

Motivating Participation in Internet Routing Overlays

Dave Levin, Randolph Baden, Cristian Lumezanu, Neil Spring, Bobby Bhattacharjee

Department of Computer Science
University of Maryland, College Park, MD 20742
{dml,randofu,lume,nspring,bobby}@cs.umd.edu

ABSTRACT

PeerWise is an Internet routing overlay that reduces end-to-end latencies by allowing peers to forward through a relay instead of connecting directly to their destinations. Fundamental to PeerWise is the notion of peering agreements between two peers, wherein they agree to forward for one another. In this paper, we consider the problem of motivating users to establish and maintain peerings in a completely decentralized, scalable manner. We show that routing overlays present unique challenges and goals. For instance, since participants can always “fall back” on standard Internet routing, we must encourage users to stay in the system and maintain long-lived peering agreements. To address these challenges, we propose two mechanisms: First, we use Service Level Agreements (SLAs) to expressively negotiate peers’ demands and the recourses they will take when SLAs are violated. Second, we propose a mechanism to address SLA violations that differs from the standard notion of punishment via service degradation. Our simulation results demonstrate that our mechanism causes peers to avoid SLA violators in favor of long-lived peerings. Lastly, we discuss potential, emergent behaviors in a selfish routing overlay.

Categories and Subject Descriptors

C.2.1 [Computer-Communication Networks]: Network Architecture and Design; H.5.3 [Information Interfaces and Presentation]: Group and Organization Interfaces—*Collaborative computing*; J.4 [Computer Applications]: Social and Behavioral Sciences—*Economics*

General Terms

Algorithms, Design, Economics, Performance

Keywords

Incentives, Internet routing overlays, Service-level agreements

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

NetEcon’08, August 22, 2008, Seattle, Washington, USA.
Copyright 2008 ACM 978-1-60558-179-8/08/08 ...\$5.00.

1. INTRODUCTION

Routing overlays [1, 11, 7, 9] allow users to forward traffic through other users to influence the path their packets take through the Internet. With such influence, users can achieve greater resilience to network outages and decrease end-to-end latencies.

As more users participate in a routing overlay, more paths become available, and hence the utility of the overlay increases. We therefore propose as a goal of routing overlays to motivate participation: *encourage users to join the routing overlay and actively forward for others for as long as possible*. Until recently, routing overlays have been relegated to small deployments controlled by a single administrator; RON [1] for instance creates a fully connected mesh in which all nodes measure latencies to all other nodes, limiting a feasible deployment to only a few dozen nodes. PeerWise [9] makes scalability possible in these systems by using network coordinates [5] to reduce the number of peers users must contact to find shorter-latency paths to their destinations.

The feasibility of a large-scale, decentralized routing overlay raises problems of ensuring cooperative participation in the presence of selfish users. We expect users to selfishly attempt to maximize the *benefit*—latency reduction and bandwidth—they receive from the system while minimizing their *cost*—the amount of traffic they forward for others. Selfish users complicate participation because they compete with one another for latency-reducing peers, and will not forward for one another if they do not receive something in return. As a system scales to a large number of users, maintaining global state becomes infeasible. Corbo et al. use local decisions to form economically-driven peerings to model AS topology [4]; we apply a similar philosophy in routing overlays to model user interactions and provide economic incentives for users to join.

The goals of this work are to motivate users to *establish* and *maintain* mutually beneficial peerings, and to do so in a completely decentralized, scalable manner. Our first contribution in this work is in demonstrating that *routing overlays present unique challenges in providing incentives for selfish users to participate* (§2). For instance, as compared to a system like BitTorrent [3], time plays a significant role in user demands; BitTorrent peers want as much as possible *now*, while users of a routing overlay may generate traffic at varying times throughout the day, and with varying demand.

In attempting to solve the *selfish routing overlay problem*, we focus on building incentives in PeerWise. Our solution to motivating participation is based on PeerWise’s notion of *peering agreements*. A peering agreement in PeerWise is made between two participants, p and q , wherein p agrees to act as a latency-reducing relay for q , and vice versa.

