Joint Server Selection and Routing for Geo-Replicated Services

Srinivas Narayana, Wenjie Jiang, Jennifer Rexford, Mung Chiang
Princeton University

***Abstract*—The performance and costs of geo-replicated online services depend on which data centers handle user requests, and which wide-area paths carry traffic. To provide good performance at reasonable cost, service providers adapt the *mapping* of user requests to data centers (*e.g.,* through DNS), and *routing* of responses back to users (*i.e.*, through multi-homed route control).**

Mapping and routing are typically managed independently, with mapping having limited visibility into routing decisions, response path latencies, and bandwidth costs. However, poor visibility and uncoordinated decision-making can lead to worse performance and higher costs when compared to a joint decision. In this paper, we argue that mapping and routing should continue to operate modularly, but cooperate towards service-wide performance and cost goals. Our main contribution is a distributed algorithm to steer cooperating, yet functionally separate, mapping and routing provably towards a globally optimal operating point. Trace-based evaluations on an operational CDN show that the algorithm converges to within 1% of optimum in 3-6 iterations.

I. Introduction

Online service providers (OSPs) carefully optimize the performance their customers experience, since even small increases in user-perceived latency can have significant impact on revenue [1]. To improve performance and reliability, OSPs run services out of multiple geographically distributed service locations (*e.g.,* data centers), each typically peering with multiple ISPs. However, good performance must be balanced against operational costs resulting from the electricity and bandwidth needed to serve large amounts of data on a daily basis [2], [3].

Client traffic loads, path latencies, and electricity and bandwidth costs, vary over space and time. To provide good performance at reasonable costs, OSPs adapt the wide-area paths carried by traffic in two key ways (Fig. 1)—*mapping* requests to specific locations (*e.g.,* DNS redirection [4], [5]), and *routing* responses back to clients through peer selection at the egress to the Internet (*e.g.,* multi-homed routing [2], [6]). These 'knobs' allow an OSP to exploit path and cost diversity to adapt to time-varying demands, path performance, and costs.

To our knowledge of the state of the art, mapping and routing are managed independently: mapping decisions are made without full visibility into the path metrics, loads and decisions of the (multi-homed) routing system. However, as we show in §II, misaligned objectives, lack of information visibility and uncoordinated decision-making can lead to bad performance and high costs. For example, the mapping may direct user requests to locations with limited upstream bandwidth, poor routing performance [4], [7], or expensive ISP connectivity [2]. As a Google CDN study [4] shows, clients can experience latencies inflated by several tens of milliseconds even when served by the geographically closest node, in part due to routing inefficiencies. In such cases, better visibility into routing decisions and path performance can allow poorly performing clients to be mapped to other locations.

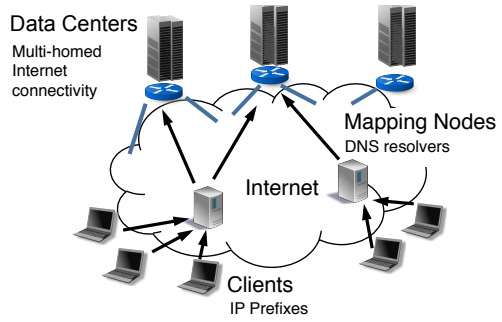


Fig. 1. An online service with multiple data centers across the Internet.

A natural approach to improve the situation is to jointly compute mapping and routing decisions—after all, the OSP may have administrative control over both. However, the ability to *separately compute* decisions is desirable for two key reasons. First, an OSP may not be running its own infrastructure on all parts of its global traffic management system. For example, an OSP may employ an external mapping service, *e.g.,* [5], [8]. In such cases, the OSP may only dictate high-level policies but may be unable to compute or configure the mapping decisions directly. Second, a large OSP may already have operational mapping and routing infrastructure working independently [4], [7]. Hence, instead of building a monolithic system, we argue that mapping and routing should continue to operate *modularly*, but coordinate to achieve the service-wide performance and cost of a *joint* system.

Our goal is to construct a *distributed algorithm* to coordinate *administratively separate* mapping and routing systems, and provably achieve *global* performance and cost goals. Unlike prior works on OSP joint traffic engineering [2], [7], [9], this algorithm retains the modularity of mapping and routing while addressing major concerns that couple their decisions (*e.g.,* link capacities). Our contributions are as follows.

Identifying challenges in coordination. We show why a naive distributed algorithm, where mapping and routing iteratively optimize decisions—even with full information visibility—can lead to bad performance and high costs (§II).

