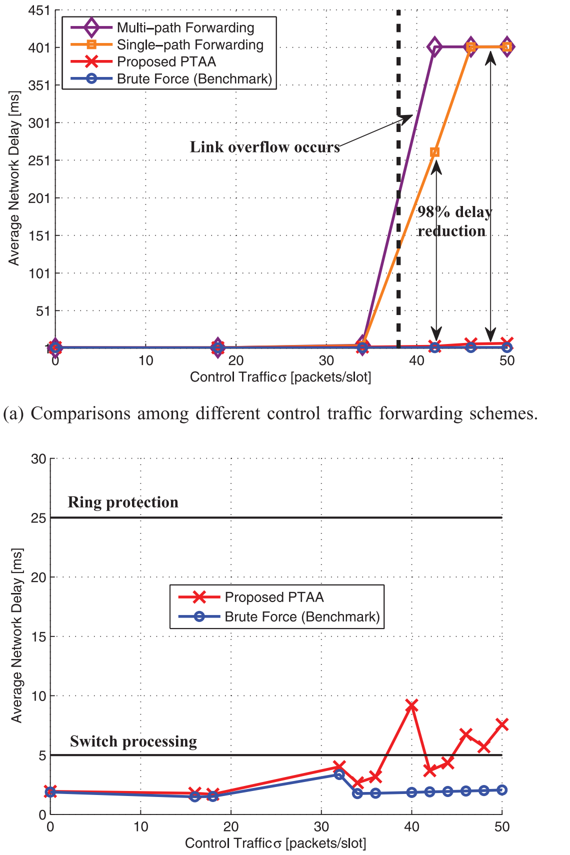


Fig. 3. Linear-fast convergence of the proposed PTAA in Algorithm 1 with rate $O(1/c^m)$.

In addition, to get a sense for the magnitude of these network delays, we also compare the proposed PTAA and benchmark to three bounds relevant to today's networks, such as expected recovery times and delays expected within a forwarding device. These bounds are scaled by half to align with one-way latency of our concerns. In particular, the lowest one is **switch processing**: roughly 10 [ms] corresponds to the measured delay of today's Openflow switches. It measures the time from a packet is sent to a response is received, through an unloaded controller directly attached to the switch. The middle one is **ring protection**: 50 [ms] provides the target restoration time of a SONET ring. It covers the time from fault detection to when traffic is flowing in the opposite direction along the ring. Finally, the highest one is **shared-mesh restoration**: around 200 [ms] serves as the point at which voice calls start to drop, or ATM circuit rerouting may be triggered.

5.1. Internet2 OS3E network

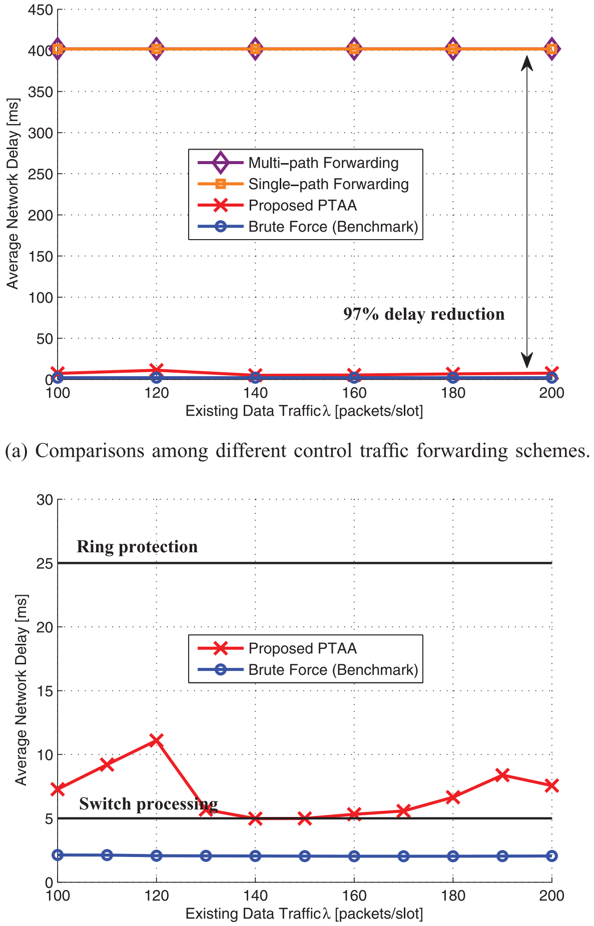
Regarding traffic statistics in Internet2 OS3E with 27 nodes and 36 links, the data arrival rate λ_j and serving rate μ_j over links $j \in J$ are set as 200 and 1000 [packets/slot], respectively. Fig. 4 shows that the average network delay of proposed PTAA and several possible solutions with respect to the control traffic. Specifically, in Fig. 4a, PTAA always has lower delay than single- and multi-path solution, and closes to the benchmark. With increasing control traffic from switches, both single- and multi-path solutions bring dramatic delay increase due to the occurrence of link overflow; however, our solution is able to tolerate such higher loads through distributing extra control traffic over links with lighter data loads. Fig. 4b further shows that PTAA can maintain network



