



James K. Smith
Executive Director-
Federal Regulatory

AT&T Services Inc. T: 202.457.3010
1120 20th Street, NW F: 202.457.3070
Suite 1000
Washington, DC 20036

March 12, 2008

VIA ELECTRONIC SUBMISSION

Ms. Marlene H. Dortch
Secretary
Federal Communications Commission
445 12th Street, S.W.
Washington, D.C. 20554

Re: Ex Parte Statement: Broadband Industry Practices
(WC Docket No. 07-52)

Dear Ms. Dortch:

On March 11, 2008, Hank Hultquist, Jack Zinman, and James Smith, on behalf of AT&T Services, Inc. (AT&T), met with Jeremy Miller, Jonathan Reel, Nicholas Alexander, William Kehoe, and Melissa Kirkel of the Wireline Competition Bureau to discuss basic positions AT&T has taken as set forth in its filings made in the above reference proceeding. The attached slides were used as a basis for discussion. In addition, the attached P4P document was provided to staff in the course of the meeting.

In accordance with Section 1.1206 of the Commission's rules, this letter is being filed electronically with you office for inclusion in the public record of this proceeding.

If you have any questions, please do not hesitate to contact me at (202) 457-3010.

Respectfully submitted,

A handwritten signature in black ink, appearing to read "James K. Smith".

Attachments

Cc: N. Alexander
W. Kehoe
M. Kirkel
J. Miller
J. Reel

WC Docket No.07-52
Free Press/Vuze Petitions

March 11, 2008



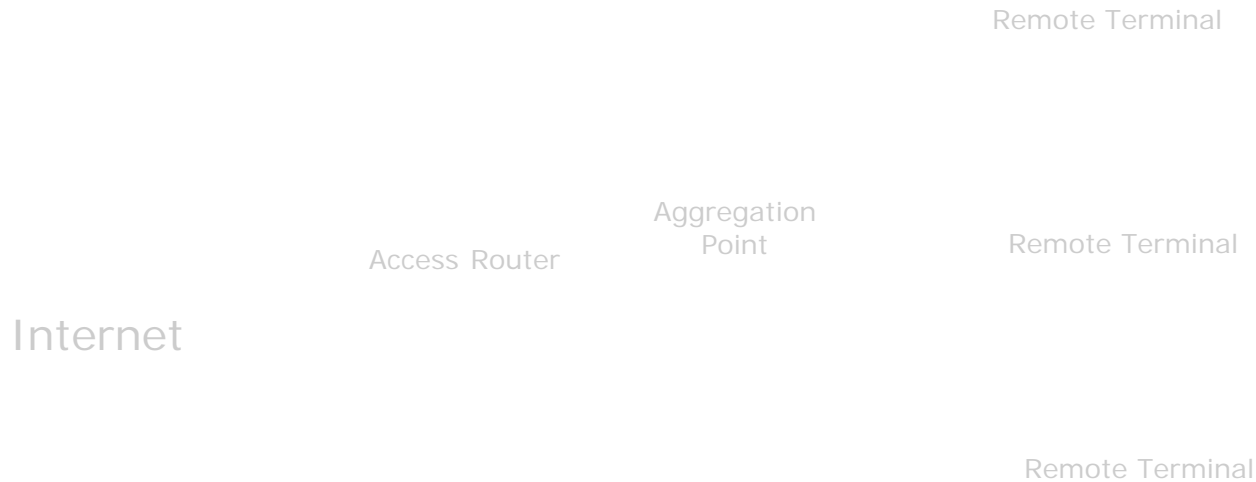
Summary

- Broadband networks are inherently shared networks.
- P2P is not necessarily an efficient technology in its present form.
- High bandwidth applications are driving growth and usage.
- Strict nondiscrimination only serves the interests of elitist users.



Wireline Broadband Networks Are Inherently Shared Networks That Must Be Managed

Shared DSL/FTTP



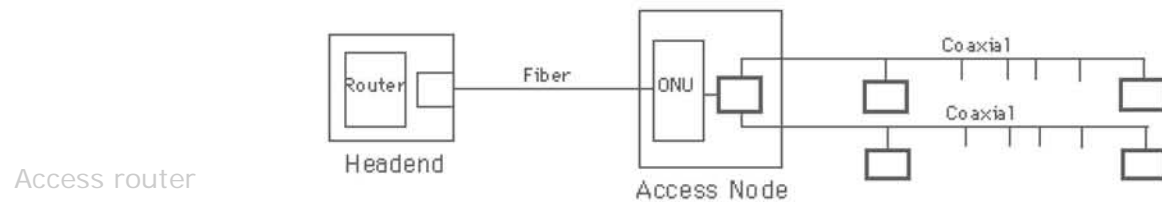
Shared transport facilities designed based on engineering assumptions

