DISTRIBUTED JOINT OPTIMIZATION OF TRAFFIC ENGINEERING AND SERVER SELECTION

Pengye Xia, S.-H. Gary Chan

Department of CSE
The Hong Kong University of Science and Technology
{xiapengye, gchan}@cse.ust.hk

Mung Chiang

Department of EE
Princeton University
chiangm@princeton.edu

Guangyu Shi, Hongbo Zhang, Liufei Wen, Zhefeng Yan

Huawei Technologies Co., Ltd
Shenzhen, 518129, China
{shiguangyu, zhanghongbo888, wenliufei, yanzhefeng}@huawei.com

This work was supported, in part, by the General Research Fund from the Research Grant Council of the Hong Kong Special Administrative Region, China (611209), and Huawei Technologies Co., Ltd. (HUAW14-15C00609/10PN).

ABSTRACT

Internet service providers (ISP) apply traffic engineering (TE) in the underlay network to avoid congestion. On the other hand, content providers (CP) use different server selection (SS) strategies in the overlay network to reduce delay. It has been shown that a joint optimization of TE and SS is beneficial to the performance from both ISP's and CP's perspectives.

One challenging issue in such a network is to design a distributed protocol which achieves optimality while revealing as little information as possible between ISP and CP. To address this problem, we propose a distributed protocol termed PETS, in which each router of ISP makes independent traffic engineering decision and each server of CP makes independent server selection decision. We prove that PETS can achieve optimality for the joint optimization of TE and SS. We also show that PETS can significantly reduce message passing and enables ISP to hide important underlay network information (e.g., topology) from CP. Furthermore, PETS can be easily extended to handle the case of multiple CPs in the network.

Index Terms— traffic engineering, server selection, joint optimization, distributed algorithm, link-state routing, per-hop forwarding

1. INTRODUCTION

Internet service provider (ISP) owns the physical network and offers Internet connectivities to its customers. Because the network topology often contains multiple paths from a source to a destination, ISP has to decide how to split the traffic over different paths in order to achieve the best network performance (delay, load, etc.). This problem is known as the traffic engineering problem (TE).

Content provider (CP) utilizes Internet connectivities to offer content services in the network. In contrast to ISP, CP does not have the authority to change the routing in the underlying network. However, for each client request, CP can decide which server to serve the client. This problem is known as the server selection problem (SS).

Conventionally, ISP and CP solve TE and SS independently without information sharing. In this non-cooperative scenario, CP has to estimate the information of underlying network. ISP may also choose to cooperate with CP by offering CP such information. In either case, ISP and CP take turns to solve TE and SS respectively so as to optimize their own objectives. Because TE changes routing which affects SS and SS changes traffic demand which affects TE, the interaction between TE and SS can be modeled as an two-player game [2, 3]. Recent research has shown that the game converges towards an equilibrium state which is not social optimum and neither non-cooperative nor cooperative scenario results in optimal performance [1–3]. To resolve this problem, it is necessary to have a joint design of TE and SS, which differs from the cooperative scenario in that ISP and CP interact in a certain way to jointly optimize TE and SS rather than iteratively optimizing their own objectives. Such joint design is beneficial to many applications in which server-client model applies, such as video-on-demand services.

One challenging issue in the joint optimization of TE and SS[1] is to design a distributed protocol which achieves optimality while revealing as little information as possible between ISP and CP. In [1], a modularized protocol COST has been proposed. COST is guaranteed to achieve optimality. However, it has two major limitations. Firstly, COST suffers from slow converging speed and large amount of message exchange. This is because at each iteration, ISP has to solve a TE problem, CP has to solve a SS problem, and they communicate iteratively by updating the price on each link in the network. Secondly, COST requires ISP to share some key

[1] In this work, we use joint design of TE and SS and joint optimization of TE and SS interchangeably.

underlay network information (such as topology) with CP to solve the SS problem. This requirement is often not desirable for ISP. Besides, when there are multiple CPs in the network, the requirement becomes even more problematic.

In this work, we address these limitations by proposing a novel distributed protocol PETS (**PE**FT-based joint **T**raffic Engineering and **S**erver Selection) to jointly and efficiently optimize TE and SS. Our contribution is as follows. First, we present how to extend the Multi-Commodity Flow (MCF) problem formulation (whose solution is traditionally used to define the optimality of TE) to define the joint optimality of TE and SS. The extension is done by adding virtual nodes and virtual links to transform the network topology. Based on the extension, we then propose a distributed protocol PETS which utilizes link-state routing and per-hop forwarding to achieve such optimality. PETS makes use of the traffic engineering protocol PEFT [4], which originally is not intended to handle the joint optimization of TE and SS. We modify the Network Entropy Maximization (NEM) framework in PEFT and formulate a NEMR problem to solve for the optimal link weights. Using the link weights, we present how each router of ISP can make independent traffic splitting and forwarding decision to solve TE; and each server of CP can make independent demand splitting and forwarding decision to solve SS without knowing the underlay information such as topology and link weights.