A selfish PeerWise user is faced with two interdependent decisions: (1) choosing with whom to form peering agreements, and (2) choosing how much to allocate to the peers it chooses. We pro-

pose two mechanisms to address these. The first mechanism (§4) borrows from the notion of Service Level Agreements (SLAs) that autonomous systems in the Internet use to establish business agreements with one another. SLAs in PeerWise allow peers to expressively negotiate their demands and the recourses they take when the SLAs are violated.

SLA violations require recourse. The second mechanism we present (§5) addresses the problem of avoiding negotiations with peers who continually violate their SLAs. An intuitive solution to this is for peers to punish one another by offering reduced quality of service, but this runs the risk of incurring unnecessary performance degradation or, if engineered poorly, system-wide collapse. Our mechanism takes a subtly different approach than punishment. We propose that peers instead use SLA violations as a negative reflection on the perceived benefit that violators offer. We present simulation results that show that our mechanism limits how often victims return to an SLA violator, and achieves long-lived peerings with peers that satisfy SLAs. We conclude with a discussion of potential emergent behaviors in a selfish routing overlay, and open problems (§6).

2. THE SELFISH ROUTING OVERLAY PROBLEM

The problem of providing incentives in routing overlays represents a new point in the design space of incentives for selfish participants. Broadly, we aim to promote participation and resource sharing among selfish peers. In this section, we formulate the problem of providing incentives in routing overlays in the context of how it differs from well-studied problems.

2.1 Participants may opt out

Latency-reducing routing overlays are an optimization on standard Internet routing. Participants in routing overlays may always leave the system and “fall back” on the higher-latency paths that BGP returns. Thus, when a participant in a routing overlay perceives their costs from remaining in the system to exceed their benefits, they may *opt out* of the system. In this sense, fairness comes “for free”: routing overlay participants can always achieve correctness without their costs exceeding their benefit. This is unlike most P2P systems such as BitTorrent [3], where for many files the only way to download the file is via BitTorrent.

We view the fundamental goal of providing incentives in routing overlays to be to *motivate peers to stay in the system as long as possible*. This equates to allowing peers to easily find benefit from the system (if it exists for them) and to maintain that benefit over long periods of time.

2.2 Not all peers benefit from one another

It is not the case that a given peer can benefit from every other peer. Though the triangle inequality does not always hold in the Internet, it often does [5, 14, 15]. Conversely, in many peer-to-peer systems, all peers stand to benefit from all others. In BitTorrent, for example, a peer can potentially benefit from every other peer, as long as they do not have all of the same pieces of the file. Peers in a routing overlay, however, may have a small (or perhaps even empty) set of other peers who can offer them lower latency paths to a given destination. A goal is thus to *match up the peers in a way that benefits as many as possible*.

2.3 Users have varying demand

Users’ traffic and latency demands in the Internet vary across the applications they use, the destinations they contact, the time of the

day, and so on. For some applications, like ssh, a small amount of bandwidth with a large reduction in latency is sufficient, while in others, like VoIP, a reduction in latency is allowed only as long as there is a moderate amount of bandwidth. This is distinct from systems like BitTorrent, wherein everyone’s *demand* is the same: download the file as quickly as possible. Another goal of a routing overlay is thus to allow peers to communicate their demands with one another. If two peers are willing to relay for one another, then they should ideally be able to negotiate a mutually advantageous agreement. Put simply: *allow willing peers to peer*. We propose to apply the notion of Service Level Agreements (SLAs) to routing overlays to achieve this expressiveness (§4).

2.4 Long-lived agreements are preferred

Users vary not only the *amount* of resources they demand, but also the *time* at which they demand it. Often in routing overlays, peers may be expected to generate traffic at different times. Conversely, BitTorrent peers benefit from one another *simultaneously*. An important goal in selfish routing overlays is therefore to *motivate peers to maintain long-lived agreements with one another to forward, and to not require simultaneous interest*. Long-lived agreements are natural in routing overlays, because if p benefits from having r as a relay at one point in time, p is likely to continue benefiting from r in the future [9].

This differs from applications where there is an end-game. In BitTorrent, for example, there is a finite-sized file, so peers will not benefit from one another indefinitely. Ad hoc wireless networks might also be considered to have an end-game in that the comprising nodes are generally battery-constrained, or are mobile and therefore unlikely to interact with others for long periods of time.

We propose a mechanism to ensure long-lived peerings. Interestingly, our mechanism does not require any degraded service, and is resilient to varying network conditions (§5).