Distributed mapping and routing algorithm. We present a distributed algorithm that systematically addresses the causes of suboptimality, hence allowing modular mapping and routing to provably converge to a global optimum (§IV). The solution is summarized in Fig. 2. Our optimization model captures con-

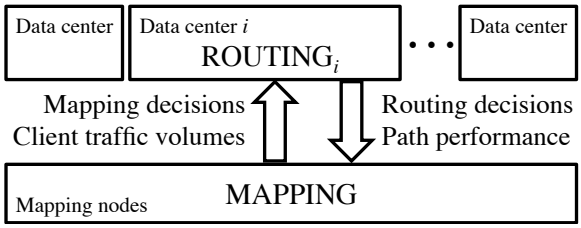


Fig. 2. Distributed mapping-routing algorithm (§IV). Data center i solves local problem **ROUTING**$_i$ to obtain multi-homed routing decisions, and mapping nodes solve **MAPPING** to compute global mapping decisions. They then exchange measured traffic volumes, path performance, and computed mapping and routing decisions, and iteratively reoptimize until convergence.

siderations like traffic demands, path performance, bandwidth costs, mapping policies [5], [8], and link capacities (§III).

Evaluation on CoralCDN traces. We evaluate the distributed algorithm on traces from CoralCDN [10], an operational content distribution network, and show that it converges to within 1% of the global optimum in 3-6 iterations (§V).

II. Coordinating Mapping & Routing

Performing client-mapping and network-routing independently can miss opportunities to improve performance or reduce costs. Through an example OSP which has two data centers (DCs) each with two ISP links, and one client IP prefix, we highlight some challenges that hinder independent mapping and routing systems from arriving at good collective decisions.

Misaligned objectives can lead to suboptimal decisions. Typically, mapping and routing systems work with different objectives—*e.g.,* mapping performs latency and load-based DC selection, while routing considers end-to-end performance and bandwidth costs to choose a peer to forward responses. In Fig. 3(a), the routing system at each DC picks the peer with the least cost per unit bandwidth, while the mapping system picks the DC with the least propagation delay given current routing choices—resulting in a situation where the service neither has globally optimal cost nor optimal latency.

Incomplete visibility can lead to suboptimal decisions. In Fig. 3(b), the mapping and routing systems both optimize latency, and users are sending traffic at the rate of 5Gb/s. Without information on link loads and capacities, the mapping system directs too much traffic to the DC on the left, leading to large queuing delays and packet losses. If the mapping system uses information about link capacities, it could easily direct most traffic to the alternate DC with ample spare capacity, and only slightly worse latency.

Coupled constraints can lead to suboptimal equilibria. Even if mapping and routing have aligned objectives (*e.g.,* minimize latency) and complete visibility, optimizing their decisions separately taking turns can still lead to globally suboptimal situations. In Fig. 4(a), mapping and routing are locally optimal given each other's decisions, as traffic is served through the least latency paths respecting link capacities. However, in the globally optimal allocation, all traffic is served by the DC on the right, using both peers.

Bad routing decisions for prefixes with no traffic contribution to a DC can lead to suboptimal equilibria. Consider the scenario in Fig. 4(b), where a mapping initialization (*e.g.,* from geo-proximity) directs all traffic to the DC on the left. No traffic from this user reaches the other DC, so all routing decisions here are equally good. Suppose this DC chooses to route traffic through the 100ms path. Then, these mapping and routing decisions are locally optimal to each other—preventing the routing from exploring the globally optimal 50ms path.

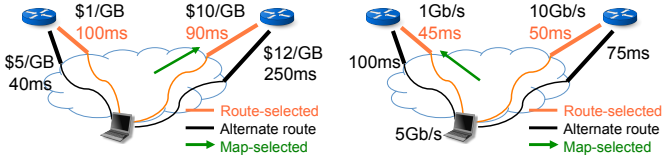


Fig. 3. (a) Mapping is suboptimal because of misaligned objectives; (b) Mapping is suboptimal because routing does not share link-related information.

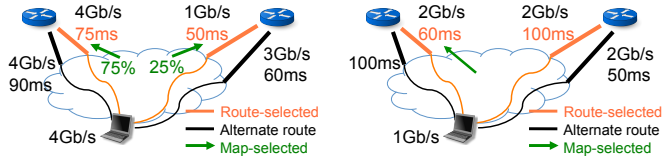


Fig. 4. (a) Separate mapping and routing can be inefficient under coupled constraints; (b) Routing is suboptimal when mapping sends no traffic to a DC.

