Optimal Capacity Sharing of Networks with Multiple Overlays

Zheng Ma*, Jiang Chen*, Yang Richard Yang* and Arvind Krishnamurthy†
*Department of Computer Science
Yale University,New Haven, CT 06511
Email: {zhengma,criver, yry}@cs.yale.edu
†Computer Science and Engineering
University of Washington,Seattle, WA 98195
Email: arvind@cs.washington.edu

Abstract— **Overlay networks have emerged as a generic networking paradigm to improve network performance and construct new applications. Although many overlay algorithms have been proposed lately, they tend to focus on a single overlay, without considering how to share network capacity with other traffic and other overlays. In this paper, we study optimal capacity sharing of network with multiple overlays. We first formulate the problem of optimal capacity sharing of networks with multiple overlays as a nonlinear optimization problem. We show that traditional *flow-level rate controllers* result in sub-optimal sharing results between the different overlays. We design efficient and distributed *overlay flows control* algorithms and demonstrate the effectiveness of our design.**

I. Introduction

Overlay networks are emerging as a new paradigm for constructing new distributed applications and improving network performance. In the last few years, many overlay networks have been proposed, evaluated, and deployed. For instance, many large-scale, overlay-style content-delivery networks such as Akamai have been deployed. Various large-scale testbed networks such as PlanetLab [1] and Emulab [2] are also structured as overlay networks. The traffic generated by peer-to-peer file-sharing applications such as BitTorrent, and VoIP applications such as Skype, is continuously on the rise. In some networks, the traffic generated by such overlays is becoming the dominant traffic.

However, there are serious concerns about the wide deployment of overlay networks. One particular concern is that current overlay algorithms tend to focus on a single overlay, without considering the co-existence of other traffic and other overlays. An overlay network may make resource allocation decisions too aggressively to utilize all available network resources without consideration of others, resulting in sub-optimal resource allocation, unfairness, and/or wild oscillations [3], [4]. Thus, a serious concern is that overlays improve their performance at the expense of others, and wide deployment of overlays may lead to overall degradation of the performance of the entire network.

One way to regulate the behavior of overlays is to implement network congestion control. While many traditional network congestion control schemes have been successfully proposed (*e.g.*, [5], [6], [7], [8]) they work for point-to-point connections as opposed to the many-to-many connectivity seen in the world of overlays. For example, under a widely used framework introduced by Kelly *et al.* in [9], the basic congestion control unit is a single unicast flow from one specific source to a specific destination. However, an overlay network typically consists of many unicast flows belonging to multiple source-destination pairs. If we view each overlay link as a single unicast flow, the whole overlay network could be treated as a collection of end-to-end flows. We call them *flow-level rate controllers* if the flows are controlled individually. If they coordinate with each other, we call them *overlay-level rate controllers*. As we will show in Sec. II, *flow-level rate controllers* may result in unfair and sub-optimal share of network resource between overlays. Thus, a new *overlay-level rate control* scheme is needed to share network resources efficiently, fairly, and distributedly in the emerging world of overlay networks.

In this paper, we present a systematic framework to study the resource allocation and congestion control for networks with adaptive traffic generated by multiple overlays. Specifically, we first formulate the optimal capacity sharing problem among multiple overlays using nonlinear optimization theory. By modeling a traditional end-to-end flow as an overlay with only two nodes, our formulation is more general than existing models and considers both current adaptive point-to-point traffic and general overlay traffic.

A technical challenge we need to address is how to solve the nonlinear optimization problem where an overlay may consist of multiple logical links in a distributed and efficient way. In this paper, inspired by recent advances in optimization based congestion control, we derive general *overlay-level congestion control* algorithms to achieve optimal capacity sharing among overlays using queuing delay dynamics as a feedback from underlay networks. Similar to many delay-based approaches, such as TCP Vegas [7] and FAST TCP [10], our solution can be implemented at each overlay node to adjust the traffic rate in a distributed way.

We demonstrate the effectiveness of our algorithm using a case study of heterogeneous networks where multiple overlays try to find the maximum low from its sender to receiver. We illustrate that overlays can share network capacity more efficiently by implementing our algorithms. Overlays can improve overall network performance by probing for available network resources in a controlled manner, instead of by being unfair to other adaptive flows.

The rest of the paper is organized as follows. In Section II, we present our problem formulation and the sub-optimal results of capacity sharing among overlays. In Section III, we derive a primal-dual algorithm to optimally share network capacity among multiple overlays. We propose a distributed protocol to implement the algorithm. In Section IV, we use overlay maximum flow as a case study to show the effectiveness of our algorithms by simulations. Related work is discussed in Section V and Section VI concludes the paper.

II. Problem Formulation

We begin this section by defining the physical network and multiple overlays on top of it. After that we formulate the optimal capacity sharing of multiple overlays as a constrained nonlinear optimization problem. Then, we illustrate that the resource allocation could be sub-optimal if one uses only the *flow-level rate control* in this scenario.

A. Network and Flow Model

Our underlay *physical network* is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{L}, \mathbf{c})$, where $\mathcal{V}$ is the set of physical nodes, and $\mathcal{L}$ is the set of undirected physical links. We assume each physical link has capacity c_l. Thus $\mathbf{c} = (c_l), l \in \mathcal{L}$ is the vector representation of all link capacities. The physical network resources are shared by multiple overlays.

We represent each *overlay network* O_i as a graph $\mathcal{G}_i = (\mathcal{H}_i, \mathcal{E}_i)$, where $\mathcal{H}_i$ is the set of hosts, and $\mathcal{E}_i$ is the set of directed overlay links. The union of all overlay nodes is $\mathcal{H} = \bigcup_{i=1}^{n} \mathcal{H}_i$ and the union of all overlay links is $\mathcal{E} = \bigcup_{i=1}^{n} \mathcal{E}_i$.

The end-to-end flow rate of each overlay link $e \in \mathcal{E}$ is x_e. If an overlay link belongs to $\mathcal{E}_i$, it is controlled by overlay O_i. We define *overlay flows*, $\mathbf{x}_i = (x_e^i) = (x_{uv}^i), e = (h_{iu}, h_{iv}) \in \mathcal{E}_i$, as a vector of all end-to-end flows controlled by O_i. Generalizing traditional *flow-level rate control*, which regulates the flow rate of a single sender to a single receiver, *overlay flows rate control* adjusts a vector of flow rates among multiple participants. Note that a traditional end-to-end flow is an overlay with just two nodes.

Each overlay link $e \in \mathcal{E}$ corresponds to a physical path consisting of multiple physical links. We use $\mathcal{L}(e) \subseteq \mathcal{L}$ to represent this set of physical links. We also define a $L \times E_i$ routing indicator matrix $\mathbf{A}_i = (A_{le}^i), l \in \mathcal{L}, e \in \mathcal{E}_i$ for each overlay O_i, where $A_{le}^i = 1$ if $l \in \mathcal{L}(e)$ and $A_{le}^i = 0$ otherwise. Each physical link l can be used by multiple overlay links. We define $\mathcal{E}(l) \subseteq \mathcal{E}$ as the set of overlay links that uses l.