(a) Comparisons among different control traffic forwarding schemes.

(b) Comparisons with benchmark and bounds relevant to today's networks.

Fig. 4. Average network delay in Internet2 OS3E with 27 nodes and 36 links [18] with respect to control traffic.



(a) Comparisons among different control traffic forwarding schemes.

Fig. 5. Average network delay in Internet2 OS3E with respect to existing data traffic. Control traffic rate $\sigma_i = 40, \forall i \in V$.

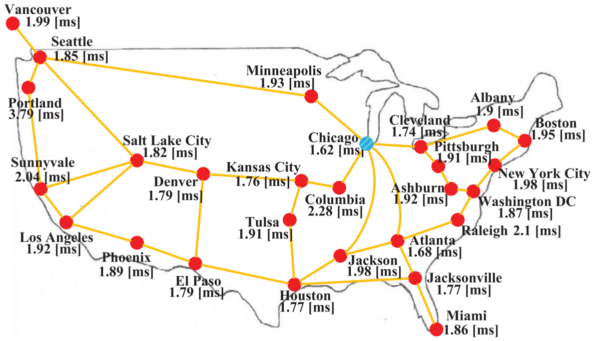


Fig. 6. Optimal controller location in **Chicago** in Internet2 OS3E with control traffic rate $\sigma_i = 25, \forall i \in V$, link data traffic rate $\lambda_j = 200$, and link serving rate $\mu_j = 1000, \forall j \in J$.

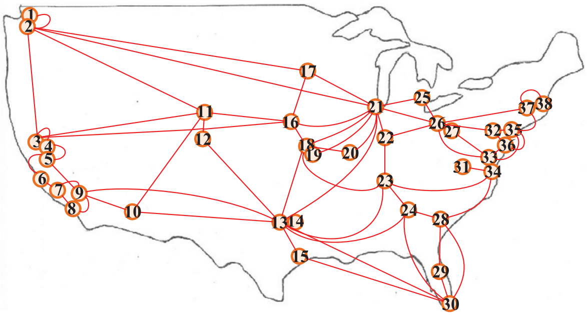


Fig. 7. Sprint GIP network topology of North America with 38 nodes and 66 links [19].

delay lower than the latency of switch processing for less control traffic, and has slightly increased delay for increasing traffic.

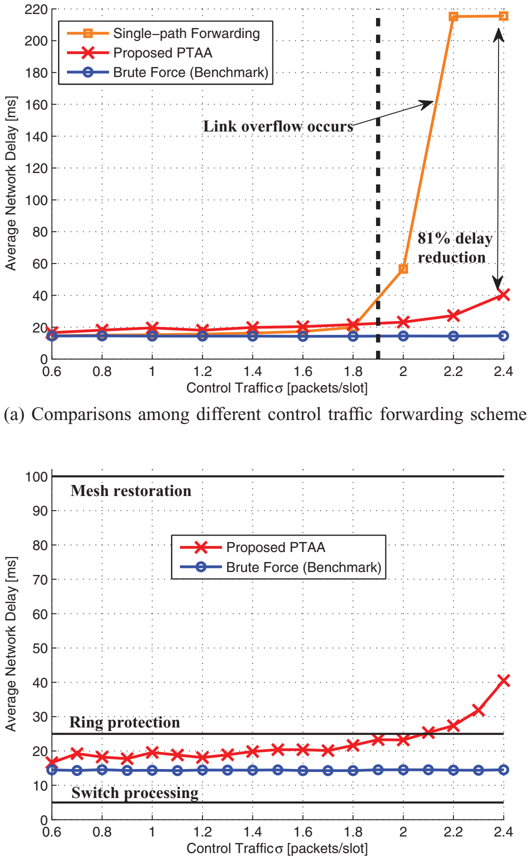
Fig. 5 shows the impact from the increasing data traffic, given control traffic rate from switches 40 [packets/slot]. PTAA outperforms single- and multi-path solutions with 97% delay reduction, and can support network delay much lower than the latency of ring protection, exhibiting the capability of balancing control traffic. In addition, Fig. 6 provides the optimal controller location with regards of control and data traffic statistics. While the hop-count is the only concerned attribute in Heller et al.'s previous work [11], the optimal location from the proposed traffic driven designs in Chicago matches their concerns of average hop number. However, once the control or data traffic pattern changes, we can still obtain the respectively optimal location due to more realistic latency consideration, thus providing greater practicability in SDNs.

