Network Optimization for DHT-based Applications

Yi Sun, Y.R. Yang, Xiaobing Zhang, Yang Guo, Jun Li, Kavé Salamatian

Network Optimization for DHT-based Applications

Yi Sun*, Y. Richard Yang†, Xiaobing Zhang‡, Yang Guo*, Jun Li*, Kave Salamatian§
*Institute of Computing Technology, Chinese Academy of Sciences, Beijing, P. R. China
†Yale University, New Haven, CT, USA
‡Shanghai Synacast Media Tech. CO. LTD (PPLive), Shanghai, P. R. China
§University of Savoie France, Paris, France

***Abstract*—P2P platforms have been criticized because of the heavy strain that some P2P services can inflict on costly inter-domain links of network operators. It is therefore necessary to develop network optimization schemes for controlling the load generated by P2P platforms on an operator network. Previous focus on network optimization has been mostly on centralized tracker-based systems. However, in recent years multiple DHT-based P2P networks are widely deployed due to their scalability and fault tolerance, and these networks have even been considered as platforms for commercial services. Thereby, finding network optimization for DHT-based P2P applications