Improving the Interaction between Overlay Routing and Traffic Engineering

Gene Moo Lee[1] and Taehwan Choi[2]

[1] Samsung Advanced Institute of Technology
Yongin-si, Gyeonggi-do, 446-712, Korea
gene.lee@samsung.com
[2] Department of Computer Sciences
University of Texas at Austin, TX 78712, U.S.A.
ctlight@cs.utexas.edu

Abstract. Overlay routing has been successful as an incremental method to improve Internet routing by allowing its own users to select their logical routing. In the meantime, traffic engineering (TE) are being used to reduce the whole network cost by adapting physical routing in response to varying traffic patterns. Previous studies [1, 2] have shown that the interaction of the two network components can cause huge network cost increases and oscillations. In this paper, we improve the interaction between overlay routing and TE by modifying the objectives of both parties. For the overlay part, we propose TE-awareness which limits the selfishness by some bounds so that the action of overlay does not offensively affect TE's optimization process. Then, we suggest COPE [3] as a strong candidate that achieves close-to-optimal performance for predicted traffic matrices and that handles unpredictable overlay traffic efficiently. With extensive simulation results, we show the proposed methods can significantly improve the interaction with lower network cost and smaller oscillation problems.

1 Introduction

Overlay routing has been proposed as an incremental method to enhance the current Internet routing without requiring additional functionality from the IP routers. Overlay techniques have been successful for many applications, including application-layer multicast [4, 5, 6], web content distribution [7], and overlay routing [8, 9].

In an overlay network, overlay nodes form an application-layer logical network on top of an IP layer network. Overlay networks enable users to make routing decisions at the application layer by relaying traffic among overlay nodes. We can achieve better route than default IP routing because some problematic and slow links can be bypassed. In addition, overlay routing can take advantage of some fast and reliable paths, which could not be used in the default IP routing due to business relationship.

By its nature, overlay routing has selfish behavior [1,10]. In other words, overlay acts strategically to optimize its performance. This nature of overlay makes impact on the related components of the network. We term that overlay routing has *vertical interaction* with IP layer's traffic engineering (TE). Whenever an overlay network changes its logical routing, the underlay routing observes physical traffic pattern changes. Then network operators use TE techniques [11,12,13] to adapt the routing to cope with the new traffic demands. This new routing, in turn, changes link latency observed by the overlay network, and then overlay makes another decision to change its routing. TE cares about the network as a whole, in order to provide better service to all the users. However, the main objective of overlay routing is to minimize its own traffic latency. Then an interesting issue is to understand the interaction between overlay routing and IP routing.

The interaction between overlay routing and TE was first addressed by Qiu et al. [1], where the authors investigate the interaction of overlay routing with OSPF and MPLS TE. Keralapura et al. [14] examine the interaction dynamics between the two layers of control from an ISP's view. Liu et al. [2] formulate the interaction as a two-player game, where overlay attempts to minimize its delay and TE tries to minimize the network cost. The paper shows that the interaction causes a severe oscillation problem to each player and that both players lose as the interaction proceeds.

In this paper, we propose *TE-aware overlay routing*, which takes the objective of underlay routing into account, instead of blindly optimizing its performance. Moreover, we argue that it is better off for both players if the underlay routing is oblivious to the traffic demands. We suggest COPE [3] as a strong candidate for this purpose.

The paper is organized as follows. First, we formally describe the mathematical model in Section 2. Section 3 formulates the interaction of overlay routing and TE as a non-cooperative two-player game. Then various underlay routing schemes are described in Section 4, and TE-aware overlay routing is introduced in Section 5. Section 6 evaluates the proposed methods with extensive simulation results. Finally, Section 7 concludes the paper and gives future direction.

2 Model

In this section, we describe the mathematical model, which will be used throughout the paper. Basically, TE and overlay have different viewpoints of the network. Network operators know all the underlying structure of the physical network, whereas overlay has a logical view of the network.

Table 1 summarizes the notations for vertical interaction. First, we use a graph $G = (V, E)$ to denote an underlay network, where V is the set of physical nodes and E is the set of edges between nodes. We use l or (i, j) to denote a link and $cap(l)$ to refer the capacity of link l. For the overlay network, we use a subgraph $G' = (V', E')$ of the underlay graph G. In G', we use i' to represent the overlay node built upon physical node i in underlay graph G. Overlay node i' is connected to j' by a *logical link* (i', j'), which corresponds to a physical *path* from i to j in G.