When multiple overlays co-exist in the same physical network, the *physical constraints* will place a limit on the combined traffic of *all* overlays. This is where multiple overlays interact with each other. Let $\mathbf{A} = (\mathbf{A}_i)$, $\mathbf{x} = (\mathbf{x}_i)$. The physical capacity constraint can be represented as

$$\mathbf{A} \cdot \mathbf{x} \leq \mathbf{c}. \tag{1}$$

Many overlay networks are application specific. While physical constraints specify the capacity limit on the physical network, application constraints preserve the relationship between different x_e within an overlay. For instance, flow conservation at each overlay node in many overlay applications is one such constraint. Specifically, the flow conservation constraint can be represented as the follows. Let $e = (u, v)$. For overlay O_i, we can define a $H_i \times E_i$ matrix $\mathbf{F}_i = (F_{he}^i), h \in \mathcal{H}_i, e \in \mathcal{E}_i$ as following:

$$F_{he}^i = \begin{cases} 1 & \text{if } h = v \\ -1 & \text{if } h = u \\ 0 & \text{otherwise.} \end{cases}$$

Thus, a compact representation of the flow conservation constraint is

$$\mathbf{F}_i \cdot \mathbf{x}_i = \mathbf{0}. \tag{2}$$

For the general case, $\mathbf{F}_i$ specifies the data relaying relationship in application level. It is determined by application level topology, which is under control of the overlay topology construction algorithms.

Note that in the preceding problem formulation, the same physical node is considered to be different in different overlays. This is also true for overlay links that connect the same pairs of physical nodes. This representation allows different *overlay flows* to use different control algorithms.

Table I summarizes the notations. In the example by Fig.1, there are 6 physical links (L=6). Overlay network O_1 has 5 nodes ($H_1 = 5$) and 5 overlay links ($E_1 = 5$). Overlay network O_2 is a TCP flow from 2 to 5. Assume shortest path routing by number of hops at physical network. Inequality (1) becomes

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{13}^1 \\ x_{32}^1 \\ x_{34}^1 \\ x_{25}^1 \\ x_{45}^1 \\ x_{25}^2 \end{pmatrix} \leq \begin{pmatrix} 2 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{pmatrix}.$$

Equation (2) is application-specific. For instance, in Fig. 1, we assume the objective of O_1 is to find the maximum flow from node 1 to node 5. Since the source and destination node of maximum flow will not have any flow conservation constraints, the equality constraint for O_i is

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_{13}^1 \\ x_{32}^1 \\ x_{34}^1 \\ x_{25}^1 \\ x_{45}^1 \end{pmatrix} = \mathbf{0}.$$

B. Optimal Capacity Sharing of Multiple Overlays

Each overlay can have its own objective. The objective of overlay network O_i can be represented by a *utility function* U_i, which overlay O_i tries to maximize. The input to U_i is an *aggregation function* applied to $\mathbf{x}_i$. We use $f_i(\mathbf{x}_i)$ to represent

TABLE I

NOTATION FOR OVERLAY FLOWS CONTROL FORMULATION.

Notation	Definition
$O_i \in \{1, 2, ..., n\}$	Overlay Network Session i.
$\mathcal{H}_i = \{h_{i1}, h_{i2}, ..., h_{ij}\}, \lvert\mathcal{H}_i\rvert = H_i$	End hosts in session i.
$h \in \mathcal{H} = \bigcup_{i=1}^{n} \mathcal{H}_i$, $\lvert\mathcal{H}\rvert = H$	Set of all overlay nodes.
$\mathcal{E}_i = \{e_{i1}, e_{i2}, ..., e_{ik}\}$	Overlay links in session i.
$e = (u, v) \in \mathcal{E} = \bigcup_{i=1}^{n} \mathcal{E}_i$, $\lvert\mathcal{E}\rvert = E$	Set of all overlay links.
$\mathbf{x}_i = (x_{uv}^i), e = (h_{iu}, h_{iv}) \in \mathcal{E}_i$	Flow rate vector in session i.
$f_i(\mathbf{x}_i) : \mathbf{x}_i \to R$	Aggregation function for session i.
$l \in \mathcal{L} = \{1, .., L\}$	Set of physical links.
$\mathbf{c} = (c_l, l \in \mathcal{L})$	The capacity constraint vector.
u, v, w	Overlay nodes.
$\mathbf{A}_i = (A_{le}^i)_{L \times E_i}$	Routing matrix for session i.
$\mathbf{F}_i = (F_{he}^i)_{H_i \times E_i}$	Application constraint matrix for i.

this aggregation function. Thus, the objective of overlay O_i is to maximize $U_i(f_i(\mathbf{x}_i))$, under physical and application constraints.

Let $\mathbf{F} = (\mathbf{F}_i)$. If the system design objective is to maximize the sum of the utilities of all overlays, we can write the system problem as:

$$\mathbf{P}: \quad \textbf{Maximize:} \quad \sum_{i=1}^{n} U_i(f_i(\mathbf{x_i})) \tag{3}$$

$$\textbf{Subject to:} \quad \mathbf{A} \cdot \mathbf{x} \leq \mathbf{c} \tag{4}$$

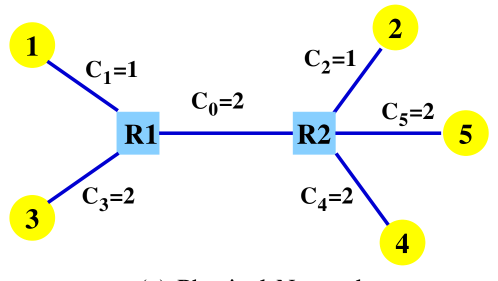
$$\mathbf{F} \cdot \mathbf{x} = \mathbf{0} \tag{5}$$

$$\textbf{over:} \quad x_e \geq 0, \forall e \in \mathcal{E}. \tag{6}$$

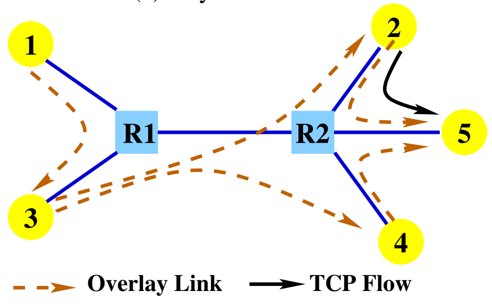
As a special case when all overlays are single end-to-end flows, $f_i(x) = x$, $\mathbf{F}$ does not exist, and the above formulation is reduced to that of Kelly's framework in [9]. U_i and f_i are application specific. We assume that U_i is strictly concave. For f_i, we assume that it is differentiable but does not need to be strictly concave. As an example, $f_i(\mathbf{x}_i) = \sum_{e \in \mathcal{E}_i} x_e$.

Many overlays [11] use periodical probing for available bandwidth and adjusting the flow rates to achieve the overall performance goal. By formulating the overlay's objective in the optimization framework, we can have a better understanding of its behavior when they share the network resource with others.

To solve the system problem $\mathbf{P}$ requires centralized computation and global knowledge of the network including each overlay's utility function, routing and capacity information in the underlay, and flow conservation information of each application. Those make it almost impossible to solve in practice. We will show a distributed algorithm that decouples the computation in a scalable way in Section III.



(a) Physical Network



(b) Overlay Network O_1 co-exists with a TCP flow

Fig. 1. Sub-optimality TCP performance when using only flow level rate control.

C. Sub-optimality of using only flow-level rate control

Before we solve the system problem $\mathbf{P}$, we first show the necessity of solving $\mathbf{P}$. Since we are concerned with rate control and resource sharing, one possibility to do is to ignore the structure of overlays. Thus, each overlay link will be considered as an individual flow. The rate of each overlay link then is determined by traditional transport mechanism. We call such a scheme *flow-level* rate control. We will show that *flow-level* rate control is inefficient for overlays. We illustrate our points using two examples.

1) Example 1: Unfair sharing when overlays interact with TCP: The first example consists of an overlay network O_1 and a competing TCP flow. The physical network is shown in Fig. 1(a). The overlay network and the TCP flow are shown in Fig. 1(b).

