DATE: Distributed Adaptive Traffic Engineering

Jiayue He
Dept. of EE, Princeton University
Email: jhe@princeton.edu

Mung Chiang,
Dept. of EE, Princeton University
Email: chiangm@princeton.edu

Jennifer Rexford,
Dept. of CS, Princeton University
Email: jrex@cs.princeton.edu

I. Introduction

Congestion in the network causes poor throughput and long delays for end users, and also leads to an inefficient usage of network resources. In the Internet today, end users run the Transmission Control Protocol (TCP) to adapt their sending rates to congestion. Independently, Internet Service Providers (ISPs) monitor their networks for signs of overloaded links and adapt routing to alleviate congestion in a process known as traffic engineering (TE). The current state of the art for TE occurs at the timescale of hours and is centralized [1], [2], [3]. As traffic patterns shift on the order of seconds, a dynamic and distributed TE is an attractive alternative. Yet, previous research on load-sensitive routing [4], [5] has shown that dynamic, distributed routing is prone to instability. More recently, there have been proposals for online multipath traffic engineering, namely MATE [6] and TeXCP [7]. They have both made strides in showing it is possible to achieve stable distributed dynamic traffic engineering.

Still, both MATE and TeXCP ignore the congestion control. We propose a distributed adaptive traffic engineering (DATE) algorithm. DATE stands out from previous work by simultaneously satisfying the needs of users and ISPs. Each user wants to maximize their own utility (e.g.a function of throughput or delay) and ISPs want to efficiently use network resources. These needs can be at odds with each other, and DATE aims to balance them. There is an alternative body of work which studies multipath congestion control, as in [8], [9], [10]. Those works are purely theoretical, however, and ignore implementation and deployment issues.

The design goals for DATE can be summarized as follows:

1) **Distributed:** in order to adapt at a small timescale, the algorithm should not require centralized computation or a full network view.
2) **Optimal:** the algorithm must maximize a weighted sum of user and operators's objectives.
3) **Stable:** the algorithm should converge quickly and not have oscillatory behavior.
4) **Implementable:** only a limited number of changes to a limited number of routers is required.
5) **Efficient:** the computations should be efficient and not prevent the routers from performing other functions.

In the following section, we introduce the notion of consistency price to lead to a distributed gradient algorithm that avoids the classical instability problem associated with dynamic routing. Section II describes how DATE works and Section III addresses implementation issues. We talk about ongoing plans for simulation and experimentation in Section IV. Finally, we provide the theoretical background to prove the convergence of DATE to a global optimum TE in Section V and conclude in Section VI.

This work was supported NSF grant CNS-0519880 and Cisco grant GH072605.

II. DATE Algorithm

In this section we describe the DATE algorithm, as graphically illustrated in Figure 1.

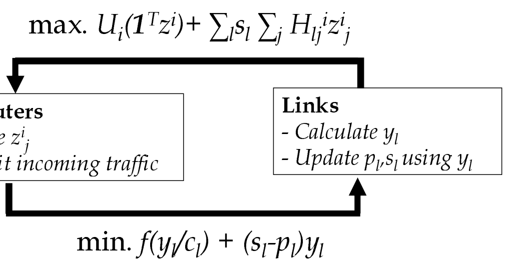
max. $U_i(\mathbf{1}^T z^i) + \sum_l s_l \sum_j H_{lj}^i z_j^i$
Edge Routers
- Calculate z_j^i
- Rate limit incoming traffic
Links
- Calculate y_l
- Update p_l, s_l using y_l
min. $f(y_l/c_l) + (s_l - p_l) y_l$

Fig. 1. A graphical view of DATE algorithm.

Like its TE predecessors, DATE is limited to a single AS where the topology is known and routing can be controlled. DATE denotes each TCP session i's concave utility function as $U_i(x_i)$, where x_i is the transmission rate. Following approaches in TE studies [1], [3], candidate routing solutions are evaluated based on a convex, increasing cost function $f(u_l)$, where u_l is the link utilization.

The DATE algorithm uses *two* kinds of link prices: (i) congestion price $\{p_l\}$ and (ii) consistency price $\{s_l\}$. Congestion prices implicitly exist in the Internet e.g. queuing delay, and they ensure the capacity constraints c_l are not violated at equilibrium. One potential reason for instability in a dynamic distributed algorithm is that the routing algorithm reacts only to the *measured* load on each link, rather than some notion of a reasonable *target* load. Here we introduce the notion of consistency price to ensure a matching between measured link load and target link load at equilibrium.

Let the underlay topology be defined by H, where j indexes to the multiple paths of i. Let the time for feedback from each core router be T_p, note that T_p must be greater than the round trip time (RTT) plus the time it takes for each core router to compute its feedback.

DATE Algorithm
Core Routers:

- Core Router Problem:

$$y_l(t+T_p) = \text{minimize}_{y_l} f(y_l/c_l) + (s_l(t) - p_l(t))y_l.$$

where y_l is the target load.

- Congestion Price Update:

$$p_l(t+T_p) = [p_l(t) - \beta_p(c_l - y_l(t))]^+,$$

where β_p is the congestion price step-size. Since, $p_l \geq 0$, it must be mapped to a non-negative value.

