SODA: Towards a Framework for Self Optimization via Demand Adaptation in Peer-to-Peer Networks

Haiyong Xie
University of Science & Technology of China
Hefei, China
haiyong.xie@ustc.edu

Bonan Min
Peking University
Beijing, China
min@cs.nyu.edu

Yafei Dai
Peking University
Beijing, China
dyf@cs.pku.edu.cn

***Abstract*—Peer-to-Peer (P2P) applications have been consuming an increasingly significant fraction of Internet bandwidth. They are becoming a financial burden to Internet Service Providers (ISPs), creating hot spots in the Internet, and causing potential performance degradation to other applications. As a result, there has been increasing tensions between P2P applications and network service providers. In this paper, we propose a framework called SODA for P2P applications to be self-adaptive and optimize their demands in order to more efficiently utilize network resources. Through preliminary experiments using two representative P2P applications, we demonstrate that SODA can effectively reduce bandwidth consumption by adapting the demands among peers.**

***Keywords*-Peer-to-Peer; self adaptation; optimization;**

I. Introduction

There have been increasing tensions between Internet Service Providers (ISPs) and Peer-to-Peer (P2P) applications. P2P applications (P2Ps for short) are largely oblivious to the underlying network infrastructure, leading to wide spread of P2P traffic in the Internet. As they become more and more popular, P2Ps consume an increasingly higher portion of network bandwidth. Studies (see,*e.g.*, [18]) estimate that the aggregated traffic of all P2Ps contributes to about 60-70% of the traffic in the Internet and about 80% of the traffic in the last-mile providers' networks.

The increasing P2P traffic has severe negative implications for overall network performance. On the one hand, P2Ps can generate a large amount of traffic. According to [22], a P2P application client consumes 90 times more bandwidth than an average Web client. P2P traffic can dramatically increase network utilization, and results in performance degradation to other applications (see, *e.g.*, [14], [31]). On the other hand, the increasing P2P traffic is becoming a financial burden to ISPs. Most ISPs pay their providers for the transit services to connect to the Internet. Being largely oblivious to network structures and policies, P2P clients in an ISP may upload a substantial amount of data to others in the provider networks. Hence, the ISP in question may actually become a content provider to its network providers and be charged unnecessarily [22], [23]. The world-wide extra costs due to P2P traffic are estimated to be in excess of € 500M per annum [18].

ISPs have clear incentives to manage P2P traffic. ISPs operate their networks, typically with complex internal structures, to achieve certain objectives. A common objective is to balance traffic through traffic engineering (see, *e.g.*, [7], [8], [20]). ISPs oftentimes estimate traffic matrices and determine routing based on them according to network policies. However, these information is generally not made available to P2Ps due to privacy concerns. As a result, P2Ps usually take the underlying networks as a black box with "bit pipes" inside it, being oblivious to network internal structures and objectives (*i.e.*, network-oblivious). The obliviousness can easily result in ISPs' effort being negated if P2Ps optimize their own application-level objectives.

Recently researchers have recognized these challenges and proposed novel approaches (see, *e.g.*, [1], [4], [6], [27], [28], [29]). In particular, Choffnes *et al*. [6] proposed the Ono framework for P2Ps to adjust themselves based on the redirection information obtained from existing content delivery networks (CDNs). Aggarwal *et al*. [1] and Xie *et al*. [28] proposed the Oracle framework and the P4P framework, respectively, to coordinate P2Ps and ISPs and jointly optimize the performance for both sides.

These proposals are clearly novel and beneficial; however, they also have many disadvantages. On the one hand, the Ono approach requires no support from the ISPs, but it heavily relies on inferring the status of the underlying networks from CDN redirection information. However, ISP objectives are not explicitly considered, and the inferred information can be biased, as the operations and objectives of CDNs can be significantly different from those of ISPs. Recent studies (see, *e.g.*, [12]) show that when decisions for content distribution by CDNs and decisions for traffic engineering by ISPs are made separately, as how CDNs and ISPs work today in the Internet, it leads to sub-optimal equilibrium. The equilibrium can still be sub-optimal even when network conditions are accurately and timely made available to CDNs. In addition, a lot of important network status information cannot be easily inferred via the CDN redirection information (*e.g.*, link virtual capacities [9], traffic engineering policies). On the other hand, the P4P-like approaches require explicit collaborations of both ISPs and P2P applications. However, many ISPs are reluctant to share their sensitive information in the collaboration, although some are actively involved in the past P4P field tests. Many P2Ps are actively pursuing collaboration with ISPs, but reluctant to migrate to the P4P framework, due to the potential risks of being too much dependent on the information ISPs provided via P4P. Another more serious risk is that the P4P framework is not able to differentiate harmful configurations (either intentional by malicious ISPs or unintentional by benign ISPs).

In this paper, we propose a framework called SODA: self optimization via demand adaptation, to address the challenges. In particular, the framework not only allows

978-1-4244-5067-1/09/$26.00 ©2009 IEEE

P2Ps to be collaborative without relying too much on information provided by ISPs, but also allows easy migration to the P4P-like approaches when such information is available and trusted in the future. To a certain extent, our framework combines the benefits of both P4P-like and Ono-like frameworks. In the short run, it allows P2Ps to be proactive in solving the tensions before the P4P-like frameworks are widely deployed; in the long run, it allows easy integration with the P4P-like frameworks once they are available. More importantly, it can be potentially used by P2Ps to differentiate harmful P4P-like information from beneficial ones and make intelligent decisions accordingly.