Let x_{tcp} be the throughput of the TCP flow. We write down the *overlay flows* of $\mathbf{x}_1$ as

$$\mathbf{x}_1 = (x_{13}^1, x_{32}^1, x_{34}^1, x_{25}^1, x_{45}^1).$$

We assume the objective of O_1 is to find the maximum flow from node 1 and node 5. Thus the *aggregation function*

$$f_1(\mathbf{x}_1) = x_{25}^1 + x_{45}^1.$$

Since TCP is a single flow, $f_{tcp}(x_{tcp}) = x_{tcp}$.

Now we can write down the optimization objectives of both. We consider proportional fairness among overlays. So both overlays will use a log like utility function. We have that

$$U_{tcp}(f_{tcp}(x_{tcp})) = \log(x_{tcp})$$
$$U_1(f_1(\mathbf{x}_1)) = \log(x_{25}^1 + x_{45}^1).$$

Thus any rate allocation which maximizes total utility

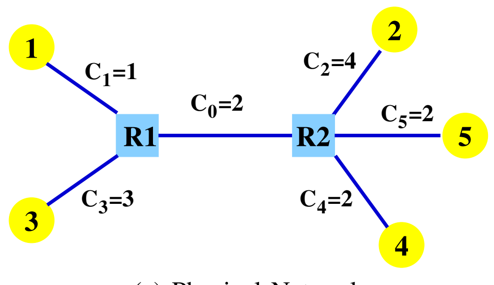
$$U_1 + U_{tcp} = \log(x^1_{25} + x^1_{45}) + \log(x_{tcp})$$

under physical and application constraints achieves system optimum. In particular, $\mathbf{x}_1 = (1, 0, 1, 0, 1), x_{tcp} = 1$ is a system optimal solution with a total utility value being 0.

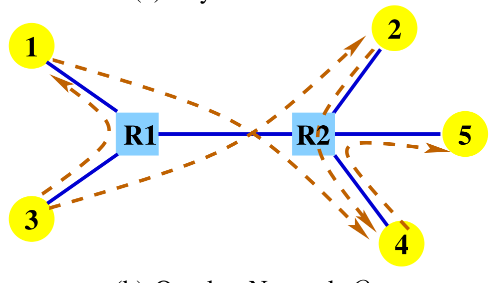
However, if we use only *flow-level* control, then we may reach the following equilibrium: $\mathbf{x}_1 = (1, 1/3, 2/3, 1/3, 2/3)$, and $x_{tcp} = 1/3$, when the network implements flow level fair sharing. This is because when the overlay O_1 and the TCP flow saturate the bottleneck link $c_2 = 1$, each flow will get $1/3$ of the link capacity. No one can further increase its utility because the probing for available bandwidth will return 0 for x^1_{25},x^1_{32} and x_{tcp}. Although overlay O_1 achieves its optimum ($U_1(f_1(\mathbf{x}_1)) = 0$), the TCP flow does not ($U_{tcp}(x_{tcp}) = -0.48$). The total utility is only -0.48.

This simple example shows that we may not be able to achieve system optimum if we use only flow-level rate control. Since overlays generally use more flows than a single TCP flow, they may improve performance at the expense of other TCP flows.

2) Example 2: Sub-optimal capacity sharing of multiple overlays: In our second example, we will show that using *flow-level* rate control may lead to sub-optimal performance for overlays. Thus overlays will have incentives to adopt new *overlay flow control algorithms* to achieve their optimum as well as system optimum.



(a) Physical Network



(b) Overlay Network O_2

Fig. 2. Sub-optimality overlay performance when using flow-level rate control only.

The example consists of two overlays O_1 and O_2. The physical network is shown in Fig. 2(a). The topology of O_1 is the same as the preceding example, and the topology of O_2 is shown in Fig 2(b). The *overlay flows* of $\mathbf{x}_2$ is

$$\mathbf{x}_2 = (x^2_{31}, x^2_{32}, x^2_{14}, x^2_{24}, x^2_{45}).$$

The objective of O_1 is still to find the maximum flow from node 1 to node 5. The objective of O_2 is to find maximum flow from node 3 to node 5. So the aggregation functions are $f_1(x_1) = x^1_{25} + x^1_{45}$, $f_2(x_2) = x^2_{45}$.

Consider the case when both overlays use linear utility functions. Thus

$$U_1(f_1(\mathbf{x}_1)) = x^1_{25} + x^1_{45}$$
$$U_2(f_2(\mathbf{x}_2)) = x^2_{45}.$$

The system optimization objective is

$$U_1 + U_2 = x^1_{25} + x^1_{45} + x^2_{45}.$$

We can see that $\mathbf{x}_1 = (1, 1, 0, 1, 0), \mathbf{x}_2 = (0, 1, 0, 1, 1)$ is a system optimal solution with total utility value being 2. However, if we use only fair sharing at flow level, then we may have the following sub-optimal equilibrium, in which O_1 and O_2 saturate the capacity of c_1 and c_4:
Equilibrium

$$\begin{aligned} \mathbf{x}_1 &= (1/3, 0, 1/3, 0, 1/3) \\ U_1(f_1(\mathbf{x}_1)) &= 1/3. \end{aligned}$$

Bottleneck: c_1,c_4

$$\begin{aligned} \mathbf{x}_2 &= (1/3, 1/3, 1/3, 1/3, 2/3) \\ U_2(f_2(\mathbf{x}_2)) &= 2/3. \end{aligned}$$

At this equilibrium, both overlays get lower utilities compared with system optimum. The total system utility value is only 1. The example shows that if the overlay O_1 or O_2 uses only flow-level control it may reach sub-optimal solutions as it does not take into account the presence of other overlays.

Given the preceding two examples, a natural question to ask then is whether it is possible to design an *overlay flows* control algorithm to make overlays friendly to TCP flows and to share network resource optimally among multiple overlays. We address both questions in the next section.

III. Algorithm Design for Optimal Capacity Sharing of Overlays

In this section, we show a systematic framework to design algorithms for solving the problem **P** formulated in the preceding section. To be scalable, we focus on distributed algorithms. The distributed algorithms should have low communications overhead and shouldn not require explicit support from the underlay network. Inspired by recent developments in optimization based Internet congestion control [6], [8], [12], we design a distributed *overlay flows* control algorithm to achieve the above goals. Since different overlays may be formulated differently in **P** (*e.g.*, with different f_i and application constraints), our presentation is a generic framework. In the next section, we will present a concrete instance.

Below we first discuss the technical challenges. Then we present primal-dual algorithms with convergence guarantees.

A. Difficulties Due to Lack of Strict Concavity

In problem $\mathbf{P}$, although U_i, the utility functions of overlays, are strictly concave, the final objective function $U_i(f_i(\mathbf{x}))$ may not be strictly concave on $\mathbf{x}$ if f_i is not strictly concave on $\mathbf{x}$. This would pose problems since the dual of the problem may not be differentiable at every point [13]. Some existing techniques based on duality (*e.g.*, [6], [12]) can only deal with strictly concave objective functions.