2.5 No modifications to destinations

Routing overlays are intended to be used to contact any destinations in the Internet. Requiring modifications to all destinations in the Internet would render deployment infeasible. We do not require any modifications to the users’ destinations.

This affects how selfish peers in a routing overlay interact with one another. Suppose peer p is relaying his traffic through peer r to reach destination p_d . Tunneling packets through r entails r rewriting the source address of the packet with his own IP address. From p_d ’s perspective, r ’s host is connecting to p_d , not p ’s. If r were to stop relaying for p while p had an open connection to p_d , then p would have no means of recovering this connection. Modifications to the destination could alleviate this, but again, at the cost of ease of deployment. This further motivates the need for long-lived peerings; by providing incentive for r to maintain its peering agreement with p , p can gain (more) assurance that its connections through r will remain.

2.6 Peerings are not roommates

Establishing peering agreements in PeerWise is similar to the classic stable roommates problem [8]. The stable roommates problem takes as input a set of agents, and outputs a *matching*: a set of pairs of agents (“roommates”). A matching is *stable* if there are no two agents that would both prefer to be matched with one another over the agents with whom they are respectively matched. A matching M is *maximal* if there is no other matching M' such that $|M'| > |M|$.

Stable and maximal matching problems appear to be a natural fit for PeerWise. Stability relates to long-lived peering agree-

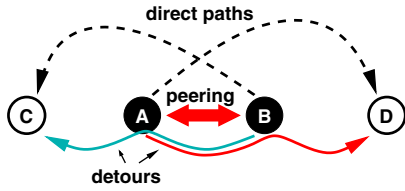


Figure 1: Obtaining faster paths with PeerWise: A discovers a detour to C through B; B also finds that it can reach D faster if it traverses A; A and B create a mutually advantageous peering which they both use to get more quickly to their destinations.

ments, and maximal matching equates to maximizing participation in PeerWise. However, the problem of choosing relays in PeerWise is vastly more complex than the stable roommates problem. Instead of being matched with one other peer, participants in PeerWise make many peering agreements; preferences in PeerWise may therefore be exponential in size. Varying network conditions and user demands add further complexity. It is therefore unlikely that a generalization of the stable roommates problem to PeerWise-like settings will allow a polynomial time algorithm. A more rigorous understanding of the connection between the two problems is an area of future work, and may help to reveal the theoretical limits of the mechanisms we present in this paper.

3. SELFISHNESS IN PEERWISE

We focus on providing incentives in the PeerWise [9] routing overlay. In this section, we review PeerWise, and define the utility of a PeerWise participant.

3.1 PeerWise Overview

PeerWise [9] is a latency-reducing routing overlay that provides scalability and fairness. The key idea of PeerWise is that two nodes can cooperate to obtain faster end-to-end paths without either being compelled to offer more service than they receive. PeerWise comprises two mechanisms: (1) it uses network coordinate systems, such as Vivaldi [5], to aid in discovering triangle inequality violations in the Internet; and (2) nodes negotiate and establish *peering agreements* to each other based strictly on mutual advantage.

Figure 1 shows a two-node PeerWise overlay. Node A discovers a faster path to D via B. However, B will not help A unless A provides a detour in exchange. Since there is a shorter path from B to C going through A, A and B can help each other communicate faster with their intended destinations. Therefore they establish a pairwise peering.

3.2 Self-interest in PeerWise

Selfish peers will strategically attempt to form peering agreements that maximize their individual utility, regardless of how it might impact the rest of the system. Here, we formalize the goals and utilities of selfish peers. We assume that peers only seek out mutually advantageous latency reduction, as opposed to one party paying the other to relay. We leave the use of transferable utility (money) as an alternative form of payment to future work.

3.2.1 A PeerWise peer’s utility

A peer’s utility is defined by the benefits and costs that it perceives. The *benefit* that a PeerWise peer p obtains from using r as a relay to reach destination d is an increasing function of its latency reduction:

$$L_p(r) \stackrel{\text{def}}{=} 1 - \frac{\ell(p \rightarrow r \rightarrow d)}{\ell(p \rightarrow d)}$$

where $\ell(x)$ denotes the latency for path x . A peer’s benefit is also a function of the amount of traffic it is allowed to forward. For example, a 99% latency reduction for one packet is likely not preferable to a 20% reduction for all of a peer’s traffic. For ease of exposition, we assume that p has a minimum amount of required bandwidth, and if r does not provide this, then p receives no benefit from r .