Dedicated



Cable Broadband Networks Are Inherently Shared Networks That Must Be Managed

Internet



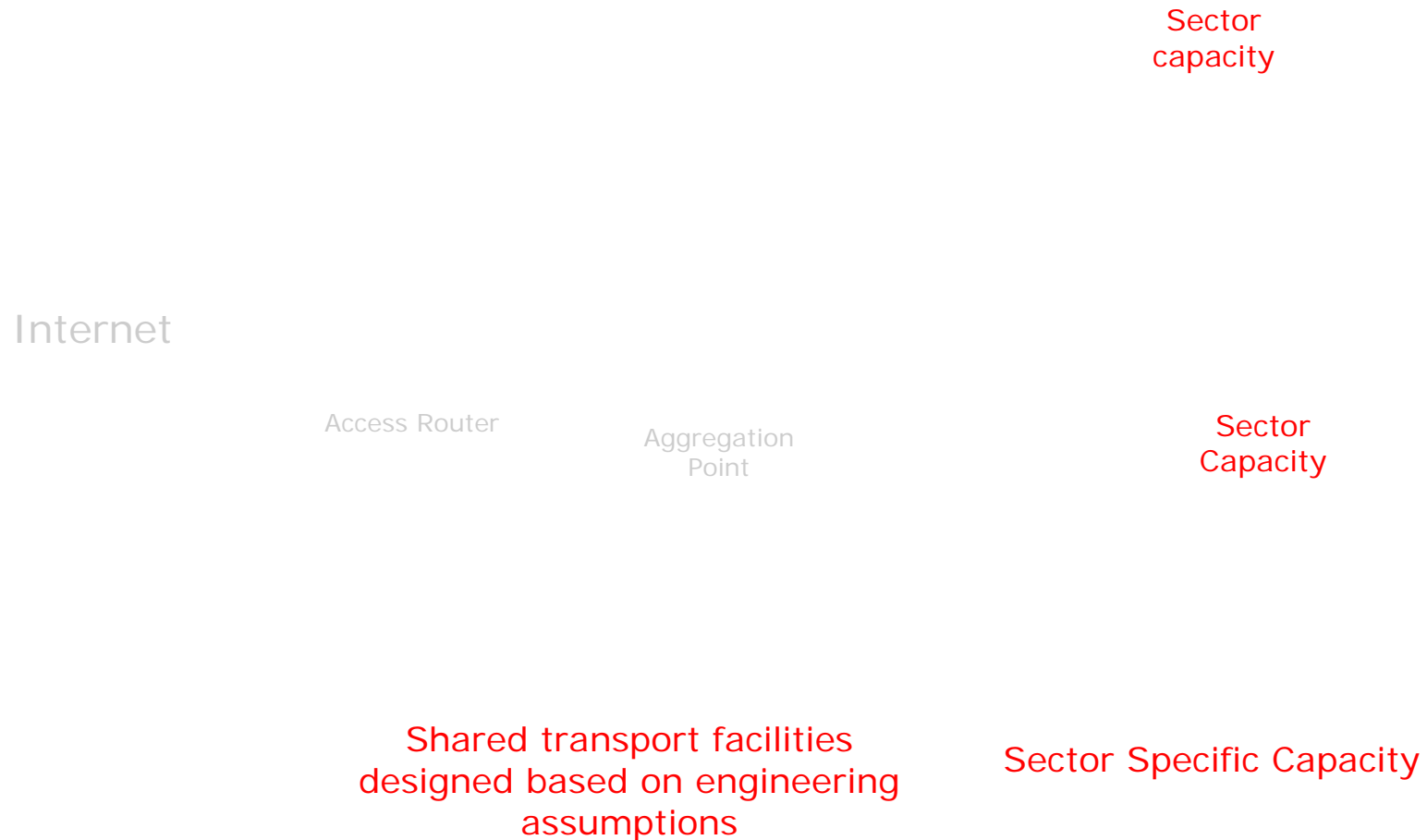
Access router

Shared among all homes served off the node.

Shared transport facilities designed based on engineering assumptions



Wireless Broadband Networks Are Inherently Shared Networks That Must Be Managed



Internet

Content Server

Your PC

Another PC

Another PC

Another PC

- In a client- server architecture, a file downloaded from the Internet comes from a central server to your computer.
- For a popular download, the server requires a high bandwidth connection to the Internet that all users downloading at the same time share.



Another PC

Another PC

Another PC

Your PC

- Peer-to-peer file sharing does not rely on a single central server.
- Content is divided into pieces and the pieces are downloaded from different computers to your computer.