III. OSP & Optimization Models

In this section, we introduce the model for (i) OSP mapping and routing decisions, (ii) performance and cost goals, and (iii) joint cost-performance optimization.

A. Online Service Provider Model

Network. The OSP network model is summarized in Fig. 1. An online service runs in a set I of service locations (*i.e.*, data centers, caches, replicas, availability zones). Every location i is connected to the Internet through a set of ISP links, which we denote by J_i. Let C be the set of clients, where each client c is an aggregate of real users (in our experiments, these are IP prefixes). Clients can be directed to different locations depending on their proximity and the current load on the compute infrastructure. Such mapping of users to locations occurs through a set of "mapping nodes", such as authoritative DNS servers or HTTP proxies. Once a request is serviced at location i, the egress router picks one or more ISP-links $j \in J_i$ to send responses. The total traffic that a link j can support is limited by its capacity cap_{ij}.

Mapping Decisions (choice of service location). We denote α_{ic} as the proportion of traffic from client c mapped to location i. We require that $\sum_i \alpha_{ic} = 1$ for all c, where $\alpha_{ic} \in [0,1]$. Different users in the same IP prefix c may be directed to different locations simultaneously. The mapping decisions are realized by a flexible mapping service *e.g.,* [5], [8], [11]. Each user is served by its local mapping node, *e.g.,* a local DNS server, and mapped to the designated service location.

Routing Decisions (choice of egress ISP). Each service location is connected to the Internet by several links to different ISPs. We use the index j to denote a link that connects a service location to an ISP. For every location i, we denote by

β_{ijc} the proportion of traffic for client c served by location i that is sent over link $j \in J_i$, where $\sum_{j:j\in J_i} \beta_{ijc} = 1$. The response-routing decision only controls the first hop, and not the entire path. Typical routing decisions β_{ijc} are often integers $\{0,1\}$, since one next-hop is chosen to reach each client c. We relax this constraint and allow OSPs to freely split client traffic across links. In practice, fractional routing can be realized by hash-based splitting or multi-homing agents [12]. Each data center is a stub autonomous system with no transit service, hence routing changes do not need BGP to reconverge.

B. Performance and Cost Goals

User-perceived latency is a key performance metric for online services, since even small increments can have significant effects on revenue [1]. User-perceived latency can depend on round trip delays between users and data centers, processing time within the compute infrastructure, TCP dynamics, and even browser page loading time. In this paper, we focus on optimizing the *round trip latency* for user requests, which impacts the completion time of short TCP flows [13], much more than metrics like bandwidth and packet loss.

Average end-to-end latency objective. We use average end-to-end path RTT (ms/request) as our performance metric. Path latencies depend on which locations handle requests, which wide-area paths deliver traffic, and packet queuing and transmission delays along paths. A service provider can obtain statistically averaged latencies through active measurements [14] or latency prediction tools [15]. We use perf_{ijc} to denote the latency from location i to client c, when i picks link $j \in J_i$ to deliver traffic. Then the performance objective is:

$$\textbf{perf} = \left(\sum_{c\in C} \text{vol}_c \sum_{i\in I} \alpha_{ic} \sum_{j\in J_i} \beta_{ijc} \text{perf}_{ijc} \right) / \sum_c \text{vol}_c,$$

where vol_c is the total request volume from client c. We ignore the impact of OSP traffic shifts on latency as long as data center-ISP links operate safely under capacity. This is a standard simplification used in prior works [2], [6].

Average cost per request objective. Operational costs are an important consideration for OSPs [2], [3]. In this paper we focus on ISP bandwidth costs, which can be significant on its own for large OSPs [2]. For tractability, we assume a cost function which is linear on the amount of data sent. The cost metric is average cost per request ($/request):

$$\textbf{cost} = \left(\sum_{c\in C} \text{vol}_c \sum_{i\in I} \alpha_{ic} \sum_{j\in J_i} \beta_{ijc} \text{price}_{ij} \right) / \sum_c \text{vol}_c$$

where price_{ij} is the cost per request on link $j \in J_i$ of data center i. In practice ISPs employ complex pricing functions (*e.g.,* 95th percentile charging). However optimizing a linear cost on every charging interval can reduce monthly 95th percentile costs [2].

C. Joint Cost-Performance Optimization

The goals of minimizing latency and minimizing costs can often be at odds. For example, an ISP with richer Internet peering may offer lower latencies to more clients, albeit at higher cost, than its competitors [2]. To strike a balance between cost and performance, we introduce a weight K that reflects the amount of additional costs ($/request) that an OSP is willing to pay for one unit of performance improvement (ms/request). Hence, the OSP's goal is to minimize $\textbf{cost} + K \cdot \textbf{perf}$, which captures pareto-optimal tradeoffs between **perf** and **cost**.