5.2. Sprint GIP backbone network

The real backbone network topology with the actual link delay of data traffic is provided by Sprint GIP network [19]. Such delay information is utilized to estimate the corresponding data traffic arrival and serving rates. As shown in Fig. 7, the GIP network topology of North America has 38 nodes and 66 links and is adopted for our evaluation as follows. Given link serving rate 1000 [packets/slot], Fig. 8 shows the average network delay with respect to control traffic. Similarly, single-path forwarding has large delay increase due to link overflow. PTAA outperforms single-path solution with small gap between benchmark, and maintains network delay lower than the latency of mesh restoration in this actual network.

Table 1
Optimal controller location in **26 Akron,OH** in Sprint GIP network with control traffic rate $\sigma_i = 12, \forall i \in V$.

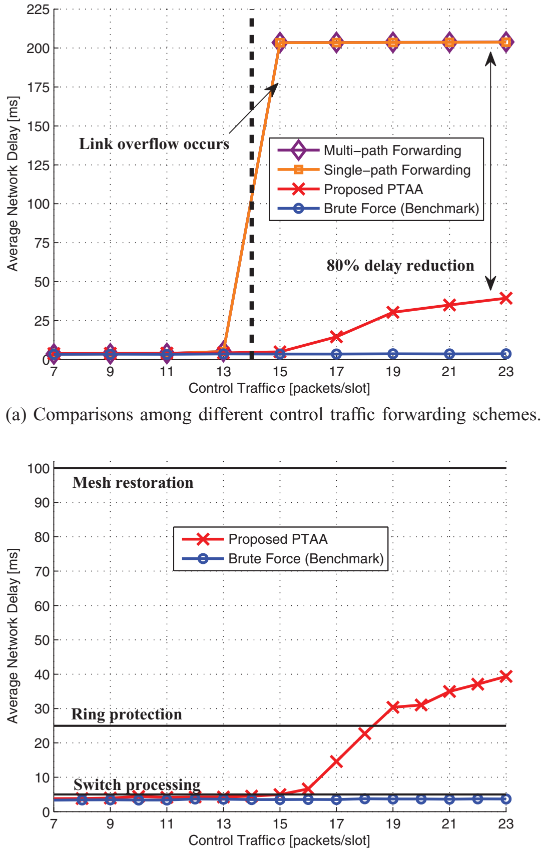
City	Delay [ms]	City	Delay [ms]	City	Delay [ms]
1 Seattle,WA	5.5	14 Dallas,TX	4.03	27 Pittsburgh,PA	4.06
2 Tacoma,WA	5.5	15 Houston,TX	5.08	28 Fairfax,SC	5.72
3 Oroville,CA	5.5	16 Omaha,NE	4.22	29 Orlando,FL	4.98
4 Rancho Cordova,CA	5.5	17 Saint Paul,MM	4.25	30 Miami,FL	**
5 Stockton,CA	5.5	18 Kansas City,MO	4.72	31 Ashburn,VA	15.2
6 San Jose,CA	5.5	19 Lee's Summit,MO	**	32 Harrison,NJ	**
7 Los Angeles,CA	5.5	20 ST. Louis,MO	4.38	33 Relay,MD	4.89
8 Anaheim,CA	5.5	21 Chicago,IL	4.17	34 Washington,DC	**
9 Rialto,CA	5.5	22 Roachdale,IN	4.67	35 New York City,NY	5.43
10 Phoenix,AZ	5.5	23 Nashville,TN	4.52	36 Pennsauken,NJ	7.23
11 Cheyenne,WY	4.05	24 Atlanta,GA	5.12	37 Springfield,MA	8.34
12 Denver,CO	8.17	25 Detroit,MI	4.48	38 Boston,MA	6.61
13 Fort Worth,TX	5.62	26 Akron,OH	4.01	**: Link overflow occurs	



(a) Comparisons among different control traffic forwarding schemes.