Another PC

Arizona

Germany

Internet

Japan

Another PC

Another PC

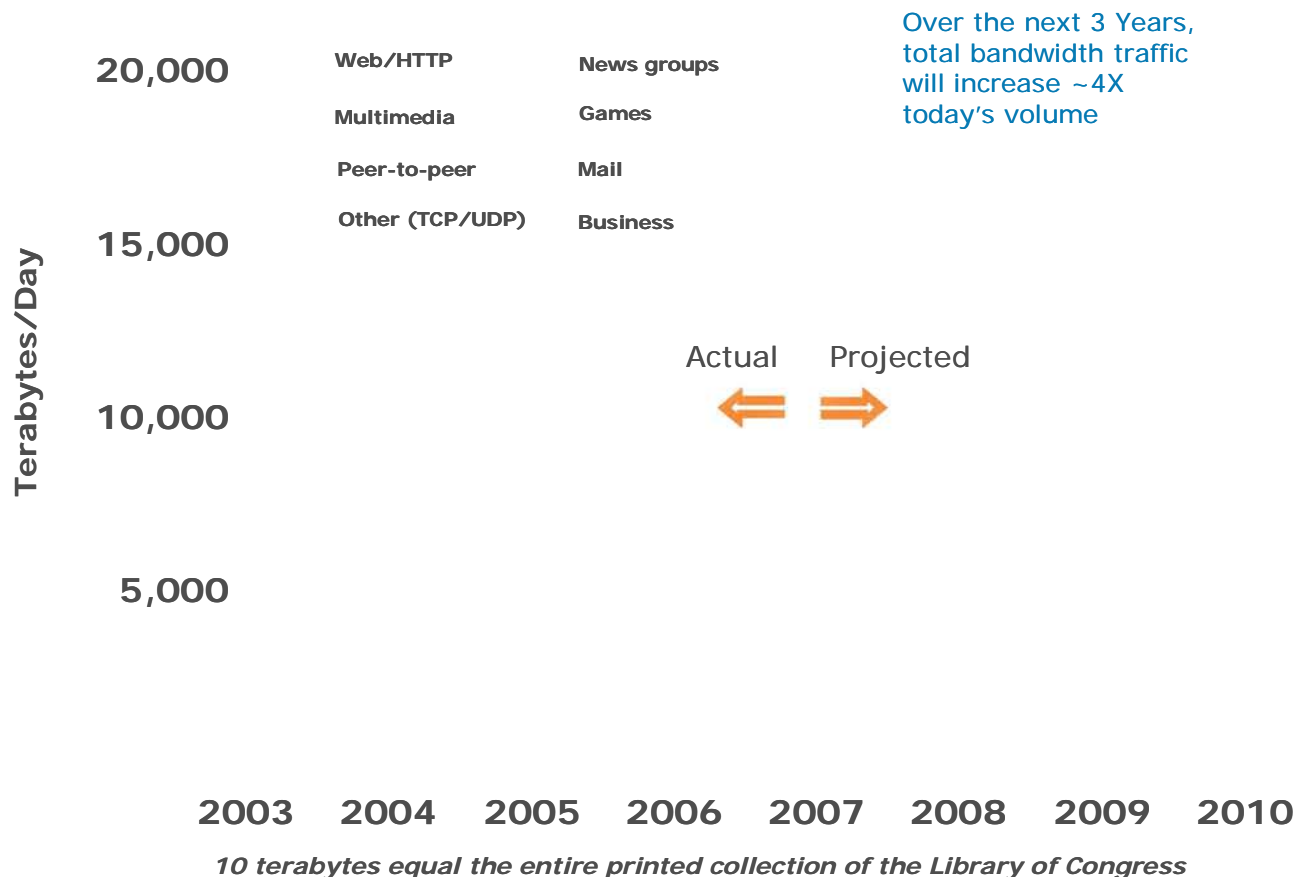
Your PC

Florida

- BitTorrent is one kind of peer-to-peer file sharing.
- Lack of proximity—peer could be down the street, across the country, around the world.
- Bandwidth used is spread across many different paths through the Internet - - upload and download.
- Network oblivious—may increase the load on shared facilities



AT&T IP Backbone Projected Traffic Growth



- Average AT&T End-user Bandwidth has increased approximately 35% per year during the 2001-2007 timeframe.
- Heavy bandwidth applications such as streaming media (Web & Multimedia) and Peer-to-peer are driving approximately 80% of total bandwidth on AT&T's Broadband Network.



Conclusion

- Certain P2P technology has upset network architecture assumptions – increasing the challenge and complexity of network management.
- DCIA P4P industry forum is looking at ways to increase the efficiency of P2P file sharing and lessen network impact.
- Increased bandwidth is not the solution because the need to manage shared networks doesn't go away as bandwidth increases.
- Resolve disputes as to what constitutes a reasonable network management practice on a case-by-case basis.



P4P: Explicit Communications for Cooperative Control Between P2P and Network Providers

Haiyong Xie[†] Arvind Krishnamurthy^{*} Avi Silberschatz[†] Y. Richard Yang[†]
^{*}University of Washington [†]Yale University

Abstract

The emergence of peer-to-peer (P2P) is posing significant new challenges to achieving efficient and fair utilization of network resources. In particular, without the ability to explicitly communicate with network providers, P2P applications depend mainly on inefficient network inference and network-oblivious peering, leading to potential inefficiencies for both P2P applications and network providers. In this paper, we propose a simple, light-weight architecture called P4P to allow more effective cooperative traffic control between applications and network providers. Our evaluations show clear performance benefits of the framework.

1 Introduction

A basic problem in a network architecture is how network applications (*i.e.*, network resource consumers) efficiently utilize the network resources owned by network providers. We refer to this problem as the network efficient traffic control problem, or traffic control for short. This problem is particularly important as it can have significant impacts on application performance, network provider efficiency and economics, and overall system complexity.

In the current Internet, for traditional point-to-point applications, efficient traffic control is largely determined by network providers alone: applications specify only the destinations of traffic; it is up to the network to control both the paths taken by the traffic and the transmission rates (through TCP feedback) on the chosen paths. Network providers can therefore improve efficiency unilaterally according to their objectives. Specifically, providers can use optimal traffic engineering to determine efficient routing and satisfy economical objectives such as implementing valley-free routing.

However, the recent emergence of P2P applications is posing significant challenges to efficient traffic control, with neither the network nor the P2P system having complete leverage over system efficiency.

First, for intradomain, the network-oblivious peering strategy of many P2P applications may cause traffic to scatter and unnecessarily traverse multiple links within a provider's network, leading to much higher load on some backbone links. A recent study [13] estimates that the aggregated traffic of all P2P applications contributes to about 50-80% of the traffic in many networks. This increasing P2P traffic has severe negative implications (see, *e.g.*, [11, 21]).

Second, for interdomain, network-oblivious peering

may cause a P2P application in a non-tier-1 network provider to relay a substantial amount of traffic between its providers [17]. This may lead to serious disruption of ISP economics. For example, in recent studies [5, 20] on Skype, the authors found that many universities (aka edge ISPs) are hosting a large number of Skype super nodes. Thus, they handle a large amount of transient traffic from and then to their providers, violating valley-free routing and leading to substantially higher operational cost. Even for tier-1 ISPs who do not make payments to network providers, P2P traffic may cause traffic imbalance between its peers, leading to potential violation of peering agreements.

Third, P2P's dynamic traffic distribution patterns do not necessarily enjoy a synergistic coexistence with network traffic engineering [10, 16] – network providers go to great lengths to estimate traffic matrices and determine routing based on them, but all of this effort could be negated if P2P applications modify their download behavior to adapt to changes in the network, thereby resulting in oscillations in traffic matrices and sub-optimal routing decisions.

In response to these kinds of issues, network providers have considered multiple new traffic control techniques. Unfortunately, none of them appear to be fully satisfactory – without P2P cooperation, the new techniques are either ineffective or degrade P2P performance and often times are too complex. One approach, for example, is to install P2P caching devices to cut down bandwidth consumed by P2P applications (*e.g.*, [6, 7, 18, 19]). However, these caches need to be designed for specific applications and speak the appropriate protocol, limiting their generality and applicability to closed protocols. Another technique is to deploy traffic shaping devices to rate limit P2P (*e.g.*, [2, 3]). These devices rely on deep packet inspection or other P2P traffic identification schemes. However, different P2P protocols use different control messages, and many P2P protocols use encryption and dynamic ports to avoid being identified. It remains unclear whether in the long run traffic shaping can effectively control the bandwidth consumption of P2P applications and reduce provider's operational costs. Furthermore, unilateral rate limiting by network providers may substantially degrade P2P performance and be at odds with consumer's needs.