For additional flexibility, we allow OSPs to express mapping policies in terms of *traffic split ratios* w_i (with tolerance ε_i) or *absolute request caps* B_i per data center i. This allows an OSP to balance load between data centers in a weighted round robin fashion [8] or limit load at certain sites [5].

We formulate the joint optimization problem as follows:

GLOBAL

$$\begin{aligned}
\text{minimize} \quad & \textbf{cost} + K \cdot \textbf{perf} && \text{(1a)}\\
\text{subject to} \quad & \sum_{c,i:j\in J_i} \text{vol}_c \alpha_{ic} \beta_{ijc} \le \text{cap}_{ij},\ \forall j && \text{(1b)}\\
& \left| \frac{\sum_c \text{vol}_c \alpha_{ic}}{\sum_{c'} \text{vol}_{c'}} - w_i \right| \le \varepsilon_i,\ \forall i && \text{(1c)}\\
& \sum_c \text{vol}_c \alpha_{ic} \le B_i,\ \forall i && \text{(1d)}\\
& \sum_i \alpha_{ic} = 1,\ \forall c && \text{(1e)}\\
& \sum_{j\in J_i} \beta_{ijc} = 1,\ \forall i,c && \text{(1f)}\\
\text{variables} \quad & \alpha_{ic} \ge 0,\ \forall i,c,\ \beta_{ijc} \ge 0,\ \forall i,j,c &&
\end{aligned}$$

where (1b) ensures the aggregate link traffic does not exceed capacity [2], [5], [9], [16]. Constraints (1c) and (1d) enforce traffic split ratios and bandwidth caps respectively. Constraints (1e) and (1f) ensure that proportions sum up to one. The problem (1) is a linear program on α and β separately, but non-convex when *both* are variables. As such, it cannot be solved using standard convex programming. However, it can be converted into an equivalent LP (whose variables couple mapping and routing) and solved efficiently [17, Appendix A].

IV. Optimal Distributed Algorithm

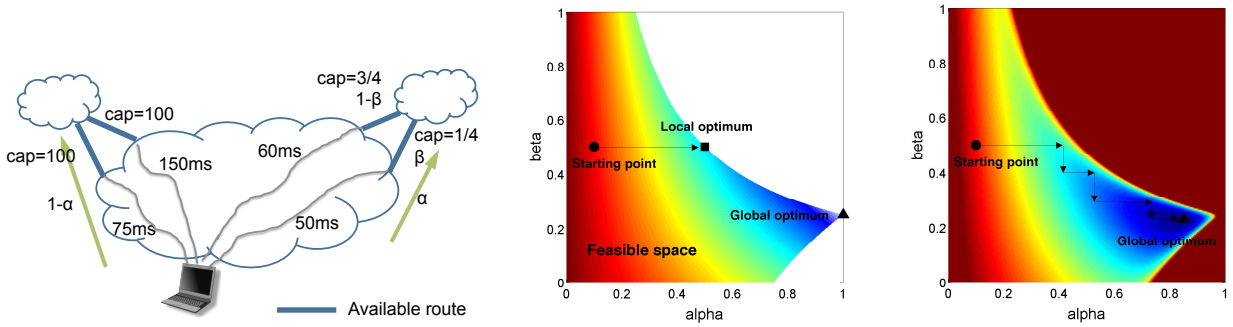
We design effective ways to avoid suboptimal equilibria (§IV-A) and present an optimal distributed algorithm (§IV-B).

A. Avoiding Suboptimal Equilibria

A straightforward distributed solution to the joint problem (1) is to alternate between the mapping nodes optimizing over α (given routing decisions β), and the routing nodes optimizing over β (given mapping decisions α). However, such separate optimizations often lead to local optima (§II).

We systematically address the causes of suboptimality from §II. First, we align mapping and routing objectives explicitly by constructing both to solve the joint problem (1) iteratively. Further, we mandate that each system has visibility into information needed to solve the joint problem, by requiring specific local measurements and exchange of statistics (§IV-B).

Next, we illustrate how we mitigate the *coupled constraints* problem through an example network in Fig. 5 (a). Let α and β be the mapping and routing decisions respectively. If the client has one unit of demand, optimal latency is achieved when $\alpha = 1, \beta = 1/4$. However, by separate optimizations, we can end up in local optima, *e.g.,* $\alpha = 1/2, \beta = 1/2$, which are valid but globally suboptimal mapping and routing profiles.