The *cost* that peer p incurs from relaying for r is an increasing function in $b_p(r)$, the amount of bandwidth p forwards for r , and is normalized by the amount of bandwidth p is willing to provide to PeerWise, B_p :

$$C_p(r) \stackrel{\text{def}}{=} \frac{b_p(r)}{B_p}$$

Such cost can come in many forms: the impact to their own applications’ performance, actual money-per-byte costs, the need to leave their computer turned on over night, and so on.

We propose the following general form utility function for peer p :

$$u_p(r) \stackrel{\text{def}}{=} \alpha_p L_p(r) - \beta_p C_p(r) \quad (1)$$

Constants $\alpha_p \geq 0$ and β_p can be chosen to suit a wide range of users:

- $\alpha_p = 0, \beta_p < 0$: p is altruistic.
- $\alpha_p > \beta_p > 0$: p is willing to pay more for his latency savings than he receives.
- $\beta_p > \alpha_p > 0$: p expects to get more than he gives back to the system.
- $\alpha_p > 0, \beta_p = 0$: p does not incur costs on his uplink.

As discussed in §2.1, a peer will not remain in the system if its total costs outweigh its total benefits. We now see that p will leave the system if its sum utility is less than zero. Note that if p is altruistic then $\alpha_p = 0$ and $\beta_p < 0$, hence u_p is never negative.

3.2.2 Selfishly selecting peering agreements

Each PeerWise peer p must decide when to commit to and when to dissolve a peering agreement. *This is the fundamental problem a selfish peer must solve in PeerWise.*

More formally, a selfish peer p ’s goal is to choose a set \mathcal{P} of peers with whom to make peering agreements. Suppose that p that is willing to contribute a total amount of bandwidth $B_p \cdot T$ to other PeerWise peers over any period of time T . To limit the *burstiness* of p ’s contribution, p may also wish to ensure that it never has to provide more than S_p bandwidth at any point in time. This protects p from having to provide all of its $B_p \cdot T$ bandwidth in a very small period of time, which could adversely affect p ’s performance. We formalize this with the following natural optimization problem:

$$\text{maximize } \sum_{q \in \mathcal{P}} u_p(q) \quad (2)$$

$$\text{s.t. } \sum_{q \in \mathcal{P}} C_p(q, t) \leq S_p, \quad \forall t \quad (3)$$

$$\sum_{q \in \mathcal{P}} \sum_{t=t_0}^T C_p(q, t) \leq B_p \cdot T, \quad \forall t_0, T \quad (4)$$

This formulation makes clear an important difference between selfish behavior in PeerWise and selfish behavior in a system like BitTorrent: *time*. Peers’ demands in BitTorrent are constant: send as much as possible *now*. However, a peer’s demands in PeerWise depend on when that user wishes to access its destinations, and could be nil when the user is asleep, for instance. We investigate this further in the context of over-committing one’s resources

(§6.1). In §4 and §5, we present mechanisms to assist PeerWise peers in obtaining a beneficial a set of peering agreements, while motivating them to uphold their responsibilities in their agreements.

4. SERVICE LEVEL AGREEMENTS

In PeerWise, nodes negotiate and establish peering agreements to each other based strictly on mutual advantage. In a previous paper [9] we showed how potentially good peers are found—by exploiting the inability of network coordinates to work with triangle inequality violations—and how the peering agreements are negotiated—nodes that provide each other latency reduction to at least one destination form a peering. Next we focus on mechanisms that ensure the validity and stability of such agreements over long periods of time.

A service level agreement (SLA) is a formal contract between two peers that establishes all aspects of the service that each provides to the other. Nodes in routing overlays are inherently selfish and SLAs are an efficient method to curtail the effects of the selfishness. SLAs ensure that each node receives the expected level of quality-of-service even when the traffic demand in the network varies and the mutual advantage offered by a peering is time dependent. We describe the design of the PeerWise SLAs next.