Table 1. Notations for vertical interaction

(i,j), l	physical link
(i',j')	logical link
$cap(l)$	capacity of a physical link l
$v_{st}(l)$	flow of d_{st} on link l
$f_{st}(l)$	fraction of d_{st} on link l
$t(l)$	traffic rate at link l
d_{st}	total TE demand on physical node pair (s,t)
$d_{s't'}$	overlay demand on pair (s',t')
d_{st}^{under}	TE demand due to underlay traffic
$d_{st}^{overlay}$	TE demand due to overlay flow
$P^{(s't')}$	set of logical paths from s' to t'
$\delta_p^{s't'}$	path mapping coefficient
$h_p^{(s't')}$	overlay flow on logical path p

Now, we need to have different notations for overlay and underlay traffic demands: d_{st} is used to indicate the total traffic demand from node s to t, including overlay and non-overlay traffics, and d_{st} is a sum of d_{st}^{under} and $d_{st}^{overlay}$. d_{st}^{under} refers to the background traffic by non-overlay demands. Next, it is important to differentiate $d_{st}^{overlay}$ from $d_{s't'}$: $d_{s't'}$ indicates the logical traffic demand from overlay node s' to t', whereas $d_{st}^{overlay}$ is the physical traffic demand on physical node pair (s,t), generated by overlay network. In other words, $d_{st}^{overlay}$ is computed by the overlay routing based on the current logical demand $\{d_{s't'} | \forall s', t' \in E'\}$.

The third group of notations is for the overlay routing. $P^{(s't')}$ is the set of logical paths from s' to t'. $\delta_p^{s't'}$ is the path mapping coefficient, where the value is 1, if logical link (s',t') is on logical path p, and 0, otherwise. $h_p^{(s't')}$ is the amount of overlay demand $d_{s't'}$ flowing on logical path p.

3 Vertical Interaction Game

Based on the formulations in the previous section, TE and overlay routing are coupled through the mapping from the logical level path to physical level links. We can formulate the interaction as a non-cooperative two-player game as described in Fig. 1. The first player is the ISP's TE and the second player is the overlay routing for the user's side. The interaction consists of *sequential* moves of the two players. Each player takes turn and makes action to optimize its performance. Based on the overlay demand and flow conditions on the physical links, overlay calculates the optimal flows on the logical routing. These logical flows and the underlay background traffic are coupled to form the total traffic matrix, which is the input for TE. Then TE optimizes its performance by adapting the flows on the physical links, which in turn affects the delays experienced by the

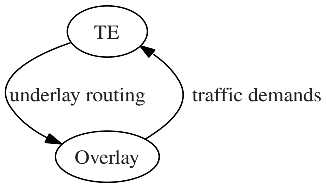

Fig. 1. vertical interaction game: TE determines the physical routing, which decides link latency experienced by overlay. Given the observed latency, overlay optimizes its logical routing and changes the physical traffic demands, which, in turn, affects the underlay routing.

overlay. This interaction continues until the two players come up with the Nash equilibrium point [15, 16], if there exists.

In game theory, Nash equilibrium is a kind of optimal collective strategy in a game involving two or more players, where no player has anything to gain by changing only his or her own strategy. If each player has chosen a strategy and no player can benefit by changing his or her strategy while the other players keep theirs unchanged, then the current set of strategy choices and the corresponding payoffs constitute a Nash equilibrium.

Liu et al. [2] prove the existence of Nash equilibrium in a simple interaction game, where the topology consists of three nodes and there is a single demand between two nodes. Even though we can prove the existence of a convergent point, the interaction process does not guarantee that two players' behaviors converge to the Nash equilibrium. Moreover, if the game gets complicated, it is even harder to anticipate the interaction process. The authors show that in a realistic scenario, both TE and overlay routing experience substantial performance loss due to the oscillation.

The main direction of our work is to improve the vertical interaction between overlay routing and traffic engineering. First, we want the interaction game to converge faster because the oscillation in this game degrades the performance of both players. Next, we try to reduce the performance variation in the transient oscillation process.