We address this problem by making the objective function in $\mathbf{P}$ strictly concave. In particular, we use the Proximal Minimization method ([13], Chapter 8) which enforces strict concavity by adding a quadratic term to the objective function and then iterates to eliminate the effects of the term. Specifically, we introduce the vector $\mathbf{b} = (b_e), \forall e \in \mathcal{E}, b_e \geq 0$ and define $U_e(x_e) = -\frac{1}{2\kappa}(x_e - b_e)^2$, where κ is a positive scalar parameter. Now consider the optimization problem $\mathbf{P1}$ under the same constraints in (4)-(6), with objective function as

$$\begin{aligned} \mathbf{P1}: \quad \textbf{Maximize:} \quad & \sum_{i=1}^{n} U_i(f_i(\mathbf{x})) - \sum_{i=1}^{n} \sum_{e \in \mathcal{E}_i} \frac{1}{2\kappa}(x_e - b_e)^2 \\ \textbf{Subject to:} \quad & \mathbf{A} \cdot \mathbf{x} \leq \mathbf{c} \\ & \mathbf{F} \cdot \mathbf{x} = \mathbf{0} \\ \textbf{over:} \quad & x_e \geq 0, \quad \forall e \in \mathcal{E}. \end{aligned} \tag{7}$$

If $\mathbf{b} = \mathbf{x}^*$, where $\mathbf{x}^*$ is the optimal solution to $\mathbf{P}$, then the optimal solution to $\mathbf{P1}$ is optimal to $\mathbf{P}$ as well. Consider the iterative *proximal minimization algorithm* in Fig. 3, where $\mathbf{x}^{(0)}$ is any feasible point and $\mathbf{b}^{(0)} = \mathbf{x}^{(0)}$.

```
for $n = 1, 2, ...$ do:
    step 1. Solve problem P1, with $\mathbf{b} = \mathbf{b}^{(n-1)}$,
            obtain the optimal solution $\mathbf{x}^{(n)}$.
    step 2. Set $\mathbf{b}^{(n)} = \mathbf{x}^{(n)}$.
```

Fig. 3. Proximal Minimization Algorithm .

We show that the algorithm in Fig. 3 converges to the optimal solution of problem $\mathbf{P}$ as $n \to \infty$ ([14], pp 233). Note that the preceding algorithm is a two-level optimization procedure. Step 2 in the algorithm is executed only when the procedure of solving Step 1 converges. When κ is large, the objective function of $\mathbf{P1}$ is close to the one in $\mathbf{P}$. Step 1 may converge to a value close to the optimum with less iterations. More detailed discussions on choosing κ and step size could be found in ([14], pp234).

B. Problem Decomposition and Algorithm Design

The preceding algorithm requires us to solve $\mathbf{P1}$. The Lagrangian of $\mathbf{P1}$ can be derived as follows ($\mathbf{b}$ is constant in each iteration):

$$\begin{aligned} & L(\mathbf{x}, \boldsymbol{p}, \boldsymbol{\mu}) \\ = & \sum_{i=1}^{n} \underbrace{\{U_i(f_i(\mathbf{x})) - \sum_{e \in \mathcal{E}_i} \frac{1}{2\kappa}(x_e - b_e)^2\}}_{\Psi_i(\mathbf{x}_i)} \\ & - \boldsymbol{p}(\mathbf{A} \cdot \mathbf{x} - \mathbf{c}) - \boldsymbol{\mu}(\mathbf{F} \cdot \mathbf{x}) \\ = & \sum_{i=1}^{n} \Psi_i(\mathbf{x}_i) - \sum_{l \in \mathcal{L}} p_l (\sum_{e \in \mathcal{E}} A_{le} x_e - c_l) - \boldsymbol{\mu}(\mathbf{F} \cdot \mathbf{x}) \\ = & \sum_{i=1}^{n} \Psi_i(\mathbf{x}_i) - \sum_{e \in \mathcal{E}} x_e \sum_{l \in \mathcal{L}} p_l A_{le} - \sum_{e \in \mathcal{E}} x_e \sum_{h \in \mathcal{H}} \mu_h F_{he} \\ & + \sum_{l \in \mathcal{L}} p_l c_l \\ = & \sum_{i=1}^{n} \Psi_i(\mathbf{x}_i) - \sum_{e \in \mathcal{E}} x_e q_e - \sum_{e \in \mathcal{E}} x_e \lambda_e + \sum_{l \in \mathcal{L}} p_l c_l \\ = & \sum_{i=1}^{n} (\Psi_i(\mathbf{x}_i) - \sum_{e \in \mathcal{E}_i} (q_e + \lambda_e) x_e) + \sum_{l \in \mathcal{L}} p_l c_l \\ = & \sum_{i=1}^{n} \Phi_i(\mathbf{x}_i) + \sum_{l \in \mathcal{L}} p_l c_l. \end{aligned} \tag{8}$$

Here $\boldsymbol{p} = (p_l, l \in \mathcal{L})$ represents the price vector of the physical links. Let

$$q_e = \sum_{l \in \mathcal{L}} p_l A_{le} = \sum_{l \in \mathcal{L}(e)} p_l$$

be the price of an overlay link. Let

$$\lambda_e = \sum_{h \in \mathcal{H}} \mu_h F_{he}$$

be the application specified price. The interpretation of the prices will vary in different applications depending on the format of $\mathbf{F}$.

Since the objective function of $\mathbf{P1}$ is a strictly concave function of $\boldsymbol{x}$, there exist a unique maximizing solution in Equation (7), say $\boldsymbol{x}^*$, and Lagrange multipliers $\boldsymbol{p}^*$ and $\boldsymbol{\mu}^*$, which satisfy the following Karush-Kuhn-Tucker conditions.

$$\begin{aligned} \frac{\partial L}{\partial x_e} &= 0 \Rightarrow \\ \forall i, e \in \mathcal{E}_i, \frac{\partial \Psi_i(\mathbf{x}_i)}{\partial x_e} - (q_e^* + \lambda_e^*) &= 0 \Rightarrow \\ U_i'(f_i(\mathbf{x}^*)) \cdot \frac{\partial f_i}{\partial x_e} - \frac{1}{\kappa}(x_e^* - b_e) - (q_e^* + \lambda_e^*) &= 0; \\ \boldsymbol{p}^*(\mathbf{A} \cdot \mathbf{x}^* - \mathbf{c}) &= \mathbf{0} \\ \mathbf{A} \cdot \mathbf{x}^* &\leq \mathbf{c} \\ \mathbf{F} \cdot \mathbf{x}^* &= \mathbf{0} \\ \forall l \in \mathcal{L} \quad p_l^* &\geq \mathbf{0}. \end{aligned} \tag{9}$$

To find the unique solution to $\mathbf{P1}$ in a decentralized fashion, we can use either dual algorithms like [6], [12] or primal-dual algorithms like [8], [15], [16]. Here, we consider primal-dual algorithms, which solve convex programming problems by computing the primal and dual variables simultaneously and moving towards the saddle point at each step.

Let $\mathbf{y} = \mathbf{A} \cdot \mathbf{x} = (y_l), l \in \mathcal{L}$, $\mathbf{z} = \mathbf{F} \cdot \mathbf{x} = (z_h), h \in \{1, 2, ..., H\}$. Also, define $(y)_x^+$ for $x \geq 0$ as follows

$$(y)_x^+ = \begin{cases} y & if \;\; x > 0 \\ \max(y, 0) & if \;\; x = 0. \end{cases} \tag{10}$$

The primal-dual algorithm is described in Fig. 4. Here,

$$\begin{aligned} &\forall e \in \mathcal{E} \quad \dot{x}_e = k_e(x_e)(U_i'(f_i(\mathbf{x})) \cdot \tfrac{\partial f_i}{\partial x_e} \\ &\qquad\qquad -\tfrac{1}{\kappa}(x_e - b_e) - (q_e + \lambda_e)). \\ &\forall l \in \mathcal{L} \quad \dot{p_l} = h_l(p_l)(y_l - c_l)^+_{p_l}. \\ &\forall h \in \mathcal{H} \quad \dot{\mu}_h = m_h(\mu_h)(z_h). \end{aligned}$$

Fig. 4. Primal-Dual Algorithm .