In our framework, we seek to exploit the tremendous flexibility that P2P applications have in forming peering relationships and choosing the replicas to download from. Similar to the P4P framework, we formulate the objectives of both ISPs and P2Ps explicitly, and formulate the demand adaptation as a constrained optimization problem. In the framework, P2Ps are self-adaptive, using active measurements to collect vital information about the underlying networks; these information will be used as inputs to solve the optimization problem. P4P-like information, when available and trusted, can be used to replace the inferred information and make better, more fine-grained decisions. Thus, our framework enables P2Ps to shape their peering relations as well as traffic distribution in order to both provide a certain level of performance guarantee for P2Ps and lower the traffic costs for ISPs.

We summarize our contributions as follows:

- we propose a framework to achieve the benefits of previously proposed P4P-like and Ono-like approaches, without explicit participation of ISPs. The framework considers typical objectives of both P2Ps and ISPs, and enables P2Ps to take a proactive role in the coordination.
- we leverage information collected from publicly available sources and active measurements, to exploit the tremendous flexibility of P2Ps. Our framework is also open and can leverage the fine-grained information via P4P when available.
- we implement the framework in two popular P2P file sharing applications, and demonstrate its effectiveness through Internet experiments.

The rest of the paper is organized as follows. In Section II, we introduce two popular P2P file sharing applications used in our studies. In Section III, we present the SODA framework for self optimization through demand adaptation. In Section IV, we describe how we implement the proposed framework in two P2P applications. In Section V, we present our evaluations methodologies. In Section VI, we present our evaluations results. In Section VII, we discuss related works. We conclude the paper in Section VIII.

II. Peer-to-Peer File Sharing Systems

We consider two representative P2P file sharing applications in this paper. The first application is BitTorrent. In BitTorrent, peers are self-organized to form a swarm to share files. The swarm is typically bootstrapped by one or multiple tracker servers. New peers join the swarm by registering with the tracker server and obtaining a randomly selected subset of active peers in the swarm. Then they establish connections to the selected peers and exchange chunks of the shared files. Peers are referred to as seeds when they have a completed copy of the shared files, and are referred to as leechers when otherwise.

The second application is Maze [30], a hybrid Napster-like P2P file sharing application. It is designed and developed by a research team in Peking University. Its user base is approximately 4 million. The number of simultaneous online users is typically 100,000 in Maze. There are approximately 200 to 300 million files shared by Maze users.

The Maze back-end servers are divided into three functional components. The first component is user servers, responsible for user management including user registration and login. The second component is central servers, responsible for building index for keywords and MD5 hash values of shared files. The third component is a cluster of query servers as a search engine to support file lookup and search.

It is common in Maze that a shared file has multiple replicas. As in other P2P applications, Maze clients can simultaneously download different portions of a shared file from multiple replicas. Maze uses a two-phase query operation to look up files and their replicas. The first phase is keyword-based query. In this phase, users supply keyword strings to the search engine, and results are grouped by their MD5 hash values. Results with the same MD5 hash value are merged into one search result when sent back to users. The second phase is hash-based query, which starts when the users choose a search result returned in the first phase. In the second phase, Maze clients automatically query Maze servers using the MD5 hash value to get currently active replicas in the system.

III. Self Optimization via Demand Adaptation

In this section, we first discuss the design rationale, then present our framework and describe how to exploit the flexibility of P2Ps.

A. Rationale

Our framework formulates the demand adaptation as a constrained optimization problem. Specifically, we explicitly take into account both ISP and P2P objectives. We would like to improve ISPs' performance, constrained by guaranteed P2P performance.

The rationale is that rather than optimize P2P performance alone and totally ignore ISP objectives, which is a common practice by most P2P application today, we believe that both objectives should be be taken into account to make P2P and ISPs co-exist well. We choose to improve ISPs' performance as the optimization objective because P2Ps typically have extremely good flexibility in shaping their traffic demands. In addition, ISPs' network resources are treated as common goods, it is clearly beneficial if P2Ps optimize themselves and avoid the tragedy of the commons.

B. A Framework for Demand Adaptation

We now describe the framework rigorously. We consider a set of clients of a P2P applications in a given ISP. We denote the ISP's network topology by $G = (V, E)$, where V and E are the set of nodes and directed links, respectively, in the underlying network. Let d denote a traffic demand

matrix, and $\mathcal{D}$ the set of feasible demand matrices of a P2P swarm. Here a demand matrix consists of the traffic volumes for all pairs of P2P clients. Note that $\mathcal{D}$ is used to facilitate the presentation only. In practice we do not need to know every single element in this set precisely.

We consider an abstract cost function, viewed by P2Ps, to characterize the cost incurred to the ISP by a given P2P traffic demand d. We denote the function by $c_{ISP}(d)$. We also consider an abstract utility function, denoted by $u(d)$ for a given P2P demand d. When given a set of feasible demands $\mathcal{D}$, a P2P application usually needs to guarantee a certain level of desirable utility $\underline{u}(\mathcal{D})$. Note that the utility is a characterization of P2P performance, and $\underline{u}(\mathcal{D})$ is used as a lower bound.

The objective of the demand adaptation is to distribute P2P demands, by choosing a demand matrix from the feasible set, so that not only the P2P performance can be guaranteed to achieve the lower bound, but also the impacts on ISP can be optimized. This is formulated in Figure 1.

$$\begin{aligned} \min_d \quad & c_{ISP}(d) \\ \text{subject to} \quad & u(d) \geq \underline{u}(\mathcal{D}), \\ & d \in \mathcal{D}. \end{aligned}$$

Figure 1. General SODA framework.