4 Traffic Engineering

In the vertical interaction game, we first discuss about three TE schemes as the first player: Multi-Protocol Label Switching (MPLS) [12], oblivious routing [17], and Common-case Optimization with Penalty Envelope (COPE) [3].

The output of TE is IP-layer routing, which specifies how traffic of each Origin-Destination (OD) pair is routed across the network. Typically, there is path diversity: there are multiple paths for each OD pair, and each path routes a fraction of the traffic.

First, Multi-Protocol Label Switching (MPLS) [12] provides an efficient support of explicit routing, which is the basic mechanism for TE. Explicit routing allows a particular packet stream to follow a predetermined path. The combination of MPLS technology and its TE capabilities enable the network operator to load-balance the traffic demands adaptively to optimize the network performance. There are two possible ways to describe the network performance: maximum link utilization (MLU) and total link latency.

Network operators worry about over-loaded links, because these links can be a bottleneck for the whole network performance. A slight change of the traffic pattern may overload the over-utilized links. Therefore, we want to minimize the MLU. We use this metric to measure the TE performance in this paper. Given the traffic demand matrix, the goal of MPLS TE is to choose a physical link flow allocation that minimizes MLU.

MPLS TE optimizes the paths based on the currently observed traffic matrix. Unfortunately, measuring and predicting traffic demands are really difficult problems. Flow measurements are rarely available on all links and ingress/egress points. Moreover, demands change over time on special events such as DoS attack, flash crowds, and internal/external network failures.

Oblivious routing [17] is proposed to resolve this issue. It calculates an optimal routing which performs reasonably well *independent* of traffic demands. In other words, this "demand oblivious" routing is designed with little knowledge of traffic demands, taking only the topology along with link capacities into account.

MPLS TE can be regarded as an extreme case of online adaptation. An advantage of this scheme is that it achieves the best performance for the current traffic demand. However, if there are significantly fast traffic changes, such method can suffer a large transient penalty. Oblivious routing is a way to handle unpredicted traffic spikes. However, a potential drawback of completely oblivious routing is its sub-optimal performance for the normal traffic demand.

Common-case Optimization with Penalty Envelope (COPE) [3] is proposed as a hybrid combination of predication-based optimal routing and oblivious routing. COPE handles both dynamic traffic and dynamic inter-domain routes and, at the same time, achieves close-to-optimal performance for normal, predicted traffic matrices.

5 Overlay Routing

Now, we formulate the strategy functions for overlay routing in the vertical interaction game. We start with the default overlay routing, which we term *selfish overlay routing*. Given the current underlay routing and experienced link latency, selfish overlay tries to minimize its total latency by changing the loads for each logical path in the overlay network.

Then, we propose a variation of overlay routing. We introduce a new optimization strategy, in which overlay takes the presence of traffic engineering into account. We term this as *TE-aware overlay routing*.

5.1 Selfish Overlay Routing

Overlay routing algorithm determines a logical path flow allocation $\{h_p^{(s't')} | \forall s', t' \in V', \forall p \in P^{(s't')}\}$ that minimizes the delay experienced by the overlay traffic. By $h_p^{(s't')}$, we denote the logical overlay demand from s' to t' allocated to path p.

Individual overlay users may choose their routes independently by probing the underlay network. However, we assume that a centralized entity calculates routes for all overlay users. Given the physical network topology, underlay routing, and experienced latency for each link, optimal overlay routing can be obtained by solving the following non-linear optimization problem:

$$\begin{aligned}
\min \quad & \sum_{l} \frac{t(l)^{overlay}}{cap(l) - t(l)} \\
\text{subject to} \quad & h_p^{(s't')} \text{ is a logical routing} \\
& \forall \text{ link } l : t(l) = \sum_{s,t} f_{st}(l)(d_{st}^{under} + d_{st}^{overlay}) \\
& \forall \text{ link } l : t(l)^{overlay} = \sum_{s,t} f_{st}(l) d_{st}^{overlay} \\
& \forall s, t \in V : d_{st}^{overlay} = \sum_{s',t',p} \delta_p^{s't'} h_p^{(s't')}
\end{aligned}$$

The first constraint ensures that the logical routing satisfies the logical flow conservation constraints. This can be expressed as follows:

$$\begin{aligned}
& \forall s', t' \in V' : \sum_{p \in P^{(s't')}} h_p^{(s't')} = d_{s't'} \\
& \forall s', t' \in V' : h_p^{(s't')} \geq 0.
\end{aligned}$$

Note that the main objective of problem is non-linear. But we can linearize the non-linear part of the program [18].