With network provider solutions being ineffective, a few P2P systems have begun to investigate self-adaptation techniques, such as considering locality in peering (*e.g.*, [7, 12]), in order to achieve efficient traffic control. Although such

techniques have the potential to improve both network efficiency and application performance in certain settings, there are fundamental limits on what P2P can achieve alone. In particular, since traditionally traffic control is primarily performed by network providers, the current network architecture supports only *implicit communications* between applications and networks. Thus, to improve efficiency, P2P applications will have to depend on reverse engineering to determine network information such as topology, status and policies. However, this is challenging if not impossible.

Overall, the P2P paradigm exposes a fundamental issue in traditional traffic control: emerging applications can have tremendous flexibility in how the data is communicated, and thus, they should be an integral part of network efficient control. However, if end hosts are to participate in network resource optimizations, then the networks cannot continue to be opaque but need to export their status information.

We propose a flexible framework named P4P to enable better cooperation between P2P and network providers through *explicit communications*. Here P4P stands for proactive network provider participation for P2P, or provider portal for P2P. The objectives of P4P are to (1) facilitate network applications, in particular P2P applications, to achieve the best possible application performance under efficient and fair usage of network resources; and (2) allow network providers to achieve efficient and fair usage of their resources to satisfy application requirements, reduce cost, and increase revenue. Note that although our presentation focuses on P2P, it can be extended to other network application paradigms.

2 The P4P Framework

The P4P framework is a flexible and light-weight framework that allows network providers to explicitly provide more information, guidelines and capabilities to emerging applications, such as P2P content distribution.

2.1 Motivation

We now motivate the need for a P4P portal to enable explicit communications between P2P and network providers.

First, P2P systems have tremendous flexibility in shaping their traffic flow. Given that a client interested in a piece of data can download it from any one of the multiple sites storing the data, there are clear benefits to be had in intelligently choosing a data source (or, alternately, choosing a peer in a tit-for-tat system). This flexibility fundamentally changes the traditional network traffic control problem, which is typically solved in the context of a given traffic demand matrix. In the updated setting, there are multiple ways of satisfying the data demands of an application, each resulting in a different traffic demand matrix, and an efficient solution would require the explicit involvement of the P2P application.

Second, the current network architecture allows only for limited, implicit communications between network providers and applications. In this setting, if a P2P application seeks to exploit the flexibility in controlling its data transfers to improve efficiency, it will have to probe the network to reverse engineer information such as topology, status and policies. However, this is rather challenging in spite of significant progress in network measurement techniques. For one thing, it is clearly redundant and wasteful to have each

application perform probing. Even if this issue is addressed by a coordinated service for topology inference (*e.g.*, [12]) to reduce the overhead, the fundamental hurdle is the ability to perform the inference in an accurate manner. New technologies, such as MPLS, and routers that do not respond to measurement probes make it difficult to infer network characteristics. More importantly, available bandwidth and loss-rate estimation from end hosts are difficult because their views are obscured by last-mile bottlenecks; it is difficult for an end host to identify which links are under-utilized or over-utilized. Furthermore, cost and policy information are difficult, if not impossible, to reverse engineer. For example, it is difficult for P2P to determine which peers are accessible through lightly-loaded intradomain links and/or lower-cost interdomain links (where the cost takes into account factors such as inter-domain policies, traffic balance ratio between peering providers, and 95% percentile based billing).

In summary, for traditional applications, routing is made by network providers using a predictable traffic demand matrix with full network knowledge. With high levels of P2P traffic, the traffic control problem needs to be jointly solved by network providers and P2P applications.

2.2 Design Rationale

We consider the following design requirements.

- Better P2P performance. While some P2P systems exploit locality and network status to have its clients refine their peerings, the performance improvement is limited due to factors such as limited network information and slow convergence that is further exacerbated by churn [15]. Using more accurate network status information, P4P should be able to identify more efficient connections.
- More efficient network resource usage. By enabling explicit communication between P2P and the network, P4P can enable applications to use network status information to reduce backbone traffic and lower operation costs.
- Scalability. P4P should support a large number of users and P2P networks in very dynamic settings; any proposed information exchange and optimization techniques should be computationally inexpensive.
- Privacy preservation. P4P should address a major incentive concern of network providers who may want to preserve privacy when releasing their network information.
- Extensibility. There are many types of P2P applications with varying features. For instance, P2P systems for file sharing and streaming might have different needs, such as P2P streaming having more stringent real-time constraints than file sharing. Also, some applications use trackers (referred to as *appTracker* hereafter) to bootstrap and guide peer selection, while others do not; in addition, peers may exchange information locally through gossip messages. P4P should be flexible to handle a wide range of P2P applications with varying requirements and features.
- Incremental deployability. We do not target a clean-slate re-design. The P4P framework should be incrementally deployable, one network provider at a time, one P2P application at a time.
- Provider contribution for P2P acceleration. A network

provider may have many capabilities which it can provide to accelerate content distribution for P2P and at the same time increase its revenue. Examples include class of service, or quality of service that a P2P content provider can request. Also, a provider may contribute fixed servers as high-capacity seeds or caches, and this information should percolate to the P2P application.

2.3 Design Overview

The P4P framework consists of a control-plane component and a data-plane component.

In the control plane, P4P introduces iTrackers to provide portals for P2P to communicate with network providers. The introduction of iTrackers allows P4P to divide traffic control responsibilities between P2P and providers, and also makes P4P incrementally deployable and extensible.

Specifically, each network provider, be it a conventional commercial network provider (e.g., AT&T), a university campus network, or a virtual service provider (e.g., Akamai), maintains an iTracker for its network. A P2P client obtains the IP address of the iTracker of its local provider through DNS query (with a new DNS record type P4P). Standard techniques can be applied to allow for multiple iTrackers in a given domain, especially for fault tolerance and scalability. An iTracker provides a portal for three kinds of information regarding the network provider: network status/topology; provider guidelines/policies; and network capabilities.