- Consistency Price Update:

$$s_l(t+T_p) = s_l(t) - \beta_s \left(\sum_i \sum_j H^i_{lj} z^i_j(t) - y_l(t) \right),$$

where β_s is the consistency price step-size.

Edge Routers:

$$z^i(t+T_p) = \text{maximize}_{z^i} U_i(\mathbf{1}^T z^i) + \sum_l s_l(t) \sum_j H^i_{lj} z^i_j$$

where z^i_j is the amount of load that TCP session i places on its j^{th} path and $x_i = \sum_j z^i_j$.

In a DATE network, edge and core routers work together to balance load, limit the incoming traffic rate and route around failures. Every core router first measures the link load of all the links connected to it. Then it computes the new target load y_l for each link using information from both prices which are updated using the gradient method, based on local information at each link.

Each edge router probes for explicit feedback s_l from the routers along the paths used by each ingress-egress (IE) pair. Using this information, each edge router tries to maximize the utility for the end hosts indexed by i connected to it. This local maximization is conducted over a vector z^i for each TCP session i, as opposed to only a scalar x_i as in the standard TCP congestion control. This captures the edge routers balancing load over multiple paths. Each edge router then rate limits the incoming traffic rate to x_i for an end-host.

Finally, since there is explicit feedback expected every T_p, if feedback is not received in a timely fashion from a particular link, that link is most likely failed or heavily congested. Hence the explicit feedback mechanism also helps to divert traffic from failed or heavily congested paths quickly.

III. Implementation and Deployment

This section examines the practical considerations for an ISP to deploy DATE.

A. Implementation Possibilities

For this section, we assume IE paths are pinned using MPLS Label Switched Paths (LSPs). Here are some existing technology which will establish the foundation for deployment.

1) *Number of LSPs:* Each ISP has 200-300 egress points, so if we create LSPs between each IE pair, there is quite a lot of such LSPs. On the other hand, since traffic on the ISP backbone moves from essentially from one PoP to another, we only need to create DATE tunnels between PoPs rather than between ingress-egress routers. For an ISP, there are only 20-100 PoPs, making the number of tunnels much more tractable.
2) *Traffic Splitting:* DATE requires edge routers to split traffic among the LSPs connecting an IE pair. Current ISP-class routers can split traffic to a destination between as many as 16 LSPs.
3) *Utilization Estimates:* DATE needs all routers to estimate link utilization. This is currently already done by the data plane of all routers. Basically, all current routers keep track of the amount of traffic sent on each data bus in a counter. These counters can be read every T_p to get a utilization estimate. In fact, routers already do this to support the Simple Network Management Protocol.

In general, DATE needs one extra function at the edge routers: per flow traffic policing. The edge routers determine the allowed rate of each flow and limit the incoming traffic by dropping packets sent above the allowed rate. The existing congestion control mechanism should let the end hosts adapt their rate when their packets are getting dropped.

Price updates are straight forward matrix multiplications. Updating y_l and z^i_j involve solving a convex optimization problem which is also not computationally intensive. In terms of the calculations involved for edge and core routers, there are three potential ways to implement them. We are currently in discussion with Cisco with regards to which option is the most feasible.

1) *Alternative Entity:* One possibility is to have all the link utilizations collected by a central routing platform such as the one proposed in [11]. This alternative entity can perform all the per link calculations to generate the consistency prices per link and calculate the traffic per path for each TCP session. The plus side is that there will be no extra load placed on the CPU of routers, but it requires the deployment of an alternate entity.
2) *Edge Routers:* Another possibility is for the core routers to send link utilization to the edge routers, and for edge router CPUs to do all the computation.
3) *Core Routers CPU:* The third approach is for each core router to update its own target load, congestion price, and consistency price. These computations can be handled by it's control plane (CPU). Instead of the link utilization, the consistency price is fed back from each core router to the edge routers every T_p. This introduces less computational overhead, since consistency price wouldn't be repeatedly computed by many IE pairs that direct traffic over the same link. The trade-off is that more software functionality needs to be placed in all core routers.

B. Incremental Deployment

While DATE and TeXCP are quite different approaches, they both establish IE pairs, fix the usable paths and get link utilization as an explicit feedback per link. Deploying TeXCP is simpler than DATE as it only requires software modification at the edge routers and no hardware modifications. The relative simplicity of TeXCP is not surprising given its objective is limited to minimizing the maximum link utilization in a given network. Hence, TeXCP can be thought of as an incremental step towards deploying DATE.

IV. Ongoing Work

Before actual deployment, it is essential to thoroughly test DATE's performance in realistic network environments. Similar to prior work [7], we use Rocketfuel backbone topologies inferred with annotated with inferred weights and link latencies [12]. We also estimate the number of users between every pair of PoPs. The following experiments are planned:

- **Convergence Properties:** Convergence time and the smoothness of convergence is an important aspect which needs to examined. Potential factors which affect this include: step size of congestion price, step-size of consistency price, frequency of feedback and feedback delay.
- **Practical Concerns:** How long would it take current routers to do the necessary computations within its own CPU? This will help us determine which of the three potential ways for implementation is the most deployable. In general, the frequency of feedback required and the number of LSPs needed would also have a big effect on whether DATE can be deployed. Therefore, the minimum frequency for feedback and the minimum number of LSPs needed would be useful parameters to determine.
- **Comparison with Other Work:** We can compare the convergence properties of DATE with TeXCP and MATE. We can also compare with the control theoretical work [9], [8], [10].