$k_e(x_e) > 0$, $h_l(p_l) > 0$, and $m_h(\mu_h) > 0$ are non-decreasing continuous functions. We can see that in this algorithm, the dual variable p_l is determined by the queuing process at physical links. μ_h could be updated by *overlay flows* control algorithms at each overlay node by comparing its total incoming flow rate with total outgoing flow rate. x_e will be updated by each flow controller using the shared information within its overlay (the aggregation function) and the feedback price information from q_e and λ_e. All those updates are fully distributed like the updates discussed in [5], [6]. In addition, we have the following theorem:

Theorem 3.1: The algorithm presented in Table 4 is globally asymptotically stable.

Proof: Please see appendix .

The global, asymptotic stability of the algorithm guarantees that starting from any initial condition, the primal-dual algorithm will converge to the unique solution of problem **P1**. In the next section, we will use overlay maximum flow problem as a case study to show a detailed protocol implementation of the above algorithm.

IV. Case Study: Overlay Maximum Flow

In this section, we use overlay maximum flow as a case study to illustrate how to apply the algorithms in Section III to a specific problem. The overlay maximum flow problem was also studied in [11] to demonstrate the importance of correlated link capacity constraints on overlay quality. We will use the example in Fig.1 and Fig.2 to show the formulation and the resulting algorithm design.

A. Overlay Maximum Flow Problem Formulation

We continue using the notation in Table I to formulate the problem. We use h_{is} and h_{ir} to represent the sender and receiver in each overlay O_i. Overlay maximum flow problem is to find the maximum flow rate between h_{is} and h_{ir}.

For each node in an overlay network, we have the following flow conservation constraints.

$$\forall i, \forall v \in \mathcal{H}, v \neq h_{ir}, v \neq h_{is}$$

$$\sum_{(u,v)\in\mathcal{E}_i} x^i_{uv} = \sum_{(v,w)\in\mathcal{E}_i} x^i_{vw} \tag{11}$$

$$\forall i, \forall v \in \mathcal{H}, h_{ir}, \sum_{(h_{ir},v)\in\mathcal{E}_i} x^i_{rv} = 0 \tag{12}$$

$$\forall i, \forall u \in \mathcal{H}, h_{is}, \sum_{(u,h_{is})\in\mathcal{E}_i} x^i_{us} = 0. \tag{13}$$

In this problem, $\forall u, v \in \mathcal{H}, x_{uv} \geq 0$, so from equation (12) and (13), we know that $\forall h_{ir}, h_{is}, v \in \mathcal{H}, x^i_{rv} = 0, x^i_{vs} = 0$. For simplicity, we remove such variables from $\mathbf{x}$ and remove constraints in equation (12) and (13) as well. Let $e = (u, v)$, the flow conservation constraints matrix $\mathbf{F}$ is:

$$F^i_{he} = \begin{cases} 1 & \text{if } h = v, h \neq h_{ir} \\ -1 & \text{if } h = u, h \neq h_{is} \\ 0 & \text{otherwise.} \end{cases}$$

Then all the flow conservation constraints can be written as $\mathbf{F} \cdot \mathbf{x} = \mathbf{0}$. We define the *aggregation function* f_i for overlay maximum flow problem as

$$f_i(\mathbf{x}) = \sum_{(u,h_{ir})\in\mathcal{E}_i} x^i_{ur} = \mathbf{M}_i \cdot \mathbf{x}. \tag{14}$$

Here, $\mathbf{M}_i$ is a $1 \times E$ row vector where $\mathbf{M}_i(e) = 1$ when $e = (u, h_{ir})$, otherwise $\mathbf{M}_i(e) = 0$. Then the objective of optimal capacity sharing among multiple overlays is

$$\sum_{i=1}^{n} U_i(\mathbf{M}_i \cdot \mathbf{x}).$$

We assume each overlay network has a strictly concave utility function U_i.

In the physical network of Fig. 2(a), there are two overlay networks ($n = 2$). The total number of overlay nodes is 10 ($H = 10$), and there are six physical links ($L = 6$). Assume $h_{1s} = h_{11}, h_{1r} = h_{15}$, $h_{2s} = h_{23}, h_{2r} = h_{25}$. Then inequality (4) becomes

$$\begin{pmatrix} 0&1&1&0&0&0&1&1&0&0 \\ 1&0&0&0&0&1&0&1&0&0 \\ 0&1&0&1&0&0&1&0&1&0 \\ 1&1&1&0&0&1&1&0&0&0 \\ 0&0&1&0&1&0&1&0&1&0 \\ 0&0&1&1&0&0&0&1&1&1 \end{pmatrix} \cdot \begin{pmatrix} x^1_{13} \\ x^1_{32} \\ x^1_{34} \\ x^1_{25} \\ x^1_{45} \\ x^2_{31} \\ x^2_{32} \\ x^2_{14} \\ x^2_{24} \\ x^2_{45} \end{pmatrix} \leq \begin{pmatrix} 2 \\ 1 \\ 4 \\ 3 \\ 2 \\ 2 \end{pmatrix}.$$

Matrix $\mathbf{F}$ in eqaution (5) becomes

$$\begin{pmatrix} 0&0&0&0&0&0&0&0&0&0 \\ 1&-1&-1&0&0&0&0&0&0&0 \\ 0&1&0&-1&0&0&0&0&0&0 \\ 0&0&1&0&-1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&-1&0&0 \\ 0&0&0&0&0&0&1&0&-1&0 \\ 0&0&0&0&0&0&0&1&1&-1 \\ 0&0&0&0&0&0&0&0&0&0 \end{pmatrix}.$$

In this example,

$$M_1 = (0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0)$$

$$f_1(\mathbf{x}) = x^1_{25} + x^1_{45}.$$

Also,

$$M_2 = (0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1)$$
$$f_2(\mathbf{x}) = x_{45}^2.$$

Let $q_e = \sum\limits_{l \in \mathcal{L}(e)} p_l$, $\lambda_e = \sum\limits_{h \in \mathcal{H}_i} \mu_h F_{he} = \mu_v - \mu_u$ when $e = (u, v)$, assume $\forall i, \mu_{h_{is}} = 0, \mu_{h_{ir}} = 0$.

B. Protocol Design and Implementation

With the preceding formulation, the *overlay flows* control for the overlay maximum flow problem can be implemented using the proximal minimization and primal-dual algorithm. In this section, we describe the assumptions, measurements and messages exchanged in implementing them in practice. We focus on the requirements of the primal-dual algorithm.

First, we assume that a flow rate x_e is controlled and adjusted by the end host, denoted as the flow owner, O_e. The flow rate adaptation could be *receiver based* or *sender based*. We use *sender based* approach similar to TCP: *try and backoff* from the sender.

Since the algorithm is designed for overlay applications, we make *no assumptions* on the knowledge of the physical network. The end hosts do not know the physical topology and capacity information, *i.e.*, $\mathbf{A}$ and $\mathbf{c}$ in equation (1). Instead, the overlays use natural accumulative path feedback metrics such as queuing delay dynamics from the underlay network as an indication of network congestion price. In particular, end hosts measures the end to end delay of each virtual link periodically and thus infer the queuing delay on it. This can be done in a way similar to TCP Vegas [7] and FAST [10]. Note that the information between the same pair of hosts can be shared among different overlays who are using it. This can save the measurement overhead.