(b) Comparisons with benchmark and bounds relevant to today's networks.

Fig. 8. Average network delay in Sprint GIP network with respect to control traffic. Link serving rate $\mu_j = 1000, \forall j \in J$.



(a) Comparisons among different control traffic forwarding schemes.

(b) Comparisons with benchmark and bounds relevant to today's networks.

Fig. 9. Average network delay in Sprint GIP network with respect to control traffic. Link serving rate $\mu_j = 1200, \forall j \in J$.

With better link serving capability (i.e., serving rate 1200 [packets/slot]), Fig. 9 shows that the proposed PTAA surpass single- and multi-path solutions with the delay close to the benchmark. The difference between single- and multi-path forwarding is almost negligible with nearly aligned network delay in Fig. 9a and our solution yields great delay reduction as compared to these approaches. As shown in Fig. 9b, PTAA supports the performance comparable to the latency of switch processing for less control traffic, and has network delay around the latency of ring protection for normal traffic. In addition, TABLE 1 provides the optimal controller location in this real network with respect to control traffic rate 12 [packets/slot] from switches. Specifically, the proposed PTAA is evaluated in every possible controller attaching location (i.e., all switches' locations) via Algorithm 2 in Section 4.4. Some cities (e.g., Lee's Summit, Miami, Harrison, and Washington) are not suitable for controller placement as they bring link overflow over network and the optimal location is obtained as Akron for Sprint GIP network. Above evaluations all suggest that by employing the information of traffic statistics, our solution resorts to better link resource utilization through fast ADMM and outperforms other possible schemes with at least 80% network delay reduction. Therefore, we introduce a new paradigm for control traffic load balancing and offer a novel avenue towards traffic statistics driven designs in SDNs.

6. Conclusion

Load-balancing of in-band control traffic is addressed and the proposed PTAA solves the balancing problem in an efficient and parallel manner. The optimal controller location is further exploited for the minimum average network delay. Performance evaluation confirms that PTAA successfully demonstrates communication efficiency with at least 80% delay reduction via a fast and low complexity approach. We have presented a novel paradigm to facilitate online configurations of centralized controller in practical SDN implementations.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: enabling innovation in campus networks, ACM SIGCOMM Comput. Commun. Rev. 38 (2) (2008) 69–74.

[2] I.F. Akyildiz, A. Lee, P. Wang, M. Luo, W. Chou, A roadmap for traffic engineering in sdn-openflow networks, Comput. Netw. 71 (2014) 1–30.

[3] I.F. Akyildiz, P. Wang, S.C. Lin, Softair: a software defined networking architecture for 5g wireless systems, Comput. Netw. 85 (2015) 1–18.

[4] ONF, "OpenFlow Switch Specification," version 1.4.0. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf.

[5] S.H. Yeganeh, A. Tootoonchian, Y. Ganjali, On scalability of software-defined networking, IEEE Commun. Mag. 51 (2) (2013) 136–141.

[6] C.Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, R. Wattenhofer, Achieving high utilization with software-driven wan, in: Proceedings of ACM SIGCOMM 2013 Conference on SIGCOMM, 2013, pp. 15–26.

[7] G. Bolch, S. Greiner, H. Meer, K.S. Trivedi, Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications, 2nd ed., Wiley, New York, 2006.

[8] E.K.P. Chong, S.H. Zak, An Introduction to Optimization, 3rd ed., Wiley-Sons, 2008.

[9] D.S. Hochbaum, Complexity and algorithms for nonlinear optimization problems, Ann. Oper. Res. 153 (1) (2007) 257–296.

[10] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, Found. Trends Mach. Learn. 3 (2011) 1–122.

[11] B. Heller, R. Sherwood, N. McKeown, The controller placement problem, in: Proceedings of ACM HotSDN Conference on Hot topics in software defined network, 2012.