In the data plane, P4P allows routers on the data plane to give fine-grained feedback to P2P and enable more efficient usage of network resources. Specifically, routers can mark the ECN bits of TCP packets (or a field in a P2P header), or explicitly designate flow rates using XCP-like approaches (e.g., [9]); end hosts then adjust their flow rates accordingly. For instance, a multihomed network can optimize financial cost and improve performance through virtual capacity computed based on 95-percentiles [4]. When the virtual capacity is approached, routers mark TCP packets and end hosts reduce their flow rates accordingly; thus the network provider can both optimize its cost and performance and allocate more bandwidth to P2P flows. We emphasize that the data plane component is optional and can be incrementally deployed.

2.4 P4P Control Plane

In this paper, we focus on the control plane of the P4P framework. Figure 1 shows the potential entities in the P4P framework: iTrackers owned by individual network providers, appTrackers in P2P systems, and P2P clients (or peers for short). Not all entities might interact in a given setting. For example, trackerless systems do not have appTrackers. P4P does not dictate the exact information flow, but rather provides only a common messaging framework, with control messages encoded in XML for extensibility.

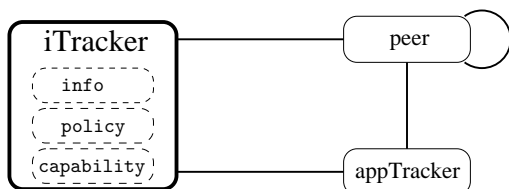


Figure 1. iTracker interfaces and information flow.

iTracker interfaces and functions

The key component of the P4P framework is iTrackers. iTrackers provide three interfaces that others can query.

The `info` interface allows others, typically peers inside the provider network, to obtain network topology and status. Specifically, given a query for an IP address inside the network, the interface maps the IP address to a (ASID, PID, LOC) tuple, where ASID is the ID of the network provider (e.g., its AS number), PID is an opaque ID assigned to a group of network nodes, and LOC is a virtual or geographical coordinate of the node. Note that the opaque PID is used to preserve provider privacy at a coarse grain (e.g., a network provider can assign two PIDs to nodes at the same point of presence or PoP). Note also that LOC can be used to compute network proximity, which can be helpful in choosing peers. When sending an `info` query, a peer may optionally include its swarm ID (e.g., info hash of a torrent). The iTracker may keep track of peers participating in a swarm.

The `policy` interface allows others, for example peers or appTrackers, to obtain policies and guidelines of the network. Policies specify how a network provider would like its networks to be utilized at a high level, typically regardless of P2P applications; while guidelines are specific suggestions for P2P to use the network resources. To name a few examples of network policies: (1) traffic ratio balance policy, defining the ratio between inbound and outbound traffic volumes, for interdomain peering links; (2) coarse-grain time-of-day link usage policy, defining the desired usage pattern of specific links (e.g., avoid using links that are congested during peak times); and (3) fine-grain link usage policy. An example of network guidelines is that a network provider computes peering relationships for clusters of peers (e.g., clustered by PID). The `policy` interface can also return a set of normalized inter-PID costs, which indicate costs incurred to the provider when peers in two PIDs communicate.

The `capability` interface allows others, for example peers or content providers (through appTrackers), to request network providers' capabilities. For example, a network provider may provide different classes of services or on-demand servers in its network. Then an appTracker may ask iTrackers in popular domains to provide such servers and then use them as peers to accelerate P2P content distribution.

A network provider may choose to implement a subset of the interfaces. The richness of information conveyed is also determined by the network provider. Note that a network provider may also enforce some access control to the interfaces to preserve security and privacy.

Examples

Now we give two examples to illustrate how the iTracker interfaces are utilized. Figure 2 shows an example P2P application with an appTracker using the `info` and `policy` interfaces to request network topology/status and guidelines/policies information. In the example, a P2P system spans two network providers A and B. Each network provider runs an iTracker for its own network. Peers *a* and *b* query their local iTrackers through the `info` and `policy` interfaces when bootstrapping; they then register with the appTracker and forward it the information obtained from

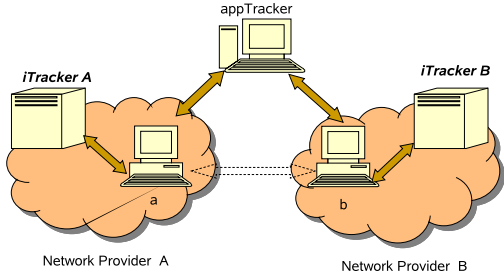


Figure 2. An example of P2P obtaining network topology/status and policies/guidelines from portal iTrackers.

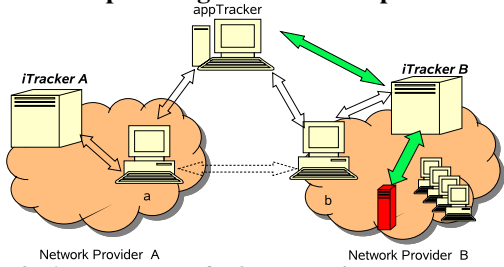


Figure 3. An example of P2P accessing network capability through iTrackers.

iTrackers. The appTracker makes peer selection considering both application requirements and iTracker information. Note that as a variant, it might be that instead of the peers query the iTrackers, the appTracker, trusted by a network provider (e.g., a major P2P developer), queries the iTrackers to reduce information exposure to general peers.

Figure 3 shows another example of using P4P. It shows how to request network capabilities through the `capability` interface. Specifically, the appTracker sends a request to iTracker B asking the network provider to allocate fixed, high-capacity servers to aid in distributing content. The iTracker allocates a server in its network and returns its address to the appTracker. The appTracker will then include the server in returned peer sets for those peers in B.

3 Evaluations

P4P’s effectiveness depends on what information is communicated through the portal and what algorithms are used for controlling traffic. We have evaluated the effectiveness of P4P using both simulations and real Internet experiments on PlanetLab. Our results show that P4P can improve not only P2P application performance but also network provider efficiency. Due to space limitations, we report on only a small fraction of our evaluations.