(a) Example network (b) **GLOBAL** (c) **GLOBAL**$_\Phi$

Fig. 5. An example of local optima with separate optimization.

Fig. 5 (b) visualizes how the local optimum is reached from an initial starting point. The color-shaded region represents the feasible decision space. The curvy boundaries reflect the capacity constraints of the 50ms and 60ms links. The color coding means the objective value, *i.e.*, latency, where red is high and blue is low. All points on these boundaries except at $\alpha = 1$ are local optima, which can be very inefficient. In general, the feasible space—determined by link capacity and mapping policy constraints—is non-convex.

Motivated by the observation that an ill-shaped feasible region does not enable efficient distributed optimization, we aim to work around the boundaries implied by the hard capacity constraints. (We show later that these are in fact the only constraints leading to suboptimal equilibria [17].) We relax the link capacity constraint by introducing a penalty function. In particular, we utilize a piece-wise linear function $\Phi_{ij}(\cdot)$ often used in ISP traffic engineering [18] to capture costs due to high link utilization. The penalty term Φ improves the prediction of client performance by capturing the effects of link congestion on queuing delay. In the refined global optimization, namely **GLOBAL**$_\Phi$, we replace the objective (1a) by

$$\mathbf{obj}_g(\alpha,\beta) = \frac{\sum_{ijc}\alpha_{ic}\beta_{ijc}\mathrm{vol}_c\left(\mathrm{price}_{ij}+K\cdot\mathrm{perf}_{ijc}\right)}{\sum_c \mathrm{vol}_c} + \sum_{ij}\Phi_{ij}\left(\sum_c \alpha_{ic}\beta_{ijc}\mathrm{vol}_c\right)/|J|, \quad (2)$$

which can be viewed as a joint cost-performance optimization with link congestion consideration. The constraint (1b) that caused the suboptimal equilibria is now removed from the problem. Figure 5(c) shows that the solution of this refined problem formulation converges to the global optimum by alternately fixing one dimension and optimizing the other. The whole decision space is now feasible, but the violation of link capacity will incur a high cost. Note that the global optima shown in Figures 5(b) and 5(c) are different, as the congestion cost is not considered in the original formulation.

B. Distributed Mapping and Routing

We present a distributed solution that builds on the existing practice of computing mapping and routing decisions independently. The key idea is to design optimization problems for each system, *i.e.*, mapping and routing, in which (i) they have aligned objectives, and (ii) each system computes only its own (local) decision variables, using local constraints and exchange of appropriate statistics. In particular, mapping nodes collectively solve the following mapping problem:

MAPPING(β)

$$\begin{aligned}
\text{minimize} \quad & \mathbf{obj}_g && (3a)\\
\text{subject to} \quad & \left|\frac{\sum_c \alpha_{ic}\mathrm{vol}_c}{\sum_c \mathrm{vol}_c} - w_i\right| \le \varepsilon_i,\ \forall i && (3b)\\
& \sum_c \alpha_{ic}\mathrm{vol}_c \le B_i,\ \forall i && (3c)\\
& \sum_i \alpha_{ic} = 1,\ \forall c, && (3d)\\
\text{variables} \quad & \alpha_{ic} \ge 0,\ \forall\ i,c
\end{aligned}$$

Further, edge routers together solve the following problem:

ROUTING(α)

$$\begin{aligned}
\text{minimize} \quad & \mathbf{obj}_g && (4a)\\
\text{subject to} \quad & \sum_{j\in J_i}\beta_{ijc} = 1,\ \forall i,c && (4b)\\
\text{variables} \quad & \beta_{ijc} \ge 0,\ \forall\ i,j,c
\end{aligned}$$

Note that **ROUTING** does need to know about **MAPPING**'s load balancing policies. Further, **MAPPING** and **ROUTING** are both convex optimizations, which can be efficiently solved.

A distributed solution, the *alternate projection* algorithm, proceeds as follows: given routing decisions β as input, mapping nodes solve (3) and optimize over α, and given mapping decisions α as input, edge routers solve (4) and optimize over β. The two steps are carried out iteratively until solutions converge. We allow mapping and routing problems to be solved at different time-scales, with the only requirement that (3) does not start until (4) is fully solved, and vice versa. This is easily ensured by having each component wait to receive inputs from the other before solving its own local optimization problem.