[12] Y. Hu, W. Wendong, X. Gong, X. Que, C. Shiduan, Reliability-aware controller placement for software-defined networks, in: Proceedings of 2013 IFIP/IEEE International Symposium on Integrated Network Management, 2013, pp. 672–675.

[13] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Holzle, S. Stuart, A. Vahdat, B4: Experience with a globally-deployed software defined wan, in: Proceedings of ACM SIGCOMM Conference, 2013, pp. 3–14.

[14] S. Karlin, H.M. Taylor, A First Course in Stochastic Processes, Academic Press, New York, 1975.

[15] S.A. Vavasis, Nonlinear Optimization: Complexity Issues, Oxford University Press, New York, 1991.

[16] A.S. Nemirovsky, D.B. Yudin, Problem Complexity and Method Efficiency in Optimization, Wiley, New York, 1983.

[17] W. Deng, W. Yin, On the global and linear convergence of the generalized alternating direction method of multipliers, Tech. Rep., Dept. Computational Appl. Math., Rice Univ., Houston, TX, USA, 2012.

[18] Internet2, "Open science, scholarship and services exchange", 2013, online available: http://www.internet2.edu/network/ose.

[19] Sprint, Overland Park, KS, "Sprint IP network performance", 2011, online available: http://www.sprint.net/performance.

Shih-Chun Lin (S'08) received the B.S. degree in electrical engineering and the M.S. degree in communication engineering from National Taiwan University in 2008 and 2010, respectively. He is a graduate research assistant in the Broadband Wireless Networking Laboratory (BWN Lab), School of Electrical and Computer Engineering, Georgia Institute of Technology. Currently, he is working toward the Ph.D. degree in electrical and computer engineering under the supervision of Prof. Ian F. Akyildiz. His research interests include wireless underground sensor networks, software defined networking, large machine-to-machine communication, cognitive radio networks, and statistical scheduling in wireless systems.

Pu Wang (M'05) received the B.S. degree in electrical engineering from the Beijing Institute of Technology, Beijing, China, in 2003, the M.Eng. degree in computer engineering from the Memorial University of Newfoundland, St. Johns, NL, Canada, in 2008, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, in 2013. He is currently an Assistant Professor with the Department of Electrical Engineering and Computer Science, Wichita State University, Wichita, KS. Dr. Wang received the BWN Lab Researcher of the Year award in 2012, Georgia Institute of Technology. He received the TPC top ranked paper award of IEEE DySPAN 2011. He was also named Fellow of the School of Graduate Studies, 2008, Memorial University of Newfoundland. His research interests include wireless sensor networks, cognitive radio networks, software defined networks, nanonetworks, multimedia communications, wireless communications in challenged environment, and cyber-physical systems.

Min Luo received the Ph.D. degree in Electrical Engineering from Georgia Institute of Technology, Atlanta, GA USA, in 1992. He also held the B.S. and M.S. degrees in 1982 and 1987, respectively in Computer Science. Currently, he is the Head and Chief Architect of the Advanced Networking at Huaweis Shannon (IT) Lab, leading the research and development in Software Defined Networking (SDN) and other future networking initiatives. He served as Chief/Executive Architect for IBM SWGs Strategy and Technology, Global Business Solution Center GCG, Industry Solutions, and Center of Excellence for Enterprise Architecture and SOA for more than 11 years. He also worked as Senior Operations Research Analyst, Senior Manager and Director of Transportation Network Planning and Technologies for two Fortune 500 companies for 7 Years. He is certified and awarded as the Distinguished Lead/Chief Architect from Open Group in 2008. He is an established expert in the field of next negation software defined networking (SDN), enterprise architecture and information systems, whole life cycle software application and product development, business intelligence, and business process optimization. He is also a pioneer and one of the recognized leading experts and educators in Service-oriented architecture (SOA), Model/business-driven architecture and development (MDA-D), and component/object-oriented technologies. He coauthored 2 books, including the pioneering Patterns: Service Oriented Architecture and Web Services in 2004, and published over 20 research papers. As a senior member of IEEE, he has been serving on the organizing committee for IEEEs ICWS, SCC and CC (Cloud Computing) Conferences, chaired sessions, presented several tutorials on SOA and Enterprise Architecture and their best practices and gave lectures at the Service University. He has served as adjunct professors in several USA and Chinese universities since 1996.