3.1 Optimization Methodology

We report our evaluations on how a network provider and peers can effectively utilize the `policy` interface. Specifically, we consider an optimization that minimizes intradomain traffic by taking into account swarm characteristics and current levels of background traffic. We consider the following setup. There are K swarms in a provider’s network. Each peer of a swarm obtains a unique swarm ID from the corresponding appTracker of the swarm, and reports it to the iTracker. The iTracker keeps track of the peers in a given swarm, including the number of peers at the same PoP, and the upstream and downstream link capacity of each peer. We refer to peers at PoP i as PoP- i peers. The iTracker then

computes the peering guidelines based on swarm statistics and network status.

Specifically, the iTracker constructs an abstract topology $G = (V, E)$ with nodes representing PoPs and edges inter-PoP links. The iTracker collects network status information including (1) b_e , the amount of background traffic on edge e , (2) c_e , the capacity of edge e , and (3) $I_e(i, j)$, the indicator of edge e being on the route from node i to j in G . The iTracker also computes \tilde{u}_i^k and \tilde{d}_i^k , the aggregated uploading and downloading capacity of all PoP- i peers of the k -th swarm in the network, respectively. Note that the iTracker can compute \tilde{u}_i^k and \tilde{d}_i^k using the information that it obtains from the P2P clients and from its knowledge of the access link capacities (both downstream and upstream) for each peer. The iTracker takes into account any user-imposed bandwidth limitations and aggregates them together to obtain u_i^k and d_i^k for each PoP i . The iTracker then computes $u_i^k = \max\{\tilde{u}_i^k - \tilde{d}_i^k, 0\}$, and $d_i^k = \max\{\tilde{d}_i^k - \tilde{u}_i^k, 0\}$. Thus u_i^k and d_i^k are the remaining uploading (supply) and downloading (demand) capacity that can be used to interface with peers in other PoPs.

The iTracker (periodically) solves a bi-level optimization problem to compute the peering guidelines: balancing the traffic (i.e., minimizing the maximum link utilization) while maximizing the overall throughput for each swarm:

$$\begin{aligned} \min \max_{e \in E} \quad & b_e + \sum_k \sum_i \sum_{j \neq i} t_{ij}^k I_e(i, j) / c_e \\ \text{s.t.} \quad & \forall k, \max \sum_i \sum_{j \neq i} t_{ij}^k, \\ & \text{s.t.} \forall \text{PoP } i, \sum_{j \neq i} t_{ij}^k \leq u_i^k, \\ & \forall \text{PoP } i, \sum_{j \neq i} t_{ji}^k \leq d_i^k, \\ & \forall i \neq j, t_{ij}^k \geq 0, \end{aligned}$$

where t_{ij}^k is the amount of traffic PoP- i peers upload to PoP- j peers in the k -th swarm. We solve this optimization problem as follows. We first solve the second-level problem to obtain the optimal throughput T_{opt}^k for each swarm k . This enables us to transform the k -th second-level problem into a constraint: $\sum_i \sum_{j \neq i} t_{ij}^k = T_{opt}^k$. Thus, we can solve the original problem. The iTracker derives the peering guidelines, which is a set of normalized weights $w_{ij}^k = t_{ij}^k / \sum_j t_{ij}^k$ for PoP i and j . A PoP- i peer would pick a PoP- j peer with probability w_{ij}^k . Next, the iTracker maps PoPs to anonymous PIDs and the computed weights to corresponding inter-PID peering probability values. The appTracker can obtain the inter-PID probability values and use it for peer selection.

3.2 Results

We have evaluated the effectiveness of P4P using both simulations and real Internet experiments on PlanetLab. For the simulation, we built a realistic BitTorrent simulator. For PlanetLab experiments, we use Liveswarms [14]. We chose these two P2P systems as one is file sharing while the other is streaming. Below, we report some of our evaluation results.

We first show the results on P4P-enabled BitTorrent, where the appTracker adopts iTracker suggested guidelines along with a small fraction of random inter-PoP peers for robustness. We use Abilene and the PoP-level topology of AT&T with all link capacities at 10Gbps. In the evaluations,

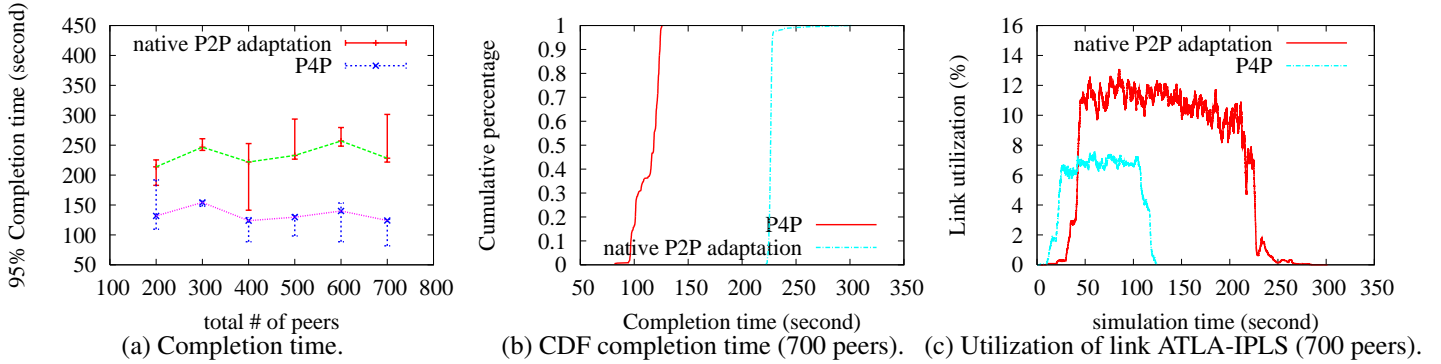


Figure 4. Results on integrating P4P with BitTorrent on Abilene.