However, we showed in Figure 4(b), in general the alternate projection algorithm may still lead to suboptimal equilibria, when some clients send no traffic to a data center. To mitigate this *no traffic* problem, we introduce a refinement to routing in

addition to the optimality of (4). (We later prove the optimality of such a refined routing strategy [17].) In practical computation of routing decisions, we introduce an approximation to (4) by incrementing an infinitesimally small demand to a client-server pair with zero traffic, *i.e.*, $\alpha_{ic} \leftarrow \delta$ if $\alpha_{ic} = 0$, where δ is a small positive constant [19]. This approximation is often used to ease the computation so that standard optimization techniques can be applied. However, clients do not need to send real traffic, making this approach practically appealing.

We show that the alternate projection algorithm with the refinements introduced above provably converges to the global optimum of $\mathbf{GLOBAL}_\Phi$ [17, Appendix B]:

Theorem *1:* Alternate projections of **MAPPING** and **ROUTING** converge to an optimal solution of $\mathbf{GLOBAL}_\Phi$.

Note that **ROUTING** can be further decomposed into local versions at each data center, as both its objectives and constraints can be decoupled, *e.g.,* $\mathbf{obj}_g = \sum_i \mathbf{obj}_g^{(i)}$, where

$$\mathbf{obj}_g^{(i)} = \sum_{j \in J_i, c} \alpha_{ic}\beta_{ijc}\mathrm{vol}_c \left(\mathrm{price}_{ij} + K \cdot \mathrm{perf}_{ijc}\right) / \sum_c \mathrm{vol}_c + \sum_{j \in J_i} \Phi_{ij} \left(\sum_c \alpha_{ic}\beta_{ijc}\mathrm{vol}_c \right) / |J|$$

is the local objective function at data center i. There is no need for coordination among data centers, and their decisions collectively attain the optimal solution to (4). The mapping problem (3) can also be distributed among mapping nodes; indeed [5] studies this problem. We omit the details here.

The distributed algorithm is summarized in Fig. 2. Mapping nodes collectively measure client request rates, and solve the global mapping problem **MAPPING** using routing decisions and path latencies from the routing system. Each data center locally measures path latencies to clients, and solves $\mathbf{ROUTING}_i$ using mapping decisions and traffic volumes from the mapping system. We assume that other optimization parameters—namely configuration of data centers, peering links, link capacities and mapping policies—vary slower than the decision reoptimization time, and are globally up to date.

V. Distributed Solution Evaluation

Experiment setup. CoralCDN [10] is a caching and content distribution platform running on Planetlab. Our request-level trace collected at 229 Planetlab sites running Coral consists of over 27 million requests and a terabyte of data, corresponding to March 31st, 2011. To emulate multi-homed service deployment, we cluster Planetlab sites in approximately the same metropolitan area and treat them as ISPs peering with the same service location—similar to the approach followed in [12]. Our setup has 12 service locations distributed in North America, Europe and Asia with 3-6 ISP connections each.

Requests arrive from about 95000 IP prefixes. For latencies to these prefixes, we use iPlane [14], a system that collects wide-area network statistics to Internet destinations from Planetlab vantage points. To correlate the traffic and latency data, we narrow down our client set to 24530 IP prefixes, which contribute 38% traffic (by requests) of the entire trace. The costs per unit bandwidth for ISP links are derived from a real-life cloud bandwidth pricing plan [20]. Due to space limitation, we refer the reader to an extended version for full details [17].

	Adaptive			Cold start		
# iterations	3	4	5	4	5	6
# instances	22	1	1	13	10	1
accuracy %	0.39	0.14	0.27	0.74	0.40	0.18

TABLE I. Solution Optimality and Convergence.

Benefits of joint optimization. We evaluate the benefits of full information visibility for **cost** and **perf** in isolation. Conceptually, this corresponds to setting $K = 0$ and $K = \infty$ respectively in the **GLOBAL** optimization. For **perf**, we compare against a baseline mapping that optimizes for average latency assuming *per-client least latency path* is used to reach each client from each data center, and a routing that optimizes **perf**. From hourly optimizations on the CDN trace, we find that **GLOBAL** achieves $\approx$ 10ms smaller **perf** than this baseline on average. For **cost**, we use a baseline mapping that optimizes for average cost assuming *average per-link cost* is incurred at each data center, and routing that optimizes **cost**. **GLOBAL** achieves between 3-55% lower cost across the trace.