We prove that PETS can achieve optimality for the joint optimization of TE and SS. PETS has the following strengths. First, PETS solves a single TE-alike problem, which is unlike the previous iterative approach and thus significantly reduces the convergence time and the number of message passing in the system. Second, ISP does not need to reveal its key underlay network information such as topology and link weights because the servers of CP can make decision distributively based on minimal information obtained from their connected routers. Moreover, PETS can be easily extended to handle the scenario where multiple CPs coexist.

The remainder of the paper is organized as follows. In Section 2, we define optimality for the joint design of TE and SS. In Section 3, we present the novel protocol PETS and prove that PETS can achieve optimality for the joint design. We present illustrative simulation results in Section 4 and conclude in Section 5.

2. DEFINING OPTIMALITY

Optimality of a TE protocol is defined through a Multi-Commodity Flow (MCF) problem. In Section 2.1, we review the Multi-Commodity Flow problem. In Section 2.2, we show that an optimal joint design of TE and SS should realize the link flows in the solution of a MCF problem with an augmented set of optimization variables. We then introduce the graph transformation technique that will be useful in the rest of the paper.

2.1. Multi-Commodity Flow Problem

We represent the network by a directed graph $G(V, E)$, where V and E are the set of nodes and links, respectively. Each node represents a router. Some nodes may be attached by a server, which will handle the requests from CP clients. For each pair of adjacent nodes u and v, let $(u, v) \in E$ be the link directed from u to v. Define the capacity of link (u, v) by $c_{u,v}$. The traffic demand between a source s and a destination t is defined as $D(s, t)$. Denote $f_{u,v}$ the aggregated traffic flow on link (u, v). We further define $f^t_{u,v}$ the (partial) flow on link (u, v) destined to t. Clearly, we have

$$f_{u,v} = \sum_{t \in V} f^t_{u,v}, \ \forall (u, v) \in E, \tag{1}$$

and

$$f_{u,v} \leq c_{u,v}. \tag{2}$$

For flow conservation, we must have

$$\sum_{(s,v)\in E} f^t_{s,v} - \sum_{(u,s)\in E} f^t_{u,s} = D(s, t), \ \forall s \in V, \ t \in V. \tag{3}$$

Let the link cost be a function of link capacity and its aggregated flow, i.e., $\Phi(f_{u,v}, c_{u,v})$. Here we consider the general case that Φ as a convex function which strictly increases with $f_{u,v}$ and satisfies

$$\Phi(f_{u,v}, c_{u,v}) \to 0, \text{ if } c_{u,v} \to \infty. \tag{4}$$

The MCF problem is to minimize

$$\sum_{(u,v)\in E} \Phi(f_{u,v}, c_{u,v}), \tag{5}$$

with variables $f_{u,v}$ and $f^t_{u,v}$, subject to Equations (1), (2) and (3). Given the traffic demand matrix D, the MCF problem is an convex optimization problem which can be solved in polynomial time. It outputs the optimal flow on each link. An optimal TE protocol should realize the flow either by link-state traffic splitting (e.g. PEFT [4]) or dividing traffic over different paths (e.g. MPLS [5]).

2.2. Optimal Link Flows for Joint Design of TE and SS

In order to find optimal link flows for TE, the demand matrix D is often assumed to be constant. However, in the context of joint design of TE and SS, this assumption is obviously invalid. This is shown as follows. Denote V_s the set of nodes attached by the servers belonging to the CP. Assume a client request is initiated at a destination t and has required a total rate of d_t. The traffic from each server node $s_i \in V_s$ to t is denoted as $D^{cp}(s_i, t)$. Each $D^{cp}(s_i, t)$ is a variable in server selection problem, and they add up to the required traffic rate, i.e.,

$$\sum_{s_i \in V_s} D^{cp}(s_i, t) = d_t, \ \forall t \in V_s. \tag{6}$$

Denote the background traffic from s_i to t by $D^{bg}(s_i,t)$. We have

$$D(s_i,t) = D^{cp}(s_i,t) + D^{bg}(s_i,t),\ \forall s_i \in V_s. \tag{7}$$

Clearly, D is no longer a constant due to CP traffic $D^{cp}(s_i,t)$.