P2P typically needs more network information to compute $c_{ISP}(d)$. However, such information is usually not made available to P2P, as it reflects internal structures of the underlying network and is considered as confidential by ISPs. Thus, P2P should leverage various measurement techniques to construct the necessary information if possible at all.

Information such as network topology G and its routing can be effectively obtained through various inference approaches (*e.g.*, [15], [16], [24]). Such inferred information is the best available information that P2P can obtain alone without the ISP's cooperation. These information are still valuable even when disclosed by ISPs. For instance, it can potentially be used by P2Ps to correlate the inferred and the officially disclosed information, identify misleading information (*e.g.*, due to ISP misconfigurations or maliciousness) and make intelligent decisions accordingly.

In this paper, we consider the bit-distance product (BDP), denoted by c_{ISP}^{BDP}, as the ISP cost function viewed by P2Ps. BDP is the sum of the distances each bit traverses from the source to the destination, *i.e.*,

$$c_{ISP}^{BDP} = \sum_{i \in V, j \in S(i)} t_{ij} p_{ij},$$

where p_{ij} is the distance from node j to i, and t_{ij} the volume of traffic flowing from node j to node i. We denote by $S(i)$ the set of replicas that i can download from.

The distance p_{ij} can be either the number of intermediate links or the sum of geographical distances of each individual link along the path from j to i. This information can be inferred through aforementioned active measurements. Note that in the P4P context, p_{ij} is the P4P distance from i to j, and such information is made available by the ISP via P4P only. In the SODA framework, p_{ij} can be adjusted to favor replicas in the same PoP, in the same regional area, or in the same ISP network. For instance, the P2P can assign a larger value to replicas in another ISP in order to limit the traffic crossing the ISP network boundary.

To make it concrete, we next describe two instances of the framework using Maze and BitTorrent as examples.

C. Maze Demand Adaptation

We next consider the P2P utility and constraints in Maze. Due to the search-based dynamics of file sharing behaviors, new demands may arrive before old demands complete. To make it conceptually clean, we divide the demands into two categories: existing demands Z_{ij} and new arrival demands T_{ij}. Note that it may not be necessary to satisfy every new arrival demands all of the time.

The P2P utility is the total satisfiable traffic volume of the new demands. The P2P constraints are that (1) at each node i, at least a specified proportion (denoted by α) of new demands should be satisfied; and (2) to maintain diversity in P2P demands and peering relationships, clients at node i and j should exchange a minimum level of traffic (denoted by β) when feasible. Typically for all i, $\sum_{j \in S(i)} \beta < \alpha$. Here we treat α as the lower bound on P2P throughput, and β the minimum percent of feasible traffic. In other words, a subset of the new arrival demands should be satisfied in order to meet the lower bound of P2P performance and diversity requirements. Figure 2 shows the formulation.

$$\begin{aligned} \min \quad & \sum_{i,j \in S(i)} (t_{ij} + Z_{ij}) p_{ij} && (1) \\ \text{subject to} \quad & \forall i, \sum_{j \in S(i)} t_{ij} \geq \alpha \sum_{j \in S(i)} T_{ij}, && (2) \\ & \forall i, j \in S(i), t_{ij} \geq \beta T_{ij}, && (3) \\ & \forall i, j \in S(i), t_{ij} \geq 0. && (4) \end{aligned}$$

Figure 2. BDP SODA for Maze.

Note that α and β are pre-determined system variables and have significant impacts on both ISP and P2P. We refer to α as the performance tolerance factor (or tolerance factor for short) and β as the diversity factor hereafter.

D. BitTorrent Demand Adaptation

We note that BitTorrent clients do not have to connect to a specific set of peers, but can rather obtain the distributed content from any set of active peers.

We compute the maximum P2P throughput for a BitTorrent swarm using the model proposed in [28]. Specifically, we denote by a_i and b_i the aggregated upstream and downstream access capacity of all peers at underlying node i. The access capacity information can be obtained from either the P2P clients themselves or the knowledge of network access link capacities for each peer. Note that a_i and b_i also takes into account the user-imposed bandwidth limit. Thus, the objective is to maximize the total throughput $\sum_{j \neq i} t_{ij}$, subject to the aggregated downstream and upstream bandwidth constraints $\sum_{j \neq i} t_{ij} \leq b_i, \forall i \in V$ and $\sum_{j \neq i} t_{ji} \leq a_i$, $\forall i \in V$, where $t_{ij} \geq 0, \forall i, j \in V$. We denote by t_{ij}^* the optimal solution to the above problem, and by $T_i^* = \sum_j t_{ij}^*$ the optimal aggregated throughput of all clients at node i.

The realization of SODA in BitTorrent is to address the throughput considerations of BitTorrent swarms and the traffic load concerns of the ISP. The formulation using c_{ISP}^{BDP} as objective can be derived similarly, by replacing the

constraint (2) in Figure 2 with the following:

$$\forall i, \sum_{j \in S(i)} t_{ij} \geq \alpha T_i^*.$$

E. Distributed Solution

It is desirable to have a distributed approach to solving the demand adaptation problems in the preceding subsections, since many P2P applications are highly decentralized. In such scenarios, no individual servers will be appropriate for making centralized decisions.