4.1 SLA design space

We identify three performance metrics to be used as the basis for an SLA between two PeerWise users:

- **latency reduction:** the minimum latency reduction that a peer offers to the other to a specific destination
- **bandwidth:** the average bandwidth at which one peer forwards packets for the other to a specific destination
- **burstiness:** the maximum bandwidth at which one peer forwards packets for the other to a specific destination

Each SLA is associated with a re-evaluation timeout which triggers a verification of the SLA against the traffic exchange since the previous timeout. We assume that each PeerWise user has the means to monitor the traffic over all its peerings. Because IP is best effort and cannot provide measurable service, we do not require the bounds on the performance metrics defined in the SLA to be strictly enforced; it is enough if most of the packets sent over the peering satisfy the bounds specified by the SLA. Furthermore, we enforce frequent re-evaluations (on the order of minutes or hours rather than days or weeks). In doing so, we seek to protect the users against long periods when although the SLA is not violated, the peering is unusable.

Next we present a simple example of SLA between two PeerWise users. In Figure 1, nodes A and B discover mutual advantage and decide to form a peering. The SLA of the peering may state that A and B must offer each other latency reductions of at least 10ms for 95% of the packets that each send to C and D. The re-evaluation timeout is set 60 seconds. Furthermore, the bandwidth that each peer uses to forward each other’s packets must not exceed 20kB/s averaged over the 60 seconds, with its maximum value always below 100kB/s.

Our SLA design draws from the agreements between autonomous systems in the Internet. Similarly to AS SLAs, PeerWise SLAs are intended to be maintained over long periods of time. In this way, long term reputation can motivate cooperation. However, AS SLAs are defined over much larger time scales. We require SLAs to be verified more often because traffic fluctuations may have bigger effects on a single link that connects two users than on a collection of links that interconnects two autonomous systems.

4.2 Dynamic SLAs

Because nodes generally do not know *a priori* their traffic demands and because the service performance metrics are chosen based on at most a few measurements, we do not require fixed SLAs between two PeerWise nodes. Instead, two nodes start with a temporary SLA, which specifies latency reductions as advertised during the negotiation and bandwidth metrics according to each node’s expected demand. This is different from AS level SLAs, which are based on previous study of the network utilization and on performance metrics averaged over long periods of time [10].

The bandwidth and burstiness parameters of the SLA can be automatically changed after every re-evaluation to align with the real traffic exchanged between the peers. We allow both positive and negative parameter scaling. If two peers respect the terms of the SLA and if they have enough spare bandwidth, they may offer to relay each other’s traffic at faster and faster rates. On the other hand, if a peer does not offer as much latency reduction as promised, then the available bandwidth through its peer will be reduced. In the next section we present mechanisms to encourage long-lived peerings even when users are selfish.

5. MECHANISMS FOR LONG-LIVED PEERINGS

PeerWise participants’ choice of relays is driven by their selfish interest in obtaining their most preferred relays. Each peer p has a partially ordered preference, \succ_p , of relays. If, for two peers a and b , $a \succ_p b$, then p will prefer to use a as a relay regardless of the impact on b .

Given the anarchistic nature of this process, it is not surprising that it can yield poor outcomes. For example, consider three peers, a , b , and c , where $b \succ_a c$, $c \succ_b a$, and $a \succ_c b$. These cyclic preferences lead to oscillating peering agreements: if a obtains its preferred relay b , then that leaves c available. Since b prefers c , b will selfishly break its peering agreement with a in favor for c . This in turn leaves a available, leading c to break its peering agreement with b , and the process repeats indefinitely. These oscillating peering agreements resemble persistent oscillations which could occur in BGP [12]. Next we present a simple mechanism that limits the effect of selfishness on the stability of peerings.

5.1 Long-term cooperation with confidence

A fundamental difference between route changes in BGP and PeerWise is that a route change in PeerWise will generally result in a lost connection (§2.5). As such, PeerWise participants desire long-lived peerings, and should prefer relays who respect their SLAs for long periods of time. Though a peer i can measure the reduction in latency that a neighbor j provides, i cannot know if or when j will violate their SLA. Instead, i maintains some measure of *confidence* that j will relay i ’s packets. i ’s confidence in j can change over time, based at least in part on the history of interactions between i and j .