5.2 TE-Aware Overlay Routing

Based on the selfish overlay routing, we can change the optimization strategy to ensure the overlay is *TE-aware*. By TE-awareness, we mean the selfishness of the overlay is limited by some bound so that the action of overlay does not offensively affect the TE's optimization process.

The basic idea is this: (1) when the current latency is below the average latency, the overlay tries to minimize its own traffic amount, given that the current latency is preserved (load-balancer). (2) If the latency is above the average, then overlay changes the logical routing to improve the latency, but, at the same time, it avoids a specific link to be overloaded (limited-optimizer).

The first part, load-balancer, can be formalized as follows:

$$\begin{aligned}
\min\ & \sum_{s,t} d_{s,t}^{overlay} \\
\text{subject to}\ & h_p^{(s't')} \text{ is a logical routing} \\
& \forall \text{ link } l : t(l) = \sum_{s,t} f_{st}(l)(d_{st}^{under} + d_{st}^{overlay}) \\
& \forall \text{ link } l : t(l)^{overlay} = \sum_{s,t} f_{st}(l) d_{st}^{overlay} \\
& \forall s,t \in V : d_{s,t}^{overlay} = \sum_{s',t',p} \delta_p^{s't'} h_p^{(s't')} \\
& \sum_l \frac{t(l)^{overlay}}{cap(l) - t(l)} \leq \Theta
\end{aligned}$$

Here, the main objective is to minimize the total overlay traffic amount. The last constraint guarantees that the current latency is preserved. (Θ in the last constraint indicates the current latency.)

Secondly, limited selfishness can be implemented by adding the following constraint to the default selfish overlay routing:

$$\forall \text{ link } l : \sum_{s,t} f_{st}(l)\ d_{st}^{overlay} \leq \theta$$

$$\theta = \max\{t(l)^{overlay} | \forall \text{ link } l\}$$

Here, θ is the maximum link load that the overlay generates in the previous run. The additional constraint limits the selfishness of overlay and prevents specific links to be overloaded.

6 Simulation

This section describes the simulation results of vertical interactions. We first compare MPLS and COPE as the underlay TE schemes, then we evaluate two overlay schemes: TE-aware overlay and selfish overlay.

We use General Algebraic Modeling System [19] to implement various optimization procedures for the experiments. Then the interaction between optimization programs is implemented by connecting the inputs and outputs of the GAMS programs through Perl scripts. Given that we run the optimization process for more than hundred iterations, we need a support of Condor [20], which is a specialized workload management system for compute-intensive jobs.

6.1 Data Set Description

We perform extensive experiments on a 14-node Tier-1 POP topology described in [21]. The underlay network topology is given in Fig. 2. We have done experiments with other topologies and observed qualitatively consistent results [18].

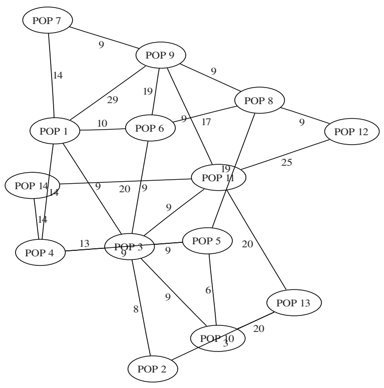

Fig. 2. 14-node Tier-1 backbone topology: each node represents a Point-of-Presence (POP) and each link represents the aggregated connectivity between the routers belonging to a pair of adjacent POPs. Four POPs (3,6,7,11) are used as overlay nodes.

On top of the physical network, we assume a four-node full-meshed overlay network. For the traffic matrix, we generate synthetic traffic demands using gravity model [21].

6.2 MPLS and COPE with Selfish Overlay

We start with the comparison between MPLS and COPE in the operator's viewpoint. We fix the overlay routing to be selfish and compare the performance of MPLS and COPE. In this experiment, we want to consider the performance of both TE and overlay.