We note that the above formulations have convex objectives and linear constraints. It is straightforward to apply classical distributed algorithms to solve such optimization problems [2]. We adopt the feasible steepest descent algorithm and update the demands at each step converging to the optimal solution as follows:

$$t_{ij}(n+1) = [t_{ij}(n) - \delta(n) p_{ij}]^+,$$

where $\delta(n)$ is the step size at n-th step, and $[\cdot]^+$ is the projection onto the feasible set $\{t_{ij} | t_{ij} \geq 0, \sum_{j \in S(i)} t_{ij} \geq \alpha \sum_{j \in S(i)} T_{ij}\}$ for Maze (similarly, $\{t_{ij} | t_{ij} \geq 0, \sum_{j \in S(i)} t_{ij} \geq \alpha T_i^*\}$ for BitTorrent).

In practice we implement the algorithm for Maze as follows. Consider a client at node i. We first sort the remote replicas in ascending order of distances from i, and assign a minimum fraction of demand $\beta \sum_j T_{ij}$ to each remote replica at node j. We then divide the remainder of the demand (*i.e.*, for a given i, $(\alpha - \sum_{j \in S(i)} \beta) \sum_{j \in S(i)} T_{ij}$) equally to the remote replicas closest to node i.

F. Extension of ISP Cost Functions

The framework can be extended to include other abstract cost functions viewed by P2Ps. One example is maximum volume of link traffic c_{ISP}^{MVL}.

We denote by $I_{ij}(e)$ an indicator variable, which equals 1 when link e is on the path from i to j, and 0 otherwise. The path is determined by the ISP's routing algorithm. $I_{ij}(e)$ can be inferred, *e.g.*, via the approach in [16]. Note that the indicator variable is a generic representation of the routing algorithm, and it frees us from the specific details of routing algorithms.

Specifically, the constrained optimization problem can be formulated as follows:

$$\begin{aligned} \min \quad & max_{e \in E} \sum_{i \in V, j \in S(i)} t_{ij} I_{ij}(e) \\ \text{subject to} \quad & (2), (3), (4). \end{aligned}$$

Figure 3. MVL SODA for Maze.

By minimizing the maximum link traffic volume, P2Ps can distribute their traffic evenly inside the network. Similarly, we can apply the steepest descent approach to derive iterative distributed algorithms to adapt P2P demands converging to the optimal solution.

IV. Design and Implementation

In this section, we describe our implementations of the SODA framework for Maze and BitTorrent.

A. Network topology

We collect the PoP-level topologies for two networks. The first network is CERNET. We take the BGP RIB dumps and measurement studies by the CERNET BGPView Project [3] to construct the topology. The CERNET topology consists 36 PoP nodes in major cities of China, and 56 bi-directional links. The second network is Abilene. Its topology is made publicly available by Internet2 Network NOC [11]. Abilene consists of 11 PoP nodes and 28 bidirectional links. For each link in both networks, we also obtain the capacity and weight. Based on the link weights, we are able to compute the routing. However, the detailed topology (with link capacity and weight information) is used in post-experiment analysis only.

We also use the BGP RIB dumps to collect the IP prefixes that are originated at each PoP node in both topologies. Thus, given an IP address, we are able to look up which PoP node it originates from.

B. SODA Implementation in Maze

We adopt a centralized SODA implementation in Maze since many clients do not automatically install the updated SODA patch. However, the centralized implementation is only to facilitate the experiments, and it should be fairly straightforward to implement the distributed solution at the client side.

In our implementation, when the index server receives the hash-based search queries issued by clients, it first looks up the locations of all active replicas for each hash value, and forwards them to the SODA server. The SODA server regularly examines the forwarded search queries with associated locations of replicas, and solves the optimization problems described in the preceding section. In other words, The SODA server periodically selects a subset of replicas for each incoming query, and returns the chosen replicas to the index server. Then the index server returns them to the clients. The SODA server can be considered as a filter to prune locations of replicas for each incoming query. Note that the SODA server only takes input from the second phase in the Maze query process.

C. SODA Implementation in BitTorrent

We implement the SODA server in BitTorrent tracker server. In our implementation, the tracker server estimates the aggregated upstream and downstream access capacity, based on the reported peer activity information. Then it computes the desired optimal throughput by solving the throughput maximization problem described in the preceding section. The tracker server next computes the desired demand matrix. Then it maps the desired demand matrix to peer connectivity as follows.

Once the t_{ij} values have been computed, we derive the weights $w_{ij} = t_{ij} / \sum_j t_{ij}$. A peer at node i would then use these weights to make randomized peering connections. Specifically, a peer at node i would select peers at j with probability w_{ij}. This scheme operates at a coarser grain. In our implementation, we do not achieve strict bounds on traffic between underlying nodes, but instead control only the number of connections in a probabilistic manner.

V. Evaluation Methodology

We use real Internet experiments to evaluate the effectiveness of SODA.

A. P2P Data Collection

We collect the client activity logs in experiments as follows. We modify the BitTorrent client program to report data exchange statistics in 5-minute intervals. Maze client has built in such activity report functionality but at a coarser granularity. Once Maze query server receives a query from a client, it forwards the query to the SODA server. While in BitTorrent, each record is aggregated transferred traffic volumes between a pair of peers. Therefore we can derive real-time traffic demands.