We propose that, in determining neighbor preferences, a peer *explicitly* incorporates its confidence that a given neighbor will maintain an SLA. Specifically, we propose that each peer i maintain a *confidence value*, c_{ij} for each peer j with whom it has interacted. A peer’s confidence in another increases or decreases based on the interactions those two peers have had. For instance, when peer j violates or prematurely dissolves a peering agreement with i , i lowers his confidence of j . Though c_{ij} could also incorporate *others’* interactions with j , we focus in this paper only on confidence based on one’s own interactions.

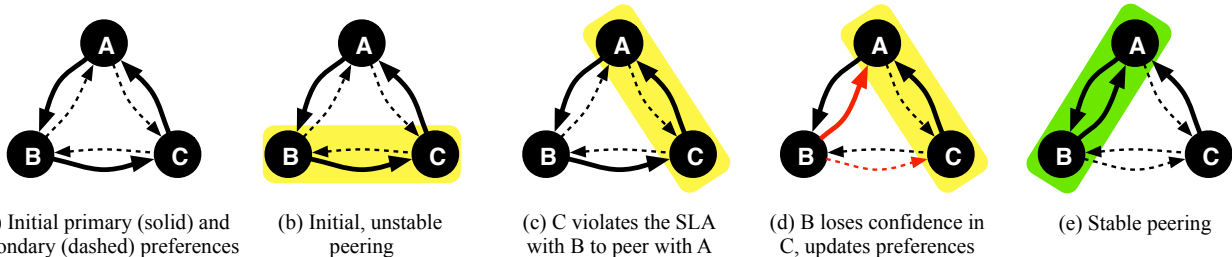


Figure 2: Cyclic preferences lead to unstable peerings, but participants prefer long-lived peerings. Algorithm 1 addresses this by incorporating into a peer B’s preference the confidence B has in its neighbors to maintain a long-lived peering.

Algorithm 1 Handling SLA violations.

Periodically run at peer i :

- when i and j have a peering agreement
 - if j respects the SLA, $c_{ij} \leftarrow c_{ij} + c_{step}$
 - if j violates the SLA or drops the peering, $c_{ij} \leftarrow c_{ij}/2$
 - when i and j do not have an agreement, but they previously had one
 - $c_{ij} \leftarrow c_{ij} + c_{step}/(s_{ij} + 1)$
-

Confidence values more accurately capture a peer’s expected utility from a neighbor because they incorporate not just the instantaneous utility, but the history of interactions between the peers. Peers’ preferences over one another can then combine the potential benefit of a peering—based on latency reduction, bandwidth, and burstiness described in §4—and the expectation that the peer will actually provide it. The power of incorporating history is demonstrated in Figure 2, wherein B’s loss of confidence in C eliminates the initial, oscillating peering agreements.

To experimentally study the use of confidence values, we now propose a specific algorithm, Algorithm 1, to capture peers’ confidence in one another. Peer i runs the algorithm at every predetermined period of time T . c_{step} represents the default value for increasing confidence. s_{ij} represents the number of times j has dropped a peering with i .

In Algorithm 1, the confidence value for a node increases slowly (additively) as the SLA is respected, and decreases quickly (multiplicatively) whenever the node violates the SLA. In this way, it will be more difficult for j to re-establish the agreement with i . This mechanism acts as a disincentive for nodes to constantly drop and add peerings on a small time scale.

5.2 Experimental results

To study the stability of PeerWise peerings we evaluate our confidence mechanism with a simulator. The simulations use a fixed topology of 16 PlanetLab nodes—the PeerWise nodes—and 16 popular web servers—the destinations. The latencies between PlanetLab nodes and web servers were measured in January 2008.

Each PeerWise node attempts to connect to each destination. SLAs are established between nodes that offer each other at least 10ms or 10% latency reduction. In our experiments, bandwidth is not a constraint. A peer violates an SLA associated with an existing peering whenever a new peering that offers higher benefit to the same destination is discovered. We assume that the re-evaluation period and the performance metrics of the SLAs never change. To assign confidence values for each node, we use the algorithm de-

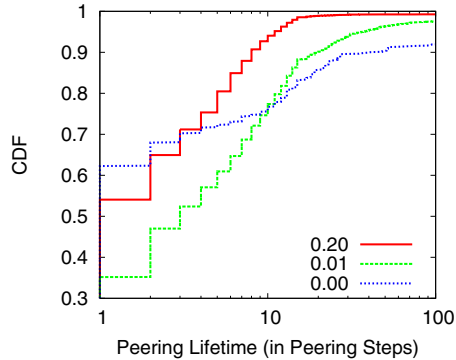


Figure 3: Peering lifetimes for different values of c_{step} .

scribed in §5.1. We restrict the confidence values to the interval $[0,1]$ and we use three different values for c_{step} : 0, 0.01 and 0.2. After each re-evaluation period (peering step), we compute the total number of existing peerings, as well as the number of peerings that have been destroyed since the last re-evaluation. We present the results in Figures 3 and 4.