We have two types of price: *network price* p_l at each physical link; *node price* μ_h at each overlay node, which indicates the flow conservation violation at that node. Each flow owner will try to send the traffic at a certain rate and then observe the resulting prices of those two. Then the flow owner will update the flow rate based on the received prices and current sending rate. The protocol is presented in Fig. 5.

C. Simulation Results

We implement the algorithm in Fig. 5 and study its effectiveness by extensive simulations using example topologies in Fig. 1 and Fig. 2 and topologies generated by the BRITE [17] topology generator. Here we show the simulation results for the two examples discussed in Section II-C. The utility function we used in the simulation is $u(x) = \log(x)$ for all users. The results are summarized as follows.

For the first example in Section II, as we can see in Fig. 6, the TCP flow converges to its optimal throughput 1 within 5 seconds. O_1 converges to its optimal maximum flow rate 1 with 10 seconds. Because *overlay flows* control need to adjust the flow rates on all of its links in a distributed manner, it generally converges slower than point-to-point flows. The time-varying values of the link prices and relay prices are plotted in Fig. 6

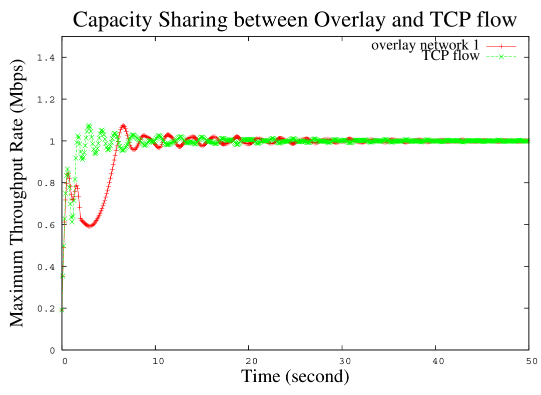
```
At time $t = 1, 2, ...$

**Link Price Update** (by link $l$):
1. Receive flow rate $x_e(t)$ from all overlay flows $e \in \mathcal{E}(l)$.
2. Update price:
    $p_l(t+1) = [p_l(t) + \gamma(\sum_{e \in \mathcal{E}(l)} x_e(t) - c_l)]^+$.
3. Send $p_l(t+1)$ to all overlay unicast flows $e \in \mathcal{E}(l)$
The above process is approximated by queuing process
at physical network.
The $p_l$ is the queuing delay at each physical link.

**Node Price Update** (by overlay node $h$):
1. Receive flow rate $x_e(t)$ from all adjacent links in its session.
2.Update price:
    $\mu_h(t+1) = \mu_h(t) + \gamma(\sum_{e=(u,h)} x_e(t) - \sum_{e=(h,v)} x_e(t))$.
3. Send the updated price $\mu_h(t+1)$ to the senders who is
    connected to this node.

**Flow Rate Adaptation**
(by each overlay link $e = (u, v)$'s sender $O_e$):
1. Measure $q_e(t) = \sum_{l \in \mathcal{L}(e)} p_l(t)$.
    This is the end to end queuing delay at the overlay link $e$.
2. Receive node prices $\mu_h(t)$ from node $u$ and $v$.
3. Receive the aggregated flow rate $M_i(\mathbf{x}(t))$
    at the overlay receiver.
4. Calculate:
    $\lambda_e(t) = \mu_v(t) - \mu_u(t)$ .
5.Adjust rate:
    $x_e(t+1) = k_e(x_e(t))(U_i'(M_i(\mathbf{x}(t))) \cdot \frac{\partial f_i}{\partial x_e}$ .
        $-\frac{1}{\kappa}(x_e(t) - b_e) - (q_e(t) + \lambda_e(t)))$ .
6. Send $x_e(t+1)$ to all links $l \in \mathcal{L}(e)$ and all neighbors.
```

Fig. 5. Protocol for finding maximum-flow in overlay networks.

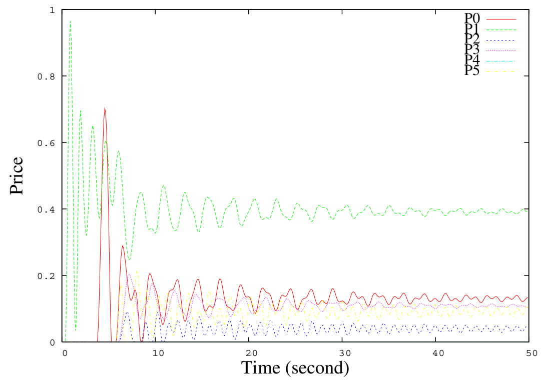
For the second example, from Fig. 7, we can see both overlays are able to converge to the system optimum with maximum flow rate 1 for each within 10 seconds. By using an algorithm which considers other flows in the network, both overlays avoid the sub-optimal equilibria.

From these examples, we can see that the primal-dual algorithm presented in Fig. 5 achieves optimality by requiring both unicast flows and overlay traffic to cooperate. The algorithm converges to the optimal rate starting from different initial values. One could consider our proposed framework as a generalization of protocol compliance requirements, such as *TCP friendliness*. We also note that our framework solves the system optimization problem effectively without requiring explicit support from underlay routers.

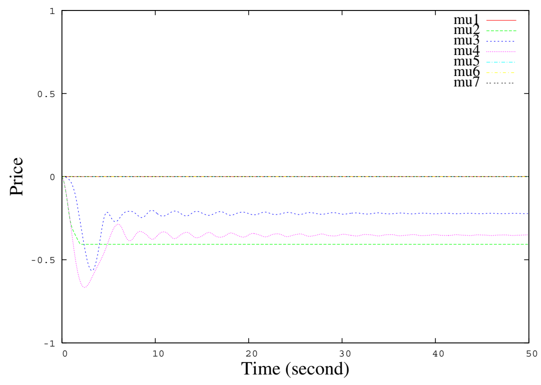
We also study the convergence of the flow rate in a dynamic environment. In example 1, we add more TCP flows between node 2 and 5 at different time and observe the reaction of TCP flow 1. The result is shown in Fig. 8. We can see that our algorithm can react to the network changes quickly. The rate of TCP flow 1 converges to the new optimal rate within 3 seconds when new members join in the physical network.



(a) Maximal flow rate



(b) Link price for all the physical links



(c) Node price for all overlay nodes

Fig. 6. Simulation result for example 1.

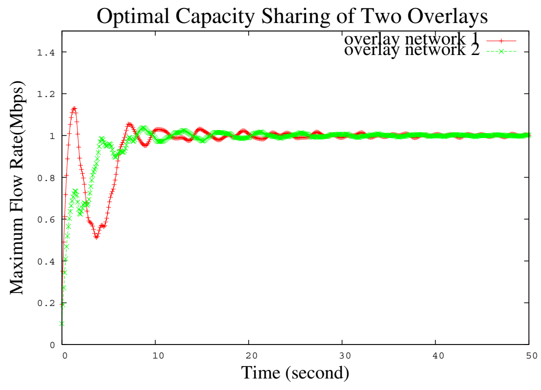


Fig. 7. Simulation result for example 2.

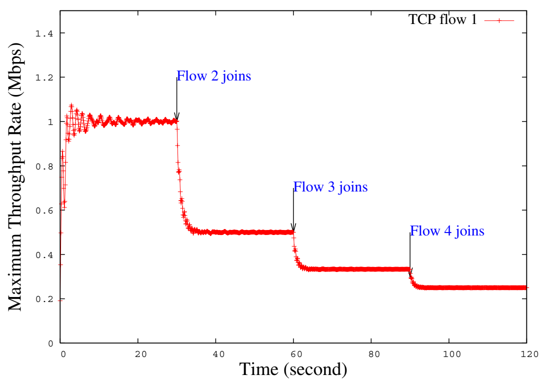


Fig. 8. Convergence of flow rate in a dynamic environment.