We then derive the traffic demands for each pair of PoP nodes from the logs. The traffic demand is estimated for each 5-minute interval in each day. We assume for simplicity that the download rate is the average traffic rate with respect to each session (*i.e.*, total number of downloaded bytes divided by the session length). Then for each 5-minute interval, we compute the total number of bytes downloaded by all sources at one PoP node from all destinations at another PoP, which we refer to as a 5-minute traffic demand matrix. To this end, we convert the download log to a set of 5-minute traffic demand matrices.

Based on the P2P traffic demand and the detailed network topology, we compute the amount of traffic and utilization for each link, the total amount of traffic in the network, and the sum of bit-distance product for each demand.

B. Internet Experiment

We integrate the SODA server in Maze and run Internet experiments in the CERNET network. We run multiple sets of Maze Internet experiments using the inferred CERNET topology in 2-hour durations. To evaluate SODA intensively, we intentionally choose to run the experiments in the busiest hours (in the late afternoon and night) in multiple days. During those time slots, the number of online users and total traffic in Maze is the highest in a day. We then collect and compute the traffic demands from the logs, and compute the link utilization using the detailed topology accordingly. We refer to the original Maze as Native and Maze with SODA as SODA for short.

We also integrate the SODA server with the BitTorrent tracker server and run Internet experiments in the Abilene network. We set up an initial seed server hosting a 10MB file, run a separate tracker server, and run BitTorrent clients from 106 PlanetLab nodes in Abilene network to download the file. All clients join the swarm in a short time period to emulate the flash crowd. Each experiment lasts 20 minutes. We run the experiments in late nights when the load on Abilene and PlanetLab nodes is relatively light. We run the experiments multiple times and compute the averages.

VI. Evaluations

In this section, we first present Maze measurement results, as they illustrate the undesirable impacts P2Ps can have on ISPs. We then present preliminary experimental results for both Maze and BitTorrent.

A. Maze Measurements

We collect and analyze logs of Native Maze to understand the impacts it has on the underlying network. The logs used in analysis were collected from September 1st, 2007 to October 3rd, 2007.

We investigate how the underlying network links are utilized by Native Maze. We compute the utilization for each link and each day. We also compute the average and maximum utilization for all links across all days.

Figure 4 plots the utilization of two most heavily used links. The capacity of both links is 2.5Gbps. We observe that the maximum utilization can be as high as 14% resulted by Maze *alone*. The average utilization during busy hours is approximately 5% and 2% for the two links, respectively. We emphasize that these link loads are resulted by Maze alone. This clearly shows that Native Maze can creates very high loads on certain links, thus potentially increasing ISP operational costs and degrading the performance of other applications that share those links.

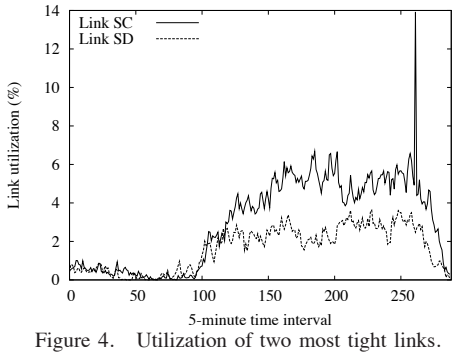


Figure 4. Utilization of two most tight links.

Figure 5 plots the statistics of utilization of all CERNET links. We observe that although the average link utilization is relatively low, some links are heavily used by Maze. This clearly shows that in Maze, the traffic is not distributed evenly. Native Maze does not optimize for traffic distribution, thus it is likely to create hot-spots in the underlying networks. With SODA server explicitly optimizing traffic distribution, we believe that the utilization of the underlying networks can be improved.

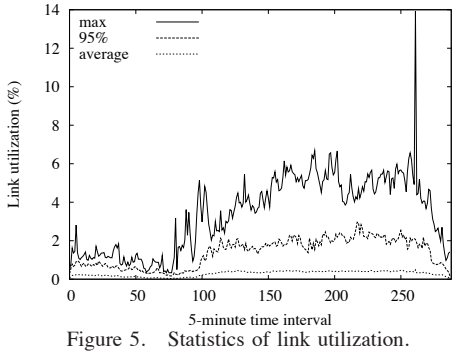
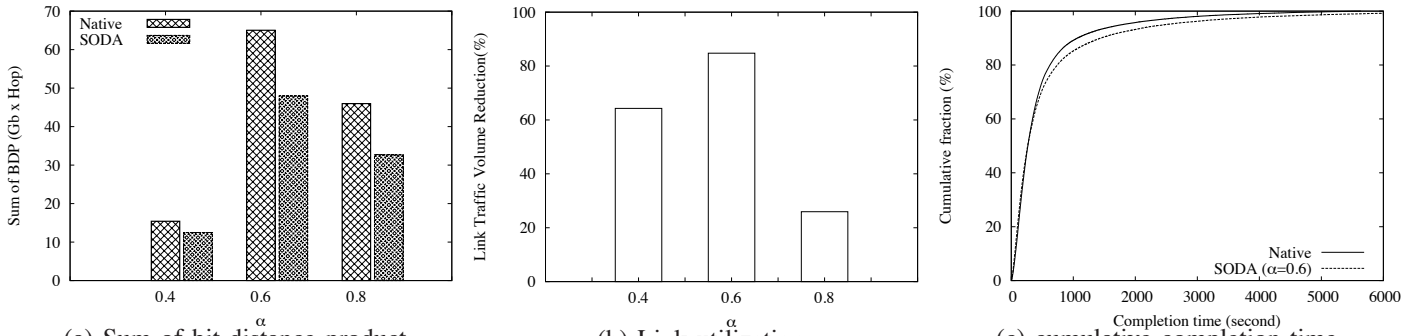


Figure 5. Statistics of link utilization.