When c_{step} is 0, the confidence value of a peer that destroys a peering can never increase. Destroyed peerings are not recreated and the number of existing peerings after each re-evaluation eventually converges to a small value (around 50 in Figure 4). However, the peerings that *do* exist are long-lived, as shown in Figure 3. As c_{step} increases, so does the chance that a destroyed peering will be re-established. For $c_{step} = 0.2$, almost 150 exist at a given moment. Though there are many more peerings with greater values of c_{step} , the longevity of these peerings is diminished.

Thus, confidence values quantify the trade-off between how long-lived a peering is—longer with a smaller c_{step} —and how many peerings there are—more with a larger c_{step} .

6. EMERGENT BEHAVIORS

In this section, we discuss two behaviors that may emerge in a system of selfish PeerWise participants.

6.1 Overselling one’s resources

Peers may be inclined to promise more bandwidth across all of the peering agreements than they choose to actually provide over any period of time. This is common with today’s ISPs, which advertise more bandwidth to home users than the ISP has provisioned. ISPs do this under the expectation that few enough users will wish to consume the advertised bandwidth at the same time. In other words, ISPs provision for average demands, and offer no assurance to customers when demand far exceeds the average.

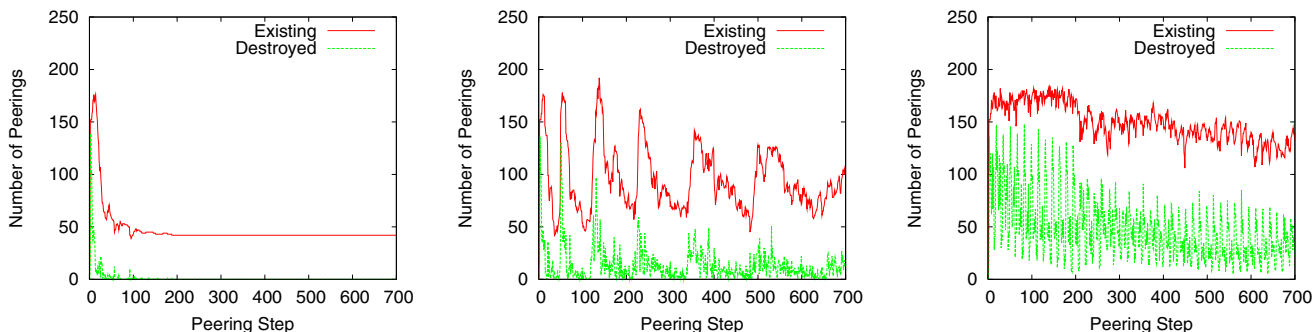


Figure 4: Example runs for different values of c_{step} : 0 (left), 0.01 (middle) and 0.2 (right).

In PeerWise, when a peer p makes a peering agreement with peer q , p commits to providing some amount $b_p(q)$ of bandwidth to q . Recall from Eq. (4) that p provisions B_p bandwidth for participation; p wishes to never provide more than $T \cdot B_p$ bandwidth over any period of time, T . This does not necessarily mean that, across all of p 's peering agreements, $\sum_{q \in \mathcal{P}} b_p(q) \leq B_p$. Whenever this inequality does not hold, we say that p has *oversold* his resources.

The benefit of overselling is that it allows a peer to create more peering agreements, and therefore reach more destinations with lower latency. The risk of overselling is that, when more peers attempt to claim their promised bandwidth than p provisioned for, then p will be forced to violate the constraints in Eqs. (3) and (4). Violating these constraints may result in p violating some of its peering agreements, and as losing some of its relays (§5).