V. Theoretical Background

The objective of each individual user is to maximize its own utility. This is well captured in today's congestion control mechanism, as shown e.g. in [13], [14]. Reverse engineering shows in a network where each TCP session is indexed by i, the following optimization is implicitly solved by TCP:

$$\begin{array}{ll} \text{maximize} & \sum_i U_i(x_i) \\ \text{subject to} & Rx \ \preceq \ c. \end{array} \tag{1}$$

The goal is to maximize aggregate user utility $U_i(x_i)$ by varying source rate x with routing R fixed, subject to the linear flow constraint that link loads cannot exceed capacity.

On the other hand, the operators running a backbone network measures the offered load between each ingress-egress pair x_i. Based on the traffic matrix, the operators try to find the best routing matrix R to minimize network congestion. The utilization of link l is $u_l = \sum_i R_{li} x_i / c_l$. To penalize routing configurations that congest the links, candidate routing solutions are evaluated based on a cost function $f(u_l)$ that increases steeply as u_l approaches 1. The following formulation has been discussed in [1], [3]:

$$\text{minimize} \quad \sum_l f(\textstyle\sum_i R_{li} x_i / c_l). \tag{2}$$

By considering the total link cost rather than trying to minimize a single bottleneck, the optimization framework would prefer a solution that utilizes a single link at 91% over one that loads many links at 90%.

A natural formulation that accounts for the network operator's goals in addition to end-user utilities is:

$$\begin{array}{ll} \text{maximize} & \sum_i U_i(x_i) - \sum_l f(\sum_i R_{li} x_i / c_l) \\ \text{subject to} & Rx \preceq c, \ x \succeq 0. \end{array} \tag{3}$$

This objective favors a solution with that provides both high aggregate utility and a low overall network congestion, to satisfy the goals of users and operators alike.

The following theorems can be proved (skipping the proof due to space limitation).

Theorem 1: The distributed algorithm DATE converges to a joint optimum of (3) for sufficiently small step sizes (β_p, β_s).

Theorem 2: When sessions arrive according to Poisson distribution each carrying a finite workload exponentially distributed, DATE Algorithm remains stochastically stable if the traffic load is within the interior of the feasible region in (3).

VI. Conclusions

In this paper we present DATE, a distributed, stable and optimal algorithm. We have also considered DATE's implementation issues and the possibility of incremental deployment. Finally, we have outlined plans for evaluating convergence time and efficiency through simulation.

References

[1] B. Fortz and M. Thorup, "Optimizing OSPF weights in a changing world," *IEEE JSAC*, vol. 20, pp. 756–767, May 2002.

[2] D. Mitra and K. Ramakrishna, "A Case Study of Multiservice Multipriority Traffic Engineering Design," in *Proc. IEEE GLOBECOM*, December 1999.

[3] J. Rexford, "Route optimization in IP networks," in *Handbook of Optimization in Telecommunications*, Springer Science + Business Media, February 2006.

[4] J. M. McQuillan and D. C. Walden, "The ARPA network design decision," *Computer Networks*, vol. 1, pp. 243–289, August 1977.

[5] Z. Wang and J. Crowcroft, "Analysis of shortest-path routing algorithm in dynamic network environment," *ACM SIGCOMM Computer Communication Review*, vol. 22, pp. 63–71, April 1992.

[6] A. Elwalid, C. Jin, S. H. Low, and I. Widjaja, "MATE: MPLS Adaptive Traffic Engineering," in *Proc. IEEE INFOCOM*, April 2001.

[7] S. Kandula and D. Katabi, "Walking the Tightrope: Responsive Yet Stable Traffic Engineering," in *Proc. ACM SIGCOMM*, August 2005.

[8] X. Lin and N. B. Shroff, "Utility Maximization for Communication Networks with Multi-path Routing," *IEEE Trans. Automatic Control*, 2006. To appear.

[9] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Overlay TCP for Multi-Path Routing and Congestion Control," in *Proc. IMA Workshop on Measurement and Modeling of the Internet*, January 2004.

[10] R. J. Gibben and F. Kelly, "On packet marking at priority queues," *IEEE Trans. Automatic Control*, vol. 47, pp. 1016–1020, December 2002.

[11] A. Greenberg, G. Hjalmtysson, D. A. Maltz, A. Meyers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang, "A clean slate 4D approach to network control and management," *ACM SIGCOMM Computer Communication Review*, October 2005.

[12] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Inferring Link Weights using End-to-End Measurements," in *Proc. Internet Measurement Workshop*, 2002.

[13] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Trans. Networking*, vol. 11, pp. 525–536, August 2003.

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