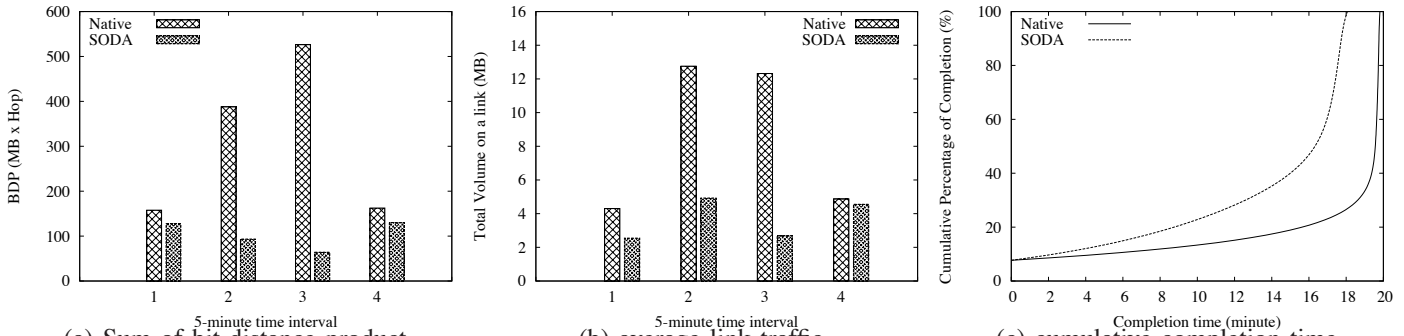
B. Maze Internet Experiments

We now present the results of Maze Internet experiments with integrated SODA server. We configure the SODA server and run the experiments with α varying from 0.2 to 0.8 and β fixed to 0.01. Here we present the experimental results when α being 0.4, 0.6, and 0.8 only.



(a) Sum of bit-distance product. (b) Link utilization. (c) cumulative completion time.

Figure 6. Maze: summary of improvement.



(a) Sum of bit-distance product. (b) average link traffic. (c) cumulative completion time.

Figure 7. BitTorrent: summary of improvement.

Figure 6 summarizes the improvement of SODA when varying the factor α and negative impacts on client download time.

Figure 6(a) plots the result of the improvement on bit-distance product. We compute the sum of bit-distance product by aggregating all traffic volume between any pair of PoP nodes in the underlying networks. We observe that SODA improves bit-distance product. Note that in these experiments, SODA indirectly improves BDP by optimizing link utilization which results in evenly distributing and localizing the traffic in the network, and as a side effect, this usually leads to lower BDP.

Figure 6(b) plots the amount of improvement on link utilization at different levels of tolerance factor. We observe that SODA can significantly improve the utilization of links in the underlying networks. When α is small, the total amount of traffic allowed by SODA is also small, thus the performance improvement on link utilization is larger. When α increases, the amount of improvement decreases accordingly. Thus aggressive tolerance factor (*e.g.*, $\alpha = 0.8$) leads to only marginal improvement. The experimental results suggest that an appropriate tolerance factor ranges from 0.4 to 0.6.

Figure 6(c) plots the cumulative download completion time when $\alpha = 0.6$. We observe that SODA results in the download completion time approximately the same as Native Maze. For downloads lasting less than 600 seconds, the completion time is almost the same; for downloads lasting longer, SODA results in less than 10% longer completion time. Thus the performance degradation to Maze with SODA is relatively small. The results with smaller α values are consistent; thus we do not present them here.

C. BitTorrent Internet Experiments

We next present the results of BitTorrent experiments in the Abilene network. The tolerance factor α is set to 0.6 and β set to 0.01 in the experiments. We collect BitTorrent client logs and compute the amount of traffic volumes for all links in Abilene network. We also compute the sum of bit-distance products. Figure 7 summarizes the results.

Figure 7(a) compares SODA against Native BitTorrent on the sum of bit-distance product. We observe that SODA results in significantly lower bit-distance product. In the 2nd and 3rd intervals where peers are exchanging large amount of traffic, the improvement is as high as 75%. However, in the starting and ending intervals, SODA has less improvement as data exchange among peers is no longer extensive. The results suggest that most of traffic is localized to local peers in the same PoP or close-by PoPs when peers are extensively exchanging data.

Figure 7(b) shows significant improvement on the traffic volume on Abilene links. SODA has improvement as high as 60% when traffic exchange is extensive in the 2nd and 3rd intervals. The improvement is marginal in the starting and ending intervals. Note that the traffic volume numbers on links can be directly translated into link utilization.

Figure 7(c) shows the improvement on the download completion time. We observe that SODA results in faster completion. The median improvement is approximately 20%, and some peers can see improvement as high as 40%. The reason is that optimizing BDP leads to peers favoring shorter connections and local data exchange; as a result, the shorter

connections usually achieve higher throughput than long connections, and the speed of piece propagation is faster among locally clustered peers.

Figure 8 plots the ratio between bit-distance product and total traffic volume. This ratio is a good indicator of how well traffic is localized across the underlying network. We observe that Native BitTorrent results in significantly higher ratio. We further analyze the logs and find that the traffic between pairs of PoP nodes are approximately equalized in Native BitTorrent, regardless of how far or close two PoP nodes are in terms of path length. This suggests that Native BitTorrent is not able to distribute the traffic in the desirable way. This is resulted by random peer selection implemented in Native BitTorrent, leading to network-oblivious traffic distribution among all pairs of PoP nodes.