In order to define optimality for the joint design, we propose the following approach that finds the optimal link flows by solving a MCF problem with transformation on its input topology G and demand matrix D. We add a new node N_s in graph G, and connect each server $s_i \in V_s$ to the new node by a directed virtual link with infinite capacity. We denote this new graph by G^*. All CP traffic $D^{cp}(s_i,t)$ is assigned to N_s in G^*. Therefore, we have

$$D(N_s,t) = d^t. \tag{8}$$

Figure 1 illustrates such transformation. There are 3 servers labeled as s_1, s_2 and s_3, and one destination t. A dashed arrow represents the traffic demand between two nodes, while a solid arrow represents a new virtual link added to the graph. Clearly, after the transformation, the traffic demand $D(s_i,t)$ only contains constant background traffic $D^{bg}(s_i,t)$. Besides, all CP traffic is assigned to N_s, which adds up to a constant d_t. In other words, the demand matrix in G^* becomes constant after the transformation, and the problem becomes a standard MCF problem.

The optimal solution of MCF problem on G^* contains optimal flow on physical links as well as virtual links (i.e., $\forall(N_s,s_i) \in G^*$). The flow on a virtual link (N_s,s_i) represents the aggregated CP traffic at server s_i, i.e.,

$$f_{N_s,s_i} = \sum_{t\in V} D^{cp}(s_i,t). \tag{9}$$

Theorem 1: An optimal joint design of TE and SS should realize the link flows in the optimal solution of the Multi-Commodity Flow problem on G^*.

Proof: See Appendix A.1.

3. AN OPTIMAL PROTOCOL: PETS

In this section, we propose a novel protocol PETS (**PE**FT-based joint **T**raffic Engineering and **S**erver Selection) for joint design of TE and SS which achieves the MCF optimality. In Section 3.1, we first review PEFT [4]. In Section 3.2, we present the details of PETS. We further show that PETS achieves optimality for the joint design. In Section 3.3, we discuss other properties of PETS.

3.1. PEFT Review

PEFT is an optimal TE protocol that uses link-state routing and per-hop forwarding [4]. There are 3 major steps in PEFT. It first solves a MCF problem to get the optimal link flows. Then, based on the link flows, it obtains a set of link weights.

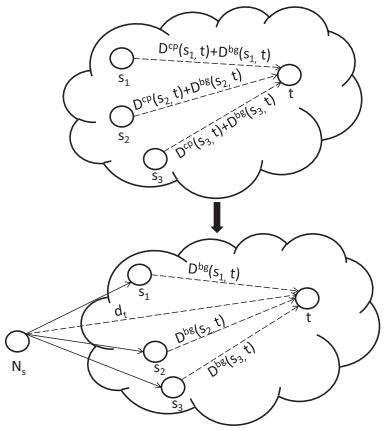


Fig. 1. Illustration of the transformation.

In the final step, each router splits the traffic based on link weights to realize the optimal flow. The first step has been shown in Section 2.1; we present the details on the remaining steps here.

Given optimal link flows, PEFT solves the following Network Entropy Maximization problem (NEM) to find out a set of optimal link weights:

NEM

$$\begin{aligned} \text{Maximize:}\quad & \sum_{s,t\in V} D(s,t)\left(\sum_i -x^i_{s,t}\log x^i_{s,t}\right), \\ \text{subject to}\quad & \sum_{s,t,i:(u,v)\in P^i_{s,t}} D(s,t)x^i_{s,t} \le f_{u,v},\ \forall(u,v), \\ & \sum_i x^i_{s,t} = 1,\ \forall s,\ t\in V, \end{aligned} \tag{10}$$

where $P^i_{s,t}$ is the i^{th} path from s to t, while $x^i_{s,t}$ is the fraction of the traffic $D(s,t)$ that is routed along $P^i_{s,t}$.

The partial Lagrangian of NEM by relaxing the inequality constraint can be written as

$$\begin{aligned} &\sum_{(u,v)\in E}\lambda_{u,v}f^t_{u,v} + \sum_{s,t\in V} D(s,t)\left(\sum_i -x^i_{s,t}\log x^i_{s,t}\right) \\ &-\sum_{(u,v)\in E}\lambda_{u,v}\left(\sum_{(u,v)\in P^i_{s,t}} D(s,t)x^i_{s,t}\right), \end{aligned} \tag{11}$$

where $\lambda_{u,v}$ are the Lagrangian multipliers and also the variables in Lagrange dual problem. Optimal $\lambda_{u,v}$ and $x^i_{s,t}$ maximize the Lagrangian subject to the equality constraint. Note that the optimal $\lambda_{u,v}$ can be used as link weights to realize the optimal flow, i.e.,

$$w(u,v) = \lambda^*_{u,v}. \tag{12}$$

Given link weights, a router at u splits the incoming traffic using the following equation:

$$f_{u,v}^t = f_u^t \frac{\Gamma\left(h_{u,v}^t\right)}{\sum_{(u,j)\in E}\Gamma\left(h_{u,j}^t\right)}, \tag{13}$$