Optimality to within 1% in 3-6 iterations. Over 24 hourly problem instances through the trace, we record the *number of iterations* for convergence of mapping-routing alternate projections, and the difference (in %) of the converged objective from the optimal value of $\mathbf{GLOBAL}_\Phi$, *i.e.*, *accuracy*. As a convergence criterion, we test if objective values from successive iterations are within a fixed percentage (say 1%) of each other. We examine two sets of initial conditions, namely *cold start*, *i.e.*, mapping is initialized with uniform round robin splitting, and *adaptive*, *i.e.*, mapping starts from the converged setting of the previous hour. The results are summarized in Table I. We find that (i) convergence occurs in 3-6 iterations, (ii) *adaptive* mapping generally converges faster than *cold start*, and (iii) the converged objectives are indeed within the fixed tolerance (1%) of the $\mathbf{GLOBAL}_\Phi$ optimum.

Optimization runtime under 1 minute on a modern machine. A single iteration of the alterate projection algorithm with 12 locations, 49 links and 24530 clients takes about 30 seconds on a desktop machine with a 2.40GHz two-core CPU.

Impact of penalty function Φ on latency. We evaluate how the value of **perf** is increased when the distributed algorithm optimizes a latency objective with link utilization penalties ($\mathbf{obj}_g$, eqn. (2)), instead of optimizing **perf** directly. Note that this relaxation in general causes an increase in **perf**. Over hourly instances through the trace, we find a uniform increase of $\approx$ 10ms **perf** over optimizing **perf** directly using 100% available link capacities. The gaps are 8ms and 5ms respectively when using only 95% and 90% of link capacities.

Picking a cost-performance tradeoff. The **GLOBAL** optimization trades off performance for cost through a parameter K. However, it can be challenging to know what a good tradeoff looks like. To understand this better, we visualize a pareto-optimal tradeoff curve between **cost** and **perf** for the the day in Fig. 6, as K varies. We see that there is a rich tradeoff space between cost and latency, with **perf** ranging between 77-145ms (88% increase possible from optimal **perf**) and **cost** ranging between 0.15-0.23 $/GB (53% increase possible). A service provider could manually pick a sweet spot from this curve—for example, the value of K corresponding to the "knee" at **perf** 98ms and **cost** $0.175/GB.

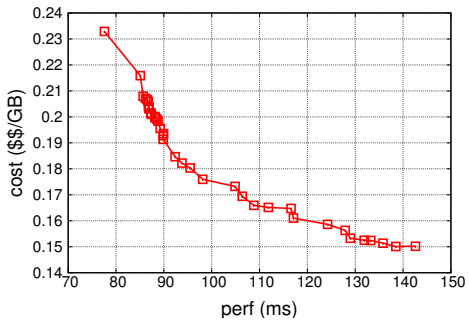


Fig. 6. Pareto-optimal tradeoff curve between latency (**perf**) and cost ($/GB).

VI. Related work

There is a rich literature exploring the benefits of visibility between networks, applications and service infrastructure.

Provider networks and selfish applications. P4P [21] and IETF ALTO [22] advocate that applications should use hints from ISPs to improve both their performance and ISP control over their traffic. OSP traffic engineering differs from this setting since the OSP controls the user-replica mapping directly.

ISP and CDN collaboration. There have been many recent proposals suggesting collaboration between ISPs and caching infrastructure to improve service performance [23]–[25]. In contrast, OSP traffic engineering considers paths *across* ISPs connected from distributed service locations. This changes the problem because flexible *content placement* inside a single ISP can provide significant performance gains. However, replication across ISPs can be expensive for OSPs hosting dynamic content [7], [26]. Our optimization model more closely resembles earlier proposals to mitigate bad interaction between replica selection and ISP traffic engineering *given* the set of replicas [16], [19], [27]. However, these works consider routing within a single administrative domain, and do not incorporate flexible mapping policy constraints, *e.g.,* (1c).

OSP joint traffic engineering. Unlike Entact [2], our algorithm achieves joint system gains by coordinating *modular* request-mapping and response-routing systems. Our goals are similar to those of PECAN [7] but our model also considers aspects which intimately couple mapping and routing decisions (*e.g.,* link capacities, bandwidth costs). Xu *et al.* [9] distribute the joint optimization for scale; however their separation of variables does not decouple the mapping and routing systems.

VII. Conclusion

We address the problem of jointly optimizing request-mapping and response-routing for online services, while retaining the functional separation between them. We highlight the challenges in achieving the gains of a joint system in a distributed setting, and mitigate these challenges through a distributed algorithm that provably converges to a globally optimum point. Hence, OSPs can attain the benefits of a joint system—namely better performance at lower cost—while only slightly refining the behavior of existing infrastructure.