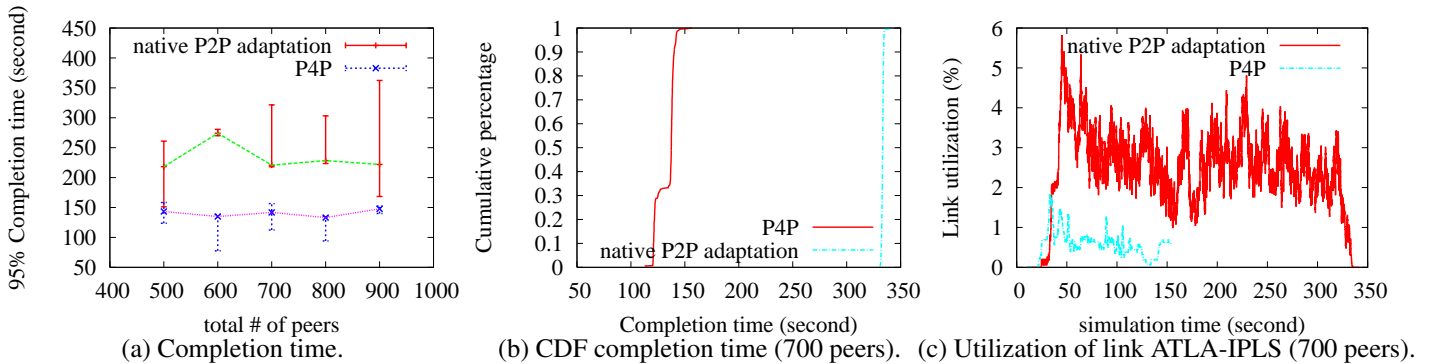


Figure 5. Results on integrating P4P with BitTorrent on AT&T PoP topology.

we connect each peer to a random PoP through an access link. The capacities of access links follow the distribution used in [1]. We evaluate two equal-sized swarms each sharing a 256MB file, with block size being 256KB. Initially each swarm has only one seed with 1Gbps upload capacity.

Figures 4 and 5 plot the completion time, cumulative completion time and link utilization for Abilene and AT&T, respectively, with a varying number of peers. We make the following observations. First, P4P improves P2P completion time by approximately 45%. Second, P4P improves the link utilization by approximately 50% and 70% in Abilene and AT&T, respectively, when compared with the native P2P adaptation. Further, P4P also reduces the duration of high traffic load by approximately a half, as peers finish their downloads faster. Thus P4P can reduce the intensity of P2P traffic load on the underlying network dramatically.

Next we report the evaluation results on integrating P4P with liveswarms. Liveswarms is a P2P-based application that adapts BitTorrent to video streaming. We conduct experiments on real Abilene network, using 53 PlanetLab nodes, to stream a large video file. Each experiment lasts 900 seconds. We log the amount of data exchanged between nodes, and compute the load on each Abilene backbone link using OSPF routing with Abilene's IGP link weights. The total amount of traffic on each Abilene backbone link is plot in Figure 6. We find that liveswarms achieves approximately the same throughput when integrated with P4P. However, the average link load is 1.1Gbps when native P2P adaptation is used, while the load reduces to 0.37Gbps when liveswarms is integrated with P4P. Thus P4P results in approximately 66% reduction on average link utilization.

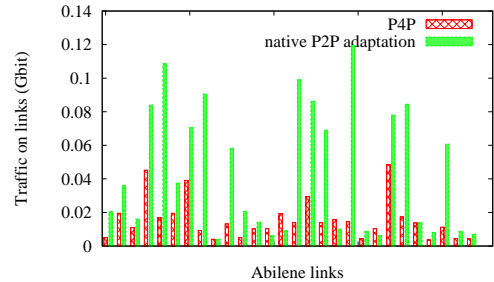


Figure 6. Traffic volumes when integrating P4P with a video streaming application.

Summary: P4P can improve both completion time and link utilization by approximately 50-70% for BitTorrent to share files, compared with the native P2P adaptation, on Abilene and AT&T networks. P4P achieve similar benefits in real experiments using liveswarms to stream video on real Abilene network. Further, P4P is robust to heterogeneity in the networks where links have different capacities.

4 Discussions

Q: [P2P Self Adaptation] Why cannot a P2P system achieve the benefits of P4P by itself? Or why do we need explicit communications between P2P and providers?

A: As we described in Section 1, it is difficult for applications to reverse engineer network status. It is even more difficult to infer provider routing policies. Although some P2P applications can use locality-based heuristics, there can be problems with this approach. For example, in a wireless network where P2P nodes communicate through a base station, peering using local peers sharing the same base station

may require more wireless bandwidth than through the base station to other non-local peers. As another example, a common issue exists in UK is that network providers buy their DSL “last mile” connectivity via a BT central pipe. More specifically, BT owns all of the exchange equipment and connectivity between a DSL customer and a central hand-off location. The connectivity from a DSL customer to its network provider is first routed through BT to a physical hand-off point. The hand-off point between BT and the network provider is what BT terms a BT central pipe. This connection can be many orders of magnitude more expensive than IP transit. Thus, it can be much more expensive for a network provider to have a customer retrieve a file from another customer on its network, than it is to go off the network for the file. Also, iTrackers provide a natural interface to access provider capabilities.

Q: [P2P incentives] It is clear that network providers can benefit from P4P. What are the incentives for P2P to participate in the P4P framework?

A: There are potentials for both network providers and P2P to benefit from adopting P4P, as we clearly demonstrated in our evaluations. In another set of experiments not shown here, we have shown that through P4P, a multihomed system (e.g., a university campus network) may allocate much more bandwidth to P2P without increasing its financial cost under a typical charging model. Furthermore, the P4P framework leaves much flexibility for P2P (e.g., P2P can integrate provider suggestions with its local application-specific requirements). On the other hand, without P4P, the providers may impose rate limiting to control their financial cost. Overall, many P2P developers are recognizing that as P2P consumes a significant portion of network resources without generating much revenue for providers, there is a real possibility that network providers may limit P2P usage to reduce cost. Thus, effective cooperation through the P4P framework can be attractive.