For the COPE, we need a pre-specified penalty envelope value. We first calculate the value (1.9969) by running oblivious routing, and find the optimal routing which minimizes the oblivious ratio. Then by multiplying 1.1 to the optimal oblivious ratio, we set the penalty envelope value. We set 10% of the total traffic demand to be operated by the selfish overlay routing, and set the load scale factor to be 0.5, which means that the maximum link utilization is 50%, when all the demands use the default underlay routing without overlay's action.

The experiment results are shown in Fig. 3. We can observe that the COPE makes better interaction with selfish overlay. MPLS TE suffers from substantially large oscillation throughout the interaction, where COPE achieves almost stable performance with its maximum link utilization. Similarly, the dynamics of overlay latency is quite stable with the interaction of COPE. Moreover, the average latency sometimes gets improved by using COPE.

We have also conducted more experiments to explore the impact of overlay fraction and link utilization to the vertical interaction [18]. With the extensive simulation, we find COPE as a strong TE technique which achieves stable performance even the selfish overlay traffic dominates a significant portion of the total traffic demand.

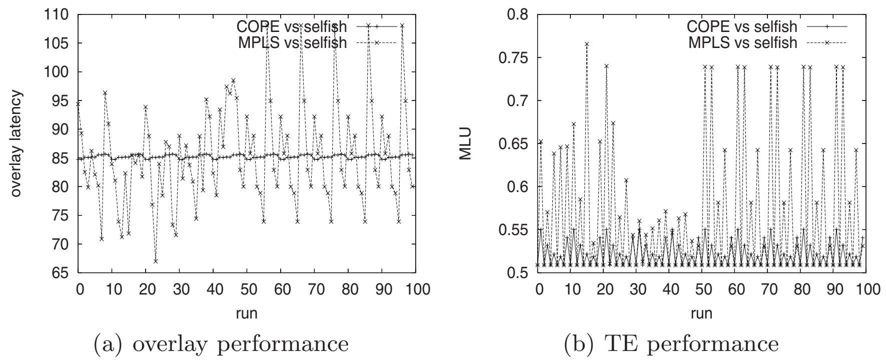

Fig. 3. MPLS and COPE with selfish overlay. 14-node topology with a 4-node overlay network, overlay fraction = 10%, load scale factor = 0.5. (MLU: Maximum Link Utilization).

6.3 TE-Aware Overlay and Selfish Overlay with MPLS

Now, we compare TE-aware overlay routing and selfish overlay routing to evaluate our proposed scheme. For the underlay routing, we again use MPLS and COPE. We first start the evaluation by comparing two overlay routings on top of MPLS TE.

In Fig. 4, we set 10% of the traffic to be operated by overlay routing and set the load scale factor to be 0.5. Considering the overlay latency, TE-aware overlay routing achieves more stable performance. Moreover, the average latency of TE-aware overlay is lower than that of selfish overlay. We can see that overlay routing can achieve better and stable routing by understanding the objective of underlay routing.

Considering the TE side, selfish overlay routing makes significant burden to the underlay routing because it generates substantially large amount of additional traffic. Thus, we can observe sudden increase of the maximum link utilization. However, TE-aware overlay limits its selfishness and tries to avoid a specific link to be over-loaded by its own traffic. Thus, the fluctuation of maximum link utilization is smaller when the overlay is TE-aware.

We have also analyzed the impact of the overlay fraction and link utilization to the interaction [18]. The experiments are conducted where the network is substantially congested (90% - 120%). Still, the proposed method makes better interaction than selfish overlay does.

With the extensive experiment results, we come up with the conclusion that TE-aware overlay routing generally makes stable interaction with MPLS TE. Selfish overlay routing experiences less predictable latency and it makes significantly large maximum link utilization of the network. However, we can achieve either convergence or regular pattern with the overlay latency when TE-awareness is embedded in overlay routing. Moreover, the network overhead to the TE is reduced by using the proposed overlay routing. Thus, TE-awareness obtains win-win game for each player in the presence of MPLS TE.