V. Related Work

Overlay networks have been used to improve reliability (e.g., [18]) and provide better QoS support (e.g., [19]). As self-organizing networks, they provide opportunities for end users to explore all available resources in a network. Many overlay algorithms have been proposed lately. However, they tend to focus on designing protocols for a single overlay network without considering sharing network resource with others.

The aggressive use of network resources by overlays raises concerns to Internet users, designers and researchers. In a recent work [3], Jiang *et. al* propose the concept of overlay optimal routing, as a contrast to selfish routing [20]. In particular, they study the "interactions" among multiple overlays. They show that when overlays optimize for performance in a selfish manner, the resulting equilibrium is inefficient and unfair. Our work can be regarded as a study of what happens when the overlay networks play by the rules; we show that global optimality can be achieved if each overlay is well-behaved and reacts to feedback from the physical network.

Co-existing overlays have also been studied recently by Keralapura *et al.* in [4]. They investigate potential causes of race conditions and oscillations. Overlapping routes and periodic path probing processes are some of the causes. Coordination among overlays could alleviate this problem. In our study, we realize that overlays interact with each other at saturated physical links. Thus congestion control can be used to achieve the resource allocation goal and potentially help to reduce oscillations.

Our work is also inspired by the study of correlations of overlay links [11]. In particular, Zhu *et al.* show that even a single overlay can benefit from finding the correlation among its links due to physical constraints, which means overlays not only interact with each other at bottleneck links but also have self-interference at those links. Our design addresses both issues.

Optimization based rate control (*e.g.*, [21], [6], [7], [22]) provides us a powerful tool to solve system optimization problem in an efficient and distributed way. In our model, each overlay network is a user, which extends the traditional flow based formulation.

VI. Conclusion

In this paper, we study optimal capacity sharing of multiple overlays when each overlay network maximizes its own utility function. We first show that using traditional *flow-level* rate control may not achieve system-wide optimality. Then we provide a framework to solve the problem systematically. We use congestion control mechanisms to allocate resources so that overlays can interact with other adaptive traffic, generated by unicast TCP flows or other overlays, in a friendly way. We generalize traditional point-to-point congestion control mechanisms to work in multi-point, multi-overlay settings. We then use our proposed framework to study the overlay maximum flow problem and provide simulation results to show the effectiveness of our distributed algorithm. We hope this work can lead to further discussions on how to model and control the interactions among overlays, before overlay networks gain further traction in the Internet.

VII. Acknowledgement

We thank Hao Wang and Haiyong Xie for their valuable discussions. We also thank the anonymous reviewers for their helpful comments and suggestions. Zheng Ma is supported by NSF grants ANI-0238038 and CNS-0435201. Yang Richard Yang is supported in part by NSF grants ANI-0238038 and CNS-0435201.

Appendix
Proof of the Stability of Primal-Dual Algorithm

Proof: We prove the stability of the primal-dual algorithm in Table 4 by using Lyapunov stability theory. For more details on this, we refer interested readers to [8, Section 3.10]. Let's assume the unique solution of the optimization problem **P1** is $(\boldsymbol{x}^*, \boldsymbol{p}^*, \boldsymbol{\mu}^*)$. It's easy to verify that it is an equilibrium point of the primal-dual algorithm. We now prove that this point is globally asymptotically stable. We will construct a continuously differentiable function $V(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu})$ such that

$$V(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) > 0, \forall\ (\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) \neq (\boldsymbol{x}^*, \boldsymbol{p}^*, \boldsymbol{\mu}^*),$$

and also

$$\dot{V}(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) \leq 0, \forall\ (\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) \tag{15}$$

$$\dot{V}(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) < 0, \forall\ (\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) \neq (\boldsymbol{x}^*, \boldsymbol{p}^*, \boldsymbol{\mu}^*) \tag{16}$$

$$V(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) \rightarrow \infty, when\ \|(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu})\| \rightarrow \infty. \tag{17}$$

Consider the following function as a candidate for the Lyapunov function:

$$\begin{aligned} V(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) =& \sum_{i=1}^{n} \sum_{e \in \mathcal{E}_i} \int_{x_e^*}^{x_e} \frac{1}{k_e(\alpha)} (\alpha - x_e^*) d\alpha \\ &+ \sum_{l \in \mathcal{L}} \int_{p_l^*}^{p_l} \frac{1}{h_l(\beta)} (\beta - p_l^*) d\beta \\ &+ \sum_{h \in \mathcal{H}} \int_{\mu_h^*}^{\mu_h} \frac{1}{m_h(\gamma)} (\gamma - \mu_h^*) d\gamma. \end{aligned} \tag{18}$$

Note that $V(\boldsymbol{x}^*, \boldsymbol{p}^*, \boldsymbol{\mu}^*) = 0$. Since $k_e(\alpha) > 0$, when $x_e \neq x_e^*$, we have the following

$$\int_{x_e^*}^{x_e} \frac{1}{k_e(\alpha)} (\alpha - x_e^*) d\alpha > 0.$$

This argument can be extended to other terms as well. So when $(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) \neq (\boldsymbol{x}^*, \boldsymbol{p}^*, \boldsymbol{\mu}^*)$, we have $V(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu}) > 0$ and unbounded when $\|(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\mu})\| \rightarrow \infty$. We only need to show Eq. 15 and Eq. 16 are true.

Now, using the primal-dual algorithm in Table 4, we have

$$\begin{aligned} \dot{V} =& \sum_{i=1}^{n} \sum_{e \in \mathcal{E}_i} (\frac{\partial \Psi_i(\boldsymbol{x})}{\partial x_e} - q_e - \lambda_e) \cdot (x_e - x_e^*) \\ &+ \sum_{l \in \mathcal{L}} (p_l - p_l^*)(y_l - c_l)_{p_l}^+ \\ &+ \sum_{h \in \mathcal{H}} (\mu_h - \mu_h^*) \cdot z_h. \end{aligned} \tag{19}$$

Note that

$$(p_l - p_l^*)(y_l - c_l)_{p_l}^+ \leq (p_l - p_l^*)(y_l - c_l). \tag{20}$$

This is true because the inequality is an equality if either $p_l > 0$ or $y_l - c_l > 0$. When $p_l = 0$ and $y - c_l \leq 0$, $(y_l - c_l)_{p_l}^+ = 0$ and since $p_l^* \geq 0$, $0 \leq (p_l - p_l^*)(y_l - c_l)$.
Therefore,

$$\begin{aligned} \dot{V} \leq& \sum_{i=1}^{n} \sum_{e \in \mathcal{E}_i} (\frac{\partial \Psi_i(\boldsymbol{x})}{\partial x_e} - q_e - \lambda_e) \cdot (x_e - x_e^*) \\ &+ \sum_{l \in \mathcal{L}} (p_l - p_l^*)(y_l - c_l) \\ &+ \sum_{h \in \mathcal{H}} (\mu_h - \mu_h^*) \cdot z_h \\ =& \sum_{i=1}^{n} \sum_{e \in \mathcal{E}_i} (\frac{\partial \Psi_i(\boldsymbol{x})}{\partial x_e} - q_e^* - \lambda_e^*) \cdot (x_e - x_e^*) \\ &+ \sum_{l \in \mathcal{L}} (p_l - p_l^*)(y_l^* - c_l) \\ &+ \sum_{h \in \mathcal{H}} (\mu_h - \mu_h^*) \cdot z_h^* \\ &+ \sum_{i=1}^{n} \sum_{e \in \mathcal{E}_i} (-q_e - \lambda_e + q_e^* + \lambda_e^*) \cdot (x_e - x_e^*) \\ &+ \sum_{l \in \mathcal{L}} (p_l - p_l^*)(y_l - y_l^*) \\ &+ \sum_{h \in \mathcal{H}} (\mu_h - \mu_h^*) \cdot (z_h - z_h^*). \end{aligned} \tag{21}$$