Q: [Scalable Implementation] There can be a large number of P2P networks in a provider’s network. How can it be feasible for the iTracker to handle the load associated with orchestrating all these networks?

A: Scalability could be addressed using several techniques. First, we need to consider optimizing only the heavy-hitters, namely those P2P networks that comprise of a large number of peers and generate a substantial amount of traffic. If a small number of P2P networks account for a large fraction of traffic, then the provider can focus its attention on those P2P networks. In order to quantify to what extent this phenomenon appears in practice, we analyzed the instantaneous swarm behavior of every movie torrent published by thepiratebay.org, a popular portal for BitTorrent content. In total, we analyzed 34,721 swarms to determine the number of leechers and the size of data being requested by the leechers. We find that only 0.72% of swarms had an excess of hundred leechers, and that the bulk of the instantaneous demand for content is from a small number of swarms, specifically, 1.22% of the swarms are responsible for about 50% of the demand. Analyzing and optimizing a small fraction of swarms should mostly suffice.

Second, we could replicate the iTracker functionality and further organize the iTrackers into a two-level hierarchy. The top-level server aggregates information from multiple P2P systems, solves the optimization problem, and distributes allocation decisions. The bottom-level servers are the ones contacted by the clients and are tasked with performing other operations, such as finding local connections. We just need to ensure that all peers within the same P2P network are directed to the same second-level server, and this could be done using a consistent hashing scheme [8].

Q: [Robustness] A major issue in P2P is to provide robustness. For instance, a BitTorrent client maintains a pool of randomly selected neighbors with which it is just exchanging meta-data information. Do the locality-aware P4P techniques reduce robustness?

A: P4P does not limit the mechanisms for improving robustness. In the basic operation mode, the iTrackers provide only hints, and an appTracker can always select a certain number of random connections to ensure diversity for robustness. This typically will not substantially increase the provider cost. A related robustness feature is that iTrackers are not on the critical path. Thus, if iTrackers are down, P2P applications can still make default application decisions.

5 Conclusion and Future Work

We presented P4P, a simple and flexible framework to enable explicit cooperation between P2P and network providers. Our evaluations demonstrate that it can be a promising approach to improve both application performance and provider efficiency. There are many avenues for further study. In particular, it is important to evaluate the framework, study what information should be communicated through the portal, identify what algorithms and techniques could be jointly employed by providers and P2P to improve efficient network traffic control, and quantify its benefits in large-scale, realistic networks.

6 Acknowledgments

Haiyong Xie and Y. Richard Yang were supported in part by grants from the U.S. NSF. Charles Kalmanek, Doug Pasko, Laird Popkin, Hao Wang, and Ye Wang have given us very valuable suggestions.

7 References

- [1] A. Bharambe, C. Herley, and V. Padmanabhan. Analyzing and improving a BitTorrent network’s performance mechanisms. In *Proceedings of IEEE INFOCOM ’06*, Barcelona, Spain, Apr. 2006.
- [2] Cisco. Network-based application recognition (NBAR). www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t8/dtnbarad.htm.
- [3] F5 White Paper. Bandwidth management for peer-to-peer applications. http://www.f5.com/solutions/technology/rateshaping_wp.html, Jan. 2006.
- [4] D. Goldenberg, L. Qiu, H. Xie, Y. R. Yang, and Y. Zhang. Optimizing cost and performance for internet multihoming. In *Proceedings of ACM SIGCOMM ’04*, Portland, OR, August 2004.

- [5] S. Guha, N. Daswani, and R. Jain. An experimental study of the skype peer-to-peer VoIP system. In *Proc of IPTPS*, Santa Barbara, CA, Feb. 2006.
- [6] K. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *Proc. of SOSP '03*, Bolton Landing, Oct. 2003.
- [7] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. Should internet service providers fear peer-assisted content distribution? In *Proceedings of the Internet Measurement Conference*, Berkeley, CA, Oct. 2005.
- [8] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *ACM STOC*, 1997.
- [9] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *Proc. of SIGCOMM*, Pittsburgh, PA, Aug. 2002.
- [10] R. Keralapura, N. Taft, C.-N. Chuah, and G. Iannaccone. Can ISPs take the heat from overlay networks? In *Proc. of HotNets-III*, San Diego, CA, Nov. 2004.
- [11] Lightreading.com. P2P plagues service providers.
- [12] 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. of OSDI*, Seattle, WA, 2006.
- [13] A. Parker. The true picture of peer-to-peer filesharing. <http://www.cachelogic.com>, July 2004.
- [14] M. Piatek, C. Dixon, A. Krishnamurthy, and T. Anderson. Liveswarms: Adapting bittorrent for end host multicast. Technical Report UW-CSE-06-11-01, 2006.
- [15] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. Do Incentives Build Robustness in BitTorrent? In *Proc. of NSDI*, 2007.
- [16] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker. Selfish routing in Internet-like environments. In *Proc of SIGCOMM*, Karlsruhe, Germany, Aug. 2003.
- [17] S. Seetharaman and M. Ammar. Characterizing and mitigating inter-domain policy violations in overlay routes. In *Proc of ICNP*, 2006.
- [18] G. Shen, Y. Wang, Y. Xiong, B. Y. Zhao, and Z.-L. Zhang. HPTP: Relieving the tension between ISPs and P2P. In *Proc of IPTPS*, Bellevue, WA, Feb. 2007.
- [19] A. Wierzbicki, N. Leibowitz, M. Ripeanu, and R. Wozniak. Cache replacement policies revisited: The case of p2p traffic. In *Proc of GP2P*, Chicago, IL, Apr. 2004.
- [20] H. Xie and Y. R. Yang. A measurement-based study of the skype peer-to-peer VoIP performance. In *Proc of IPTPS*, Bellevue, WA, Feb. 2007.
- [21] ZDNet News. ISPs see costs of file sharing rise. http://news.zdnet.com/2100-9584_22-1009456.html, May 2003.