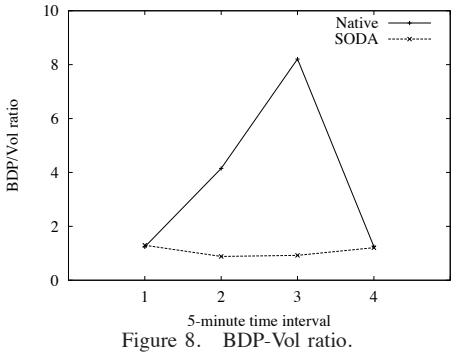


Figure 8. BDP-Vol ratio.

We also observe that the ratio increases initially and then decreases; however, in SODA experiments, the ratio is almost constant. The reason is that in Native BitTorrent experiments, peers are more densely connected. Initially only a small number of clients who peer with the seed are able to download a portion of the file. Gradually, more clients have downloaded chunks to share with others, and they start to exchange data extensively with neighbors. However, traffic is randomly spreaded across the network, resulting in increasing bit-distance product. The ratio gradually decreases when clients completes downloading. The results suggest that SODA can efficiently localize the P2P traffic and achieve a significantly more efficient utilization of network resources.

VII. Related Work

Researchers have extensively studied how to design P2P applications to share network resources and address the increasing tensions. The literature can be largely categorized into four lines.

The first line of research is adaptation based on congestion control. Researchers have investigated how to design TCP-like congestion control protocol for "lower-than-best-effort" services (*e.g.*, TCP-LP [13] and TCP Nice [25]). This approach can be applied to P2P design to reduce contention to best-effort traffic. However, it does not address the tensions between ISPs and P2Ps directly; instead, it relies end-to-end congestion control mechanisms to fairly share bandwidth on network links. This approach is complementary to ours.

The second line of research is P2P traffic caching. Researchers have studied the feasibility of various P2P traffic caching and showed that P2P traffic caching would potentially reduce wide-area bandwidth demands of P2P systems dramatically (see, *e.g.*, [10], [22], [21], [26]). There are also commercial P2P caching devices available (*e.g.*, [5]). However, many ISPs are reluctant to be involved in the P2P content distribution due to legal concerns. Further, P2P traffic caching is usually specific to individual applications, meaning that the caching devices have to implement a P2P protocol before being able to cache its traffic.

The third line of research is P2P self-adaptation (see, *e.g.*, [4], [6], [19]). In particular, Choffnes *et al*. [6] proposed the Ono framework for P2Ps to infer closeness of peers via the redirection information of existing content delivery networks, and adapt themselves accordingly. These approaches are clearly novel and promising; however, they do not consider ISP objectives and policies explicitly. For example, in the Ono approach, the redirection information may reflect the objectives of CDNs *only*, while ISP objectives may not well aligned with CDN objectives, as suggested by [12]. In addition, the effectiveness of this approach depends heavily on factors such as CDN coverage, deployment and granularity.

The fourth line of research is explicit coordination between ISPs and P2Ps (see, *e.g.*, [1], [4], [27], [28]). In particular, Aggarwal *et al*. [1] and Xie *et al*. [27], [28] independently proposed the Oracle framework and the P4P framework, respectively. The Oracle framework ranks P2P clients according to ISPs' internal metrics, while the P4P framework distributes p-distance information explicitly derived from ISP objectives and constraints (*e.g.*, intradomain and interdomain policy). Both approaches are largely dominated by ISPs, as ISPs have to deploy them and make them available to P2Ps. Otherwise, it would be difficult for P2Ps to obtain the ISP information. However, the disadvantages of both approaches are: (1) only when a sufficient number of ISPs deploy these frameworks can P2Ps make practical and effective use of them; (2) even both frameworks can preserve privacy to some degree, many ISPs are still conservative and reluctant to disclose sensitive information; and (3) in addition to lack of trust by P2P, ISPs may provide harmful information (either intentionally or unintentionally).

Our approach differs from the above approaches in that we provide a P2P-dominant framework to achieve benefits of both Ono-like and P4P-like approaches. We explicitly take into account the objectives of both ISPs and P2Ps, and address the disadvantages of the P4P-like approaches. Our approach is flexible: it can not only work when P4P-like network information is unavailable, but also easily integrate the P4P-like network information once it is available, without fundamentally changing peer behaviors. Our approach is also potentially helpful for P2P to differentiate harmful ISP information and make intelligent decisions.

VIII. Conclusion

We propose a framework called SODA for peer-adaptive demand optimization. The framework can be leveraged to achieve the benefits of recent P4P and Ono proposals. Inspired by these proposals, we seek to exploit the tremendous flexibility that P2P nodes have in choosing the replicas to download from in Maze and forming peering relationships

in BitTorrent. We implement the framework in these two popular P2P file sharing applications. Although preliminary, the Internet experiments demonstrate its effectiveness even without the strong requirement for network information as needed in P4P-like frameworks.