6.2 Reselling peering agreements

Selfish participants in PeerWise may attempt to create multi-hop paths to increase their profits. One such opportunity arises because PeerWise peers are unlikely to have global information about all relays. In a cooperative setting, if peer p knew that peer r could act as a latency-reducing relay for q —that is, $\ell(q \rightarrow r \rightarrow d_q) < \ell(q \rightarrow d_q)$ —then p would inform q about r . In a selfish environment, p may trivially attempt to charge q for this information. An interesting scenario occurs when $\ell(q \rightarrow p \rightarrow r \rightarrow d_q) < \ell(q \rightarrow d_q) < \ell(q \rightarrow p \rightarrow d_q)$. In this case, p may attempt to *pretend* that he can offer q a one-relay, latency-reducing path, and secretly “re-relay” through r . Although q may not obtain as much benefit as if he were to know of and relay directly to r , q does obtain improved performance. This provides peer p an in-band incentive to assist peers in finding lower-latency paths.

Reselling need not be an act of subterfuge. Although most latency reduction is possible with one-relay paths, there are some source-destination pairs which obtain their optimal latency reduction with multi-hop paths [9]. Building multi-hop paths is a natural extension of peering agreements in PeerWise, as above.

An interesting area of future work is in studying the incentives properties of a path of these bilateral agreements. Would, for instance, a path built of bilateral peering agreements have comparable incentive properties to mechanisms that require more communication, such as VCG routing [13, 2, 6]?

7. CONCLUSION

We have proposed the selfish routing overlay problem, and demonstrated its difference from other, well-studied problems such as incentives in BitTorrent and the stable roommates problem. This problem is complex, in part due to the fact that participants’ utilities can vary greatly. The crux of the selfish routing overlay problem is

a local decision made by each peer: which peering agreements to maintain, and when, if ever, to dissolve the agreements it has.

Within the context of the PeerWise Internet routing overlay, we proposed two mechanisms to assist selfish routing overlay participants in choosing and negotiating their peering agreements. We believe these two mechanisms to be fundamental components toward solving the selfish routing overlay problem. An in-band mechanism as general as an SLA addresses the disparate goals of these peers. Reacting to the history of interaction between peers addresses persistent free-riders and unstable preferences.

The mechanisms presented in this paper are some initial steps toward addressing the tussles that exist between routing overlay peers. Many areas of future work remain. An important area of future work is in developing specific SLAs and studying how well they perform in the Internet, and what behaviors emerge.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. This work was supported in part by NSF Awards ITR-0426683, CNS-0643443 and CNS-0626629, and MIPS grant 3808.

8. REFERENCES

- [1] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *SOSP*, 2001.
- [2] E. Clarke. Multipart pricing of public goods. *Public choice*, 11, 1971.
- [3] B. Cohen. Incentives build robustness in BitTorrent. In *P2PEcon*, 2003.
- [4] J. Corbo, S. Jain, M. Mitzenmacher, and D. C. Parkes. An economically principled generative model of AS graph connectivity. In *NetEcon+IBC*, 2007.
- [5] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: a decentralized network coordinate system. In *SIGCOMM*, 2004.
- [6] T. Groves. Incentives in teams. *Econometrica*, 41, 1973.
- [7] K. P. Gummadi, *et al.* Improving the reliability of internet paths with one-hop source routing. In *OSDI*, 2004.
- [8] R. W. Irving. An efficient algorithm for the stable roommates problem. *Journal of Algorithms*, 6:577–595, 1985.
- [9] C. Lumezanu, D. Levin, and N. Spring. PeerWise discovery and negotiation of faster paths. In *HotNets*, 2007.
- [10] J. Martin and A. Nilsson. On service level agreements for ip network. In *IEEE Infocom*, 2002.
- [11] A. Nakao and L. Peterson. Scalable routing overlay networks. In *ACM SIGOPS Operating Systems Review*, 2006.
- [12] K. Varadhan, R. Govindan, and D. Estrin. Persistent route oscillations in inter-domain routing. *Computer Networks*, 32(1):1–16, 2000.
- [13] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16, 1961.
- [14] B. Wong, A. Slivkins, and E. G. Sire. Meridian: A lightweight network location service without virtual coordinates. In *SIGCOMM*, 2005.
- [15] B. Zhang, *et al.* Measurement-based analysis, modeling, and synthesis of the internet delay space. In *IMC*, 2006.