where f_u^t is the aggregation of all incoming traffic at u which is destined to t and $f_{u,v}^t$ is the amount of the traffic split to an outgoing link (u,v). Let d_u^t be the shortest path length from u to t. Obviously, $d_v^t + w(u,v)$ is the length of shortest path from u to t passing v. Denote $h_{u,v}^t$ the difference between the length of the two paths, which is obviously given by

$$h_{u,v}^t = d_v^t + w(u,v) - d_u^t. \tag{14}$$

$\Gamma\left(h_{u,v}^t\right)$ is the traffic splitting function for link (u,v), which can be calculated using the equivalent number γ_v^t, i.e.,

$$\Gamma\left(h_{u,v}^t\right) = e^{-h_{u,v}^t}\gamma_v^t. \tag{15}$$

We call $\Gamma\left(h_{u,v}^t\right)$ in the above equation as exact traffic splitting function. To prevent loops in routing, we can use downward packet forwarding scheme where at each node, the traffic is split to those nodes with shorter distance to the destination. The resulting traffic splitting function is then

$$\Gamma\left(h_{u,v}^t\right) = \begin{cases} e^{-h_{u,v}^t}\gamma_v^t, \text{ if } d_u^t > d_v^t, \\ 0, \text{ otherwise.} \end{cases} \tag{16}$$

Similar to OSPF [6, 7], each router should have complete information on the network topology and the link weights in order to compute the shortest path and traffic splitting function. Although PEFT achieves optimal TE, it cannot be used to jointly optimize traffic engineering and server selection since traffic matrix is assumed to be fixed and given. In the next section, we show how PETS makes use of PEFT to achieve the joint optimization.

3.2. PETS

We propose PETS to jointly optimize TE and SS by modifying PEFT in each of its three steps to incorporate new variables and constraints posed by the joint design. Firstly, ISP solves a standard MCF problem based on transformed topology G^* (See Section 2.2). The solution defines the optimal link flows which can be realized by the next steps.

In this step, we find link weights by solving a "relaxed" form of NEM (Equation (10)), denoted by **NEMR**. The new problem *relaxes* the inequality in NEM for each virtual link (N_s, s_i). Using to Equation (8), the original inequalities in Equation (10) for virtual links can be rewritten as

$$\sum_{s,t,i:(N_s,s_j)\in P_{N_s,t}^i} d^t \cdot x_{s,t}^i \le f_{N_s,s_i}, \forall\, s_j \in V_s. \tag{17}$$

Because the aggregated flow on a virtual link is no larger than the total CP traffic, we can relax the inequalities as

$$\sum_{s,t,i:(N_s,s_j)\in P_{N_s,t}^i} d^t \cdot x_{s,t}^i < \sum_t d^t + \delta, \forall\, s_j \in V_s, \tag{18}$$

where δ is some positive constant.

Because NEM is strictly feasible, NEMR must be strictly feasible as well. As a result, strong duality holds for NEMR. We can solve NEMR by dual decomposition. Denote the solution to the dual problem by $\left\{\lambda_{u,v}^*\right\}$. We claim the following two theorems.

Theorem 2: The optimal dual variables associated with each virtual link λ_{N_s,s_i}^* equal 0.

Proof: See Appendix A.2.

Theorem 3: The optimal link flows can be realized by link-state routing and per-hop forwarding, if 1) we use the optimal dual variables given by NEMR as link weights; and 2) each node (including virtual node) splits traffic using traffic splitting functions as specified in PEFT.

Proof: See Appendix A.3.

By Theorem 3, we use the optimal dual variables given by NEMR as link weights, i.e.,

$$w(u,v) = \lambda_{u,v}^*. \tag{19}$$

In the final step, we use the above set of link weights to realize the optimal link flows and solve the joint optimization of TE and SS in a distributed manner. Given link weights, each node computes the traffic splitting function to realize the optimal link flows as specified in Section 3.1. *Regarding TE*, each router independently computes the traffic splitting function at its associated physical node. It then splits and forwards traffic according to the function to solve TE problem as specified in PEFT.

On the other hand, *regarding SS*, each server of CP independently computes the traffic splitting function at the virtual node N_s. Each server then uses the function to split the incoming CP traffic and forwards certain portion of traffic to other servers to solve SS. The detail is as follows. We first show how to compute traffic splitting function at the virtual node. Referring to Equation (14) and Theorem 2, we have

$$h_{N_s,s_i}^t = d_{s_i}^t - d_{N_s}^t. \tag{20}$$

The shortest path from N_s to t passes through one of the virtual links. Therefore,

$$d_{N_s}^t = \min_{s_i}\left\{d_{s_i}^t\right\}. \tag{21}$$

Given $d_{s_i}^t$ at each s_i, according to Equations (13) and (15), the traffic split to server s_i is

$$f_{N_s,s_i}^t = f_{N_s}^t \frac{e^{-h_{N_s,s_i}^t}\gamma_{s_i}^t}{\sum_{s_i} e^{-h_{N_s,s_i}^t}\gamma_{s_i}^t}. \tag{22}$$