There are multiple venues for future works. First, the experiments are still preliminary. Second, the tolerance factor α controls how aggressive P2P optimizes itself. In addition, there exists conflict of interest between ISPs and P2Ps, and different alpha values will favor different interest groups. However, there still lacks thoroughly understanding of the trade-offs in P2P and ISP performance. Third, we require that network information be inferred and collected when such information is not available. Significant progresses have been made towards this direction in the past few years (*e.g.*, [15], [16], [17], [24]); however, this may still result in overhead in SODA, especially when such inference infrastructure is not widely available yet. In fully decentralized P2Ps without trackers, such overhead may be even higher. Fourth, the inferred network information may not be as accurate. We will investigate the impacts of the inaccuracy on P2P and ISP performance as well as whether multiple P2Ps implementing SODA co-exist well in one ISP network. Last, we will investigate when network information is available from both inference and ISPs, how P2P should correlate them and identify harmful ISP configurations. We will also extend the framework to handle multiple ISPs in interdomain settings.

IX. Acknowledgments

Bonan Min and Yafei Dai were supported in part by China Grand Fundamental Research Grant No. 2004CB318204 and China Natural Science Foundation Grant No. 60873051. We are grateful to our shepherd Sonja Buchegger and anonymous reviewers for their detailed and helpful feedbacks.

References

[1] V. Aggarwal, A. Feldmann, and C. Scheideler. Can ISPs and P2P systems cooperate for improved performance? *ACM CCR*, July 2007.

[2] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2nd edition edition, 1999.

[3] CERNET BGPView Project. http://bgpview.6test.edu.cn/.

[4] R. Bindal, P. Cao, W. Chan, J. Medval, G. Suwala, T. Bates, and A. Zhang. Improving traffic locality in bittorrent via biased neighbor selection. In *Proceedings of IEEE ICDCS '06*, Lisboa, Portugal, 2006.

[5] CacheLogic. Serving cached content. http://www.cachelogic.com/products/cachepliance.php.

[6] D. R. Choffnes and F. E. Bustamante. Taming the torrent: A practical approach to reducing cross-isp traffic in p2p systems. In *Proceedings of ACM SIGCOMM '08*, Seattle, WA, August 2008.

[7] N. Feamster, J. Borkenhagen, and J. Rexford. Guidelines for interdomain traffic engineering. *ACM SIGCOMM Computer Communications Review*, Oct. 2003.

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

[9] D. Goldenberg, L. Qiu, H. Xie, Y. R. Yang, and Y. Zhang. Optimizing cost and performance for multihoming. In *Proceedings of ACM SIGCOMM '04*, Portland, OR, Aug. 2004.

[10] 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.

[11] Internet2 Network NOC. http://noc.net.internet2.edu/i2network.

[12] W. Jiang, R. Zhang-Shen, J. Rexford, and M. Chiang. Cooperative content distribution and traffic engineering. In *Proceedings of ACM SIGMETRICS/Performance '09*, Seattle, WA, June 2009.

[13] A. Kuzmanovic and E. Knightly. TCP-LP: Low-priority service via end-point congestion control. *IEEE/ACM Transactions on Networking*, 14(4), Aug. 2006.

[14] Lightreading.com. P2P plagues service providers. http://www.lightreading.com/document.asp?doc_id=31767, Apr. 2003.

[15] 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.

[16] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. Inferring link weights using end-to-end measurements. In *Proc. of IMW*, Marseille, France, 2002.

[17] J. Ni, H. Xie, S. Tatikonda, and Y. R. Yang. Network routing tree topology inference from end-to-end measurements. In *Proceedings of IEEE INFOCOM '08*, Pheonix, AZ, Apr. 2008.

[18] A. Parker. The true picture of peer-to-peer filesharing. http://www.cachelogic.com, July 2004.

[19] S. Ren, L. Guo, and X. Zhang. ASAP: an as-aware peer-relay protocol for high quality voip. In *Proceedings of IEEE ICDCS '06*, Lisboa, Portugal, 2006.

[20] M. Roughan, M. Thorup, and Y. Zhang. Traffic engineering with estimated traffic matrices. In *Proceedings of the Internet Measurement Conference*, Miami, FL, Oct. 2003.

[21] O. Saleh and M. Hefeeda. Modeling and caching of peer-to-peer traffic. Technical Report TR 2006-11, Simon Fraser University, May 2006.

[22] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy. An analysis of internet content delivery systems. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI) '02*, Boston, MA, Dec. 2002.

[23] S. Seetharaman and M. Ammar. Characterizing and mitigating inter-domain policy violations in overlay routes. In *Proc of ICNP*, 2006.

[24] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with Rocketfuel. In *Proc. of SIGCOMM*, Pittsburgh, PA, Aug. 2002.

[25] A. Venkataramani, R. Kokku, and M. Dahlin. TCP Nice: A mechanism for background transfers. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI) '02*, Boston, MA, Dec. 2002.

[26] 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.

[27] H. Xie, A. Krishnamurthy, Y. R. Yang, and A. Silberschatz. P4p: Proactive provider participation for P2P. Technical Report YALEU/DCS/TR1377, Yale University, New Haven, CT, March 1 2007.

[28] H. Xie, Y. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4p: Provider portal for p2p applications. In *Proceedings of ACM SIGCOMM '08*, Seattle, WA, August 2008.

[29] H. Xie, Y. Yang, and A. Silberschatz. Towards isp-compliant, peer-friendly p2p design. In *Proceedings of IFIP Networking 2008*, Singapore, May 2008.

[30] M. Yang, H. Chen, B. Y. Zhao, Y. Dai, and Z. Zhang. Deployment of a large-scale peer-to-peer social network. In *Proceedings of First Workshop on Real Large Distributed Systems (WORLDS) '04*, 2004.

[31] ZDNet News. ISPs see costs of file sharing rise. http://news.zdnet.com/2100-9584_22-1009456.html, May 2003.