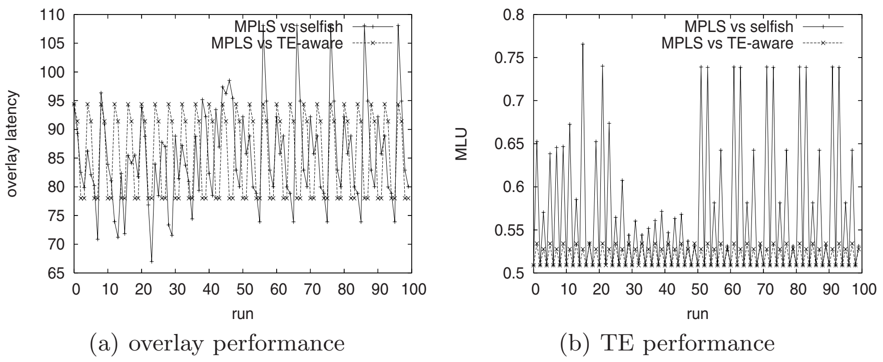

Fig. 4. TE-aware overlay and selfish overlay on MPLS. 14-node topology with a 4-node overlay network, overlay fraction = 10%, load scale factor = 0.5 (MLU: Maximum Link Utilization).

6.4 TE-Aware Overlay and Selfish Overlay with COPE

For the last experiment, we compare TE-aware overlay and selfish overlay on top of the COPE TE. In the previous experiments comparing COPE and MPLS, we have observed that COPE achieves better interaction with selfish overlay routing. Now, the question is how much gain we can get by using TE-aware overlay with COPE.

Figure 5 describes the experiment results, where 10% of the traffic is routed by overlay routing and load scale factor is 0.5. Similar to the experiment result of the previous sections, TE-aware overlay converges faster than the selfish overlay. However, comparing to the oscillation in MPLS experiments, we can see the performance variation is negligible. Similar patterns can be observed with the maximum link utilization.

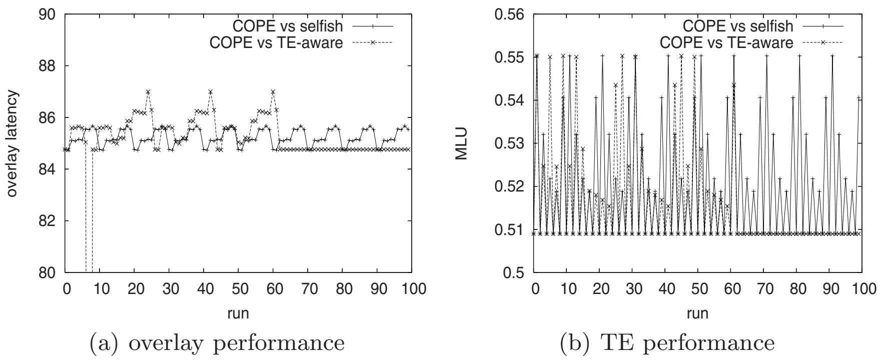

Fig. 5. TE-aware overlay and selfish overlay on COPE. 14-node topology with a 4-node overlay network, overlay fraction = 10%, load scale factor = 0.5 (MLU: Maximum Link Utilization).

6.5 Simulation Summary

Considering the overlay side, in all experiments, the latency experienced by TE-aware overlay is better than that of selfish overlay. The maximum link latency experienced by selfish overlay is sometimes twice larger than average latency. In some scenarios, selfish overlay latency keeps increasing as the interaction with underlay proceeds. TE-aware overlay shows similar pattern but makes convergence at considerably lower latency. Looking at the TE side, TE-awareness obtains either similar or better performance than selfishness.

Summarizing the interaction experiments with COPE, we can achieve considerably good interaction for both selfish overlay and TE-aware overlay. But, TE-aware overlay performs slightly better than selfish overlay routing.

7 Conclusion and Future Direction

In this paper, we improve the *vertical interaction* between overlay routing and traffic engineering by modifying the objectives of both parties. We propose *TE-aware overlay routing*, which takes traffic engineering's objective into account in overlay routing decision. We also suggest COPE as a strong traffic engineering technique, which makes a good interaction with unpredictable overlay traffic demands. We show the feasibility of the proposed methods with extensive simulation results.

The model used in this paper can be enhanced in several ways. First, we have captured vertical interaction within a single domain. We can extend the arguments to inter-domain level, where overlay nodes are spread across several autonomous systems and cooperate each other. Then the action of overlay will make interaction with inter-domain routing algorithms.