Acknowledgment. This work was funded by NSF grant 1162112, DARPA grant FA8750-11-C-0254, and a gift from HP. We thank L. Vanbever, B. Balasubramanian, M. Wang and W. Lloyd for feedback and H. Madhyastha for iPlane traces.

References

[1] J. Hamilton, "The cost of latency," http://perspectives.mvdirona.com/2009/10/31/TheCostOfLatency.aspx.

[2] Z. Zhang, M. Zhang, A. Greenberg, Y. C. Hu, R. Mahajan, and B. Christian, "Optimizing cost and performance in online service provider networks," in *Proc. NSDI*, 2010.

[3] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," in *Proc. ACM SIGCOMM*, 2009.

[4] R. Krishnan, H. V. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao, "Moving beyond end-to-end path information to optimize CDN performance," in *Proc. IMC*, 2009.

[5] P. Wendell, J. W. Jiang, M. J. Freedman, and J. Rexford, "DONAR: Decentralized server selection for cloud services," in *Proc. ACM SIGCOMM*, 2010.

[6] D. K. Goldenberg, L. Qiu, H. Xie, Y. R. Yang, and Y. Zhang, "Optimizing cost and performance for multihoming," in *Proc. ACM SIGCOMM*, 2004.

[7] V. Valancius, B. Ravi, N. Feamster, and A. C. Snoeren, "Quantifying the benefits of joint content and network routing," in *Proc. ACM SIGMETRICS*, 2013.

[8] Amazon Route 53, http://aws.amazon.com/route53/.

[9] H. Xu and B. Li, "Joint request mapping and response routing for geo-distributed cloud services," in *Proc. IEEE INFOCOM*, 2013.

[10] M. J. Freedman, E. Freudenthal, and D. Mazières, "Democratizing content publication with Coral," in *Proc. NSDI*, 2004.

[11] Y. Zhu, B. Helsley, J. Rexford, A. Siganporia, and S. Srinivasan, "LatLong: Diagnosing wide-area latency changes for CDNs," *IEEE Transactions on Network and Service Management*, vol. 9, 2012.

[12] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman, "A measurement-based analysis of multihoming," in *Proc. ACM SIGCOMM*, 2003.

[13] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," in *Proc. IEEE INFOCOM*, 2000.

[14] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane: An information plane for distributed services," in *Proc. OSDI*, Nov. 2006.

[15] M. Szymaniak, D. Presotto, G. Pierre, and M. van Steen, "Practical large-scale latency estimation," *Computer Networks*, vol. 52, pp. 1343–1364, May 2008.

[16] J. W. Jiang, R. Zhang-Shen, J. Rexford, and M. Chiang, "Cooperative content distribution and traffic engineering in an ISP network," in *Proc. ACM SIGMETRICS*, 2009.

[17] S. Narayana, W. Jiang, J. Rexford, and M. Chiang, [Online] http://www.cs.princeton.edu/~narayana/jointopt-dcc13-extended.pdf.

[18] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Communication Magazine*, 2002.

[19] D. DiPalantino and R. Johari, "Traffic engineering vs. content distribution: A game theoretic perspective," in *Proc. IEEE INFOCOM*, 2009.

[20] Amazon EC2 Pricing, http://aws.amazon.com/ec2/pricing/.

[21] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4P: Provider portal for applications," in *SIGCOMM*, 2008.

[22] IETF ALTO, https://datatracker.ietf.org/doc/draft-ietf-alto-protocol/.

[23] I. Poese, B. Frank, G. Smaragdakis, S. Uhlig, A. Feldmann, and B. Maggs, "Enabling content-aware traffic engineering," *ACM SIGCOMM Computer Communication Review*, 2012.

[24] A. Sharma, A. Venkataramani, and R. K. Sitaraman, "Distributing content simplifies ISP traffic engineering," in *ACM SIGMETRICS*, 2013.

[25] B. Frank, I. Poese, Y. Lin, G. Smaragdakis, A. Feldmann, B. Maggs, J. Rake, S. Uhlig, and R. Weber, "Pushing CDN-ISP Collaboration to the Limit," *SIGCOMM CCR*, July 2013.

[26] S. Agarwal, J. Dunagan, N. Jain, S. Saroiu, A. Wolman, and H. Bhogan, "Volley: automated data placement for geo-distributed cloud services," in *Proc. NSDI*, 2010.

[27] I. Poese, B. Frank, B. Ager, G. Smaragdakis, and A. Feldmann, "Improving content delivery using provider-aided distance information," in *IMC*, 2010.