Using Equation (22), each server independently splits the CP traffic demand which it receives from the clients. It forwards the portion of demand $f^t_{N_s,s_i}$ to s_i (if the server itself is not s_i). Note that the forwarded demand will not be split again. The CP traffic demand is then satisfied independently at the each server. Because all servers use the same traffic splitting function, it is clear that the total CP traffic demand is split correctly according to the splitting function at the virtual node. Clearly, the computation of Equation (22) only requires the shortest path lengths $d^t_{s_i}$ and equivalent number $\gamma^t_{s_i}$ for each s_i. Because each router has a complete knowledge of all $d^t_{s_i}$ and $\gamma^t_{s_i}$, each server of CP simply polls the information from its connected router. In this way, important information of ISP such as topology and link weights will not be revealed to CP.

Since traffic splitting at each physical node (which is done by a router) or at the virtual node (which is due to the joint efforts of the CP servers) is according to the splitting function definition in PEFT, we confirm that PETS is able to realize the optimal link flows given by MCF by Theorem 3. Furthermore, by Theorem 1, we show that PETS achieves the joint optimality of TE and SS.

PETS achieves MCF optimality at the cost of some routing loops (because of the use of PEFT). To prevent looping, we propose PETS-D which stands for PETS using downward packet forwarding. Each router calculates the traffic splitting function according to Equation (16). Because the virtual node N_s only has outgoing links, there is no routing loops passing N_s. Thus we do not have to apply downward forwarding at N_s. Hence, traffic splitting at each CP server remains unchanged. Clearly, PETS-D prevents routing loops at some sacrifice of optimality. We show in section 4 that the cost is negligible.

3.3. PETS Properties

PETS is a fully distributed protocol for joint optimization of TE and SS. To solve TE, Each router independently splits the incoming traffic over outgoing links. To solve SS, Each server of CP independently splits its CP requests and forwards a certain portion of the requests to the other servers. It has been proved that, in this distributed manner, the joint optimality of TE and SS can be achieved.

PETS is a fast converging protocol which does not require large amount of message exchanging in the network. As PETS modifies the steps in PEFT, the total processing time and message exchanging is comparable to solving a single TE problem with PEFT. Recall that previous approach (i.e., COST [1]) requires an iterative approach which ISP solves a TE problem and CP solves a SS problem, and ISP and CP interact through link prices updated in each iteration. Since PETS does not take this iterative approach, it has faster convergence speed and less messages to be passed in the system.

PETS lets ISP hide its key information from CP, which is important because ISP and CP may come from different entities. In PETS, each server of CP computes its traffic splitting function by polling shortest path lengths and equivalent numbers from its connected router. As a result, ISP needs not to provide CP its underlying network information such as topology and link weights.

PETS can be easily extended to handle several CPs. For each CP, we can add a virtual node and a set of virtual links to the graph. ISP uses the new graph to compute the link weights and configure traffic splitting at routers. Servers of each CP polls adjacent routers for necessary information to split the CP traffic. Clearly, each CP is independent of each other and there is no need for centralized management.

4. SIMULATION RESULTS

We have simulated PETS and PETS-D (PETS with downward packet forwarding, Section 3.2) on the network topology illustrated in Figure 2. Each physical link has the same link capacity. Two servers are attached to Node 1 and Node 4, respectively. The CP traffic demand is initiated at Node 3. There is background traffic from Node 2 to Node 3. As described before, we add a virtual node 6 to the graph and connect it to Nodes 1 and 4 by two virtual links l_{17} and l_{18}. As a simple illustration, we use two types of link cost function in the simulation: $\Phi_1(f_{u,v}, c_{u,v}) = \frac{f_{u,v}}{c_{u,v}-f_{u,v}}$, which is the M/M/1 link queuing delay, and Φ_2 is a piecewise linear function commonly used as ISP link cost [6]. Figure 3 illustrates the two link cost functions.

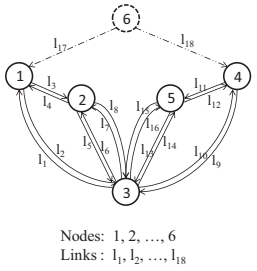
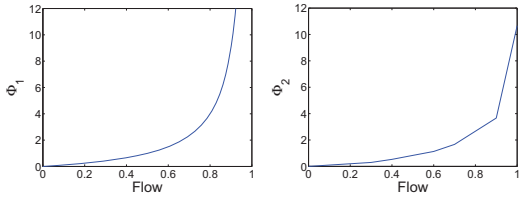


Fig. 2. Illustration of network topology. Servers are attached to Node 1 and Node 4. Demand is initiated to node 3. Node 6 is the virtual node. All links except l_{17} and l_{18} are physical links.