From the Karush-Kuhn-Tucker conditions in Eq. 9, we have

$$\begin{aligned} \dot{V} \leq& \sum_{i=1}^{n} \sum_{e \in \mathcal{E}_i} (\frac{\partial \Psi_i(\boldsymbol{x})}{\partial x_e} - \frac{\partial \Psi_i(\boldsymbol{x}^*)}{\partial x_e}) \cdot (x_e - x_e^*) \\ &+ \sum_{l \in \mathcal{L}} p_l (y_l^* - c_l) + 0 \\ &- \sum_{i=1}^{n} \sum_{e \in \mathcal{E}_i} (q_e - q_e^*) \cdot (x_e - x_e^*) \end{aligned}$$

$$
\begin{aligned}
&-\sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i}(\lambda_e-\lambda_e^*)\cdot(x_e-x_e^*)\\
&+\sum_{l\in\mathcal{L}}(p_l-p_l^*)(y_l-y_l^*)\\
&+\sum_{h\in\mathcal{H}}(\mu_h-\mu_h^*)\cdot(z_h-z_h^*)
\end{aligned}
\tag{22}
$$

For $q_e, x_e, y_l, \lambda_e, p_l, \mu_h, z_h$, we have the following relationship

$$
\begin{aligned}
q_e &= \sum_{l\in\mathcal{L}} p_l A_{le} = \sum_{l\in\mathcal{L}(e)} p_l\\
y_l &= \sum_{e\in\mathcal{E}} A_{le}x_e = \sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i} A_{le}x_e\\
\lambda_e &= \sum_{h\in\mathcal{H}} \mu_h F_{he}\\
z_h &= \sum_{e\in\mathcal{E}} F_{he}x_e = \sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i} F_{he}x_e.
\end{aligned}
$$

So

$$
\begin{aligned}
\sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i} q_e\cdot x_e &= \sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i}\sum_{l\in\mathcal{L}} p_l A_{le}x_e\\
&= \sum_{l\in\mathcal{L}} p_l \sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i} A_{le}x_e = \sum_{l\in\mathcal{L}} p_l y_l.
\end{aligned}
\tag{23}
$$

$$
\begin{aligned}
\sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i} \lambda_e\cdot x_e &= \sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i}\sum_{h\in\mathcal{H}} \mu_h F_{he}x_e\\
&= \sum_{h\in\mathcal{H}} \mu_h \sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i} F_{he}x_e = \sum_{h\in\mathcal{H}} \mu_h z_h.
\end{aligned}
\tag{24}
$$

With the above equations, Eq. 22 becomes

$$
\begin{aligned}
\dot{V} \le &\sum_{i=1}^{n}\sum_{e\in\mathcal{E}_i}\left(\frac{\partial\Psi_i(\boldsymbol{x})}{\partial x_e}-\frac{\partial\Psi_i(\boldsymbol{x}^*)}{\partial x_e}\right)\cdot(x_e-x_e^*)\\
&+\sum_{l\in\mathcal{L}} p_l(y_l^*-c_l).
\end{aligned}
\tag{25}
$$

Because of the strict concavity of $\Psi_i(\boldsymbol{x})$, when $x_e\uparrow$, $\frac{\partial\Psi_i(\boldsymbol{x})}{\partial x_e}\downarrow$. So

$$
\dot{V} \le \sum_{l\in\mathcal{L}} p_l(y_l^*-c_l), \tag{26}
$$

with equality if and only if $\boldsymbol{x}=\boldsymbol{x}^*$. So it follows that $\dot{V}\le 0$ for all $p_l\ge 0$ since $y_l^*-c_l\le 0$. Further, $\dot{V}=0$ only when $\boldsymbol{x}=\boldsymbol{x}^*$ and for each link l, either $p_l=p_l^*$ or $y_l^*=c_l$. Thus, it follows by the theory of Lyapunov stability theory that the primal-dual algorithm is globally asymptotically stable. ■

REFERENCES

[1] "PlanetLab," http://www.planet-lab.org.

[2] "Emulab," http://www.emulab.net/.

[3] W. Jiang, J. C. S. Lui, and D.-M. Chiu, "Interaction of overlay networks: properties and implications," *SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 2, pp. 27–29, 2005.

[4] R. Keralapura, C.-N. Chuah, N. Taft, and G. QIannacco, "Can coexisting overlays inadvertently step on each other?" in *Proceedings of 13th IEEE International Conference on Network Procotols (ICNP)*, 2005.

[5] S. Liu, T. Basar, and R. Srikant, "Controlling the Internet: A survey and some new results," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, 2003.

[6] S. H. Low and D. E. Lapsley, "Optimization flow control — I: basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861–874, 1999.

[7] S. H. Low, L. L. Peterson, and L. Wang, "Understanding TCP Vegas: a duality model," *J. ACM*, vol. 49, no. 2, pp. 207–235, 2002.

[8] R. Srikant, *The Mathematics of Internet Congestion Control.* Birkhauser, 2004.

[9] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.

[10] D. X. Wei, C. Jin, S. H. Low, and S. Hegde, "FAST TCP: motivation, architecture, algorithms, performance," *IEEE/ACM Transactions on Networking*, 2007.

[11] Y. Zhu and B. Li, "Overlay networks with linear capacity constraints," in *Proceedings of the 13th International Workshop on Quality of Service (IWQoS 2005)*, 2005.

[12] Y. Cui, Y. Xue, and K. Nahrstedt, "Optimal resource allocation in overlay multicast," in *Proceedings of the 11th IEEE International Conference on Network Procotols (ICNP)*, 2003.

[13] D. P. Bertsekas, *Network Optimization: continuous and discrete methods*, 1st ed. Athena Scientific, 1998.

[14] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation : numerical methods*. Prentice Hall, 1989.

[15] T. Voice, "A global stability result for primal-dual congestion control algorithms with routing," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 3, pp. 35–41, 2004.

[16] J. Wen and M. Arcak, "A unifying passivity framework for network flow control," *ECSE Dept., Rensselaer Polytechnic Institute, Troy, NY, Technical Report,Aug. 2002.*, 2002.

[17] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: Universal Topology Generation from a User's Perspective (User Manual)," Boston University, Computer Science Department, Boston, MA 02215, Tech. Rep. BU-CS-2001-003, May 2001.

[18] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proceedings of the Symposium on Operating Systems Principles*, 2001, pp. 131–145.

[19] Z. Ma, H.-R. Shao, and C. Shen, "A new multi-path selection scheme for video streaming on overlay networks," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Paris, France, June 2004.

[20] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker, "On selfish routing in internet-like environments," in *Proceedings of the ACM SIGCOMM '03*, Karlsruhe, Germany, August 2003.

[21] Y. Cui, B. Li, and K. Nahrstedt, "On achieving optimized capacity utilization in application overlay networks with multiple competing sessions," in *Proceedings of the 16th annual ACM symposium on parallelism in algorithms and architectures*. New York, NY, USA: ACM Press, 2004, pp. 160–169.

[22] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Multipath TCP: A joint congestion control and routing scheme to exploit path diversity on the internet," *IEEE/ACM Trans. on Networking*, 2007.