Secondly, multiple overlays may coexist on top of a shared underlay network. The decision of an overlay depends on probing information of logical paths. If overlays share some links but do not exchange the routing information of each other, currently observed link performance is likely be changed by the actions of other overlays. We can study this *horizontal interaction* among overlays.

Lastly, we use average latency as an indicator in TE-aware overlay routing. But it may be difficult to get this value in reality. Then we can use mechanism similar to TCP congestion control, where overlay additively increases the selfishness until it leads congestion and exponentially decreases the selfishness to ease the congestion level.

Acknowledgement

The authors would like to thank the discussion on this work with Yin Zhang.

References

1. Qiu, L., Yang, Y.R., Zhang, Y., Shenker, S.: On selfish routing in Internet-like environments. In: Proceedings of ACM SIGCOMM, Karlsruhe, Germany, August 2003, pp. 151–162 (2003)
2. Liu, Y., Zhang, H., Gong, W., Towsley, D.: On the interaction between overlay routing and traffic engineering. In: Proceedings of IEEE INFOCOM (2005)

3. Wang, H., Xie, H., Qiu, L., Yang, Y.R., Zhang, Y., Greenberg, A.: COPE: Traffic engineering in dynamic networks. In: Proceedings of ACM SIGCOMM (2006)
4. Chu, Y., Rao, S.G., Seshan, S., Zhang, H.: Enabling conferencing applications on the Internet using an overlay multicast architecture. In: Proceedings of ACM SIGCOMM, pp. 55–67 (2001)
5. Jannotti, J., Gifford, D.K., Johnson, K.L., Kaashoek, M.F., O'Toole, J.: Overcast: Reliable multicasting with an overlay network. In: Proceedings of Symposium on Operating Systems Design and Implementation (OSDI), pp. 197–212 (2000)
6. Subramanian, L., Stoica, I., Balakrishnan, H., Katz, R.H.: OverQoS: An overlay based architecture for enhancing internet QoS. In: Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI), pp. 71–84 (2004)
7. Akamai: `http://www.akamai.com`
8. Andersen, D.G., Balakrishnan, H., Kaashoek, M.F., Morris, R.: Resilient overlay networks. In: Proceedings of Symposium on Operating Systems Principles (SOSP), pp. 131–145 (2001)
9. Savage, S., Anderson, T., Aggarwal, A., Becker, D., Cardwell, N., Collins, A., Hoffman, E., Snell, J., Vahdat, A., Voelker, G., Zahorjan, J.: Detour: a case for informed internet routing and transport. Technical Report TR-98-10-05 (1998)
10. Chun, B.-G., Fonseca, R., Stoica, I., Kubiatowicz, J.: Characterizing selfishly constructed overlay routing networks. In: Proceedings of IEEE INFOCOM (2004)
11. Fortz, B., Thorup, M.: Internet traffic engineering by optimizing OSPF weights. In: Proceedings of IEEE INFOCOM, pp. 519–528 (2000)
12. Ruela, J., Ricardo, M.: MPLS - Multi-Protocol Label Switching. In: The Industrial Information Technology Handbook, pp. 1–9 (2005)
13. Stewart, I.J.W.: BGP4: inter-domain routing in the Internet. Addison-Wesley, Reading (1999)
14. Keralapura, R., Taft, N., Chuah, C.-N., Iannacco, G.: Can ISPs take the heat from overlay networks? In: Proceedings of the Third Workshop on Hot Topics in Networks (November 2004)
15. Dutta, P.K.: Strategies and games: Theory and practice (1999)
16. Nash, J.: Non-cooperative games. The Annals of Mathematics, 286–295 (September 1951)
17. Applegate, D., Cohen, E.: Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs. In: Proceedings of ACM SIGCOMM, pp. 313–324 (2003)
18. Lee, G.M., Choi, T., Zhang, Y.: Improving the interaction between overlay routing and traffic engineering. Technical Report TR-06-61, Department of Computer Sciences, University of Texas at Austin (2007)
19. GAMS: General Algebraic Modeling System: `http://www.gams.com`
20. Condor: `http://www.cs.wisc.edu/condor`
21. Medina, A., Taft, N., Salamatian, K., Bhattacharyya, S., Diot, C.: Traffic matrix estimation: existing techniques and new directions. In: Proceedings of ACM SIGCOMM, pp. 161–174 (2002)