Two parameters are varied in the simulation. One is the link capacity. When traffic is fixed, increasing the link capacity will lead to smaller link utilization. The other is the percentage of CP traffic in the total traffic. The performance



(a) Φ_1 (b) Φ_2

Fig. 3. Comparison of link cost functions Φ_1 and Φ_2.

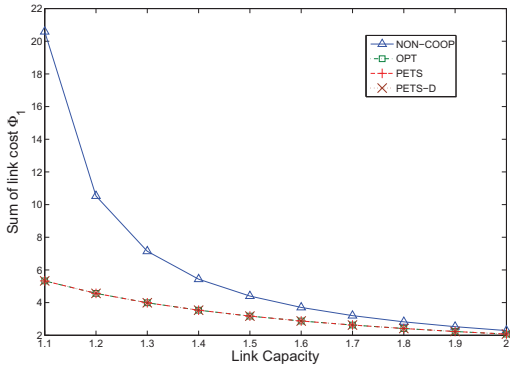
metric is the sum of link cost of each link, using either Φ_1 or Φ_2.

We have compared PETS and PETS-D with two other schemes. One is the optimal joint design of TE and SS, denoted as OPT. The other is the non-cooperative scheme in which OSPF is used for TE and a heuristic based on delay measure is used for SS, denoted as NON-COOP. The NON-COOP scheme is similar to the common practice and is equivalent to Model 1 in [1], in which ISP and CP solves TE and SS iteratively and ISP does not reveal underlay information to CP.

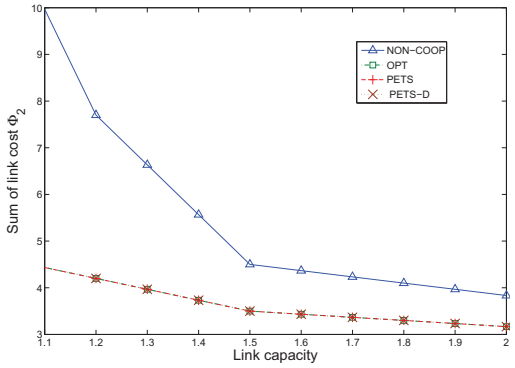
Figure 4 shows the sum of link cost versus link capacity given different schemes. Link cost Φ_1 is used in Figure 4(a) while Φ_2 is used in Figure 4(b). The sum of link cost decreases with link capacity because the link utilization decreases. PETS and PETS-D achieve the same optimum given by OPT. Their performance are much better than the NON-COOP scheme. The performance gap is more significant when the link capacity is small. In other word, PETS and PETS-D have more advantage when network becomes congested. In order to see how the link utilization decreases with link capacity, we plot the corresponding maximum link utilization versus link capacity in Figure 5. As the simulation results on Φ_1 and Φ_2 show the similar trend, we only show result of Φ_1 in the remaining part.

We show in Figure 6 the sum of link cost Φ_1 versus CP traffic percentage given different schemes. The sum of link cost first decreases and then increases with CP traffic percentage. The cost is large when CP traffic percentage is small because the paths for background traffic becomes congested. Similarly, the cost is large when CP traffic percentage is large because the paths for CP traffic becomes congested. PETS and PETS-D achieve the same optimum given by OPT, and they outperform NON-COOP scheme significantly, especially when the network is congested.

We show in Figure 7 the traffic splitting of PETS given different CP traffic percentage. For each node, the percentage on each outgoing link represents the ratio of incoming traffic that is split to that link. Note that the percentage on virtual links shows how the CP client requests is split to different servers. The left plot illustrate the scenario when CP traffic percentage is high. The servers at node 1 and node 4 have similar load because background traffic (which is from node 2



(a) with link cost Φ_1



(b) with link cost Φ_2

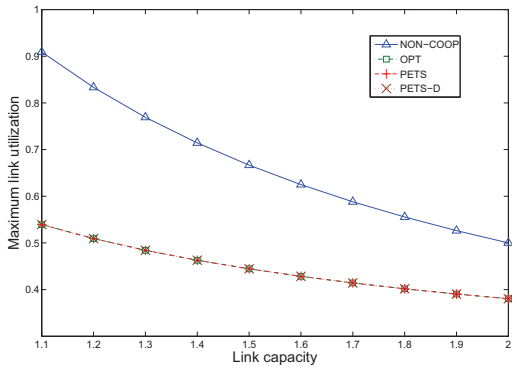
Fig. 4. Sum of link cost versus link capacity given different schemes.

to node 3) is low. The right plot illustrates the scenario when CP traffic percentage is low. Majority of CP traffic is sent from server at node 4. This is because the high background traffic makes the paths from node 1 to node 3 become more congested.

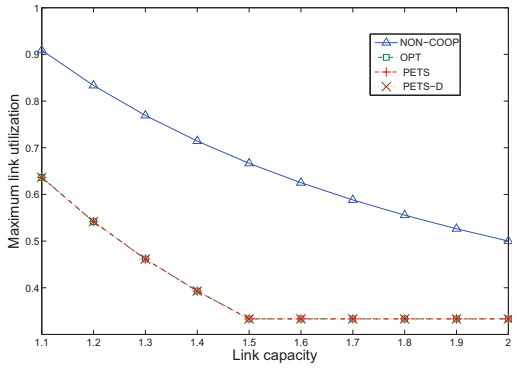
The link weights on the virtual links always have zero value, which justifies Theorem 2. Table 1 shows the set of optimal link weights of PETS. The set corresponds to the point in Figure 4(a) with link capacity of 1.5. Note that the virtual links (index 17 and 18) have zero link weights.

Table 1. Link weights of PETS.

Index	1	2	3	4	5	6
Weight	139.4	100.0	60.7	100.0	80.4	100.0
Index	7	8	9	10	11	12
Weight	80.4	100.0	139.4	100.0	60.4	100.0
Index	13	14	15	16	17	18
Weight	80.2	100.0	80.2	100.0	0	0



(a) with link cost Φ_1



(b) with link cost Φ_2

Fig. 5. Maximum link utilization versus link capacity given different schemes.

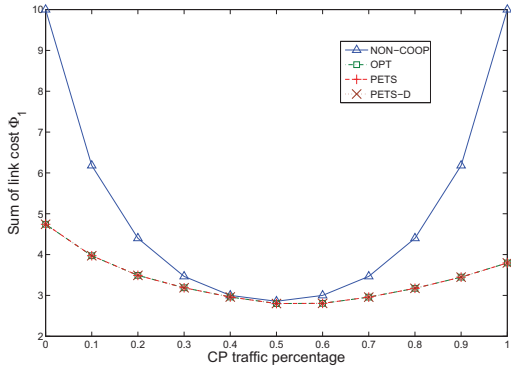


Fig. 6. Sum of link cost Φ_1 versus CP traffic percentage given different schemes.

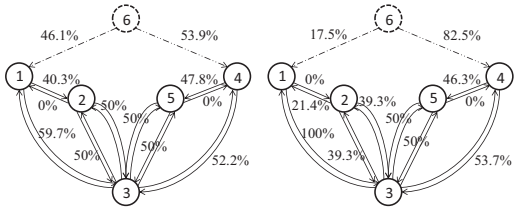


Fig. 7. Illustration of traffic splitting of PETS given different CP traffic percentage. The left figure illustrates the scenario when CP traffic percentage is high. The right figure illustrates the scenario when CP traffic percentage is low.

5. CONCLUSION

We propose a distributed protocol PETS to jointly optimize traffic engineering and server selection. In PETS, we first use a novel approach to define optimality for the joint optimization by solving a Multi-Commodity flow problem with transformed topology. We further propose a NEMR problem which can be solved for optimal link weights. We then design how a router of ISP makes distributed traffic splitting decision and how a server of CP makes distributed demand splitting decisions to jointly optimize TE and SS.

PETS advances the joint design of traffic engineering and server selection in several ways. First, PETS is a fully distributive protocol which does not need centralized management system. Second, it has faster convergence speed and less message exchange. Moreover, PETS does not need ISP to reveal its key underlying network information. Finally, PETS can be easily extended to handle a network with multiple CPs.

We prove that PETS can achieve optimality for the joint design and illustrate numerically by simulation. In particular, the simulation results shows that PETS with downward forwarding can prevent routing loops with negligible performance cost. The advantage of PETS and PETS-D over current practice can be significant especially when network is congested.

6. REFERENCES

[1] Jiang, W., Zhang-Shen, R., Rexford, J., and Chiang, M. "Cooperative content distribution and traffic engineering in an ISP network." Proc. SIGMETRICS '09. ACM, New York, NY, 239-250. 2009

[2] D. DiPalantino and R. Johari, "Traffic Engineering V.S. Content Distribution: A Game Theoretic Perspective," in Proc. IEEE INFOCOM, 2009.

[3] Liu, Y., Zhang, H., Gong, W. and Towsley, D., "On the interaction between overlay routing and underlay routing," in Proc. of IEEE INFOCOM'05.

[4] D. Xu, M. Chiang, and J. Rexford, "Link-state routing with hop-by-hop forwarding can achieve optimal traffic engineering," Proc. IEEE INFOCOM, Phoenix, AZ, April 2008.

[5] D. O. Awduche,"MPLS and traffic engineering in IP networks," *IEEE Communication Magazine*, pp. 44-47, Dec. 1999.

[6] B. Fortz, M. Thorup, "Internet Traffic Engineering by Optimizing OSPF weights," in: Proc. 19th IEEE Conf. Comp. Comm., vol. 2, 2000.

[7] A. Sridharan, R. Guerin, and C. Diot, "Achieving near-optimal traffic engineering solutions for current OSPF/IS-IS networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 2, pp. 234-247, 2005.

[8] Stephen Boyd, Lieven Vandenberghe, "Convex Optimization", Cambridge University Press, 2004.

A. APPENDIX

A.1. Proof of Theorem 1

Let MCF-G denote the Multi-Commodity flow problem on G with additional equality constraints (6) and (7). And let MCF-G* denote the Multi-Commodity flow problem on G^*. We aim to prove that the solution of MCF-G is equivalent to that of MCF-G* with one to one correspondence on their decision variables.

The objective of MCF-G* can be written as,

$$\sum_{(u,v)\in G} \Phi(f_{u,v}, c_{u,v}) + \sum_{s_i} \Phi(f_{N_s,s_i}, c_{N_s,s_i}). \quad (23)$$

Because all the virtual links have infinite capacity, combining Equation (4), we have,

$$\sum_{(u,v)\in G^*} \Phi(f_{u,v}, c_{u,v}) = \sum_{(u,v)\in G} \Phi(f_{u,v}, c_{u,v}). \quad (24)$$

Note that the right-hand side is the objective for MCF-G. Therefore, the two problem has the same objective.

Moreover, we can make one-to-one correspondence between the variables in the two problems. Each flow on physical link in MCF-G corresponds to the flow on the same physical link in MCF-G*. Each variable $D^{cp}(s_i, t)$ in MCF-G corresponds to f_{N_s,s_i} in MCF-G*. With this correspondence, it is easy to see that the constraints in two problems are equivalent.

Therefore, the two problems must have equivalent solutions. For each physical link, its flow in MCF-G is equal to that in MCF-G*. Because a joint design of TE and SS should realize link flows in MCF-G, it must also realize the link flows in MCF-G*.

A.2. Proof of Theorem 2

The optimal primal variables $x^{i*}_{s,t}$ and dual variables $\lambda^*_{u,v}$ maximize the Lagrangian of NEMR and satisfy Karush-Kuhn-Tucker (KKT) condition [8]. As required by complementary slackness in KKT condition, we have,

$$\lambda^*_{N_s,s_i} \cdot \left(\sum_{s,t,i:(N_s,s_j)\in P^i_{N_s,t}} d^t \cdot x^i_{s,t} - \sum_t d^t - \delta \right) = 0. \quad (25)$$

Since inequalities in (18) always hold strictly, we must have,

$$\lambda^*_{N_s,s_i} = 0, \forall s_i \in V_s. \quad (26)$$

Proof ends.

A.3. Proof of Theorem 3

Following the proof in [4], the two conditions ensures that the link flows from the solution of NEMR can be realized by link-state routing and per-hop forwarding. However, we have to prove that aggregated flows in NEMR are equal to that given by the Multi-Commodity flow problem on G^*. Denote the aggregated flow on link (u, v) by $\tilde{f}_{u,v}$. By definition,

$$\tilde{f}_{u,v} = \sum_{s,t,i:(u,v)\in P^i_{s,t}} D(s,t) x^i_{s,t}. \quad (27)$$

Therefore, as required by inequalities on physical links, we have,

$$\tilde{f}_{u,v} \le f_{u,v}, \forall u, v \ne N_s. \quad (28)$$

Because the link cost function $\Phi(f_{u,v}, c_{u,v})$ is a strictly increasing with $f_{u,v}$, we have,

$$\sum_{(u,v)\in G} \Phi(\tilde{f}_{u,v}, c_{u,v}) \le \sum_{(u,v)\in G} \Phi(f_{u,v}, c_{u,v}) \quad (29)$$

Therefore, if there exists a $\tilde{f}_{u,v}$ less than $f_{u,v}$, we have a strict inequality, i.e.,

$$\sum_{(u,v)\in G} \Phi(\tilde{f}_{u,v}, c_{u,v}) < \sum_{(u,v)\in G} \Phi(f_{u,v}, c_{u,v}). \quad (30)$$

Combining (24) and (30), we have,

$$\sum_{(u,v)\in G^*} \Phi(\tilde{f}_{u,v}, c_{u,v}) < \sum_{(u,v)\in G^*} \Phi(f_{u,v}, c_{u,v}), \quad (31)$$

which is contradicting to the fact that $\{f_{u,v}\}$ are the optimal solution for Multi-Commodity flow problem. Therefore, $\tilde{f}_{u,v}$ must be equal to $f_{u,v}$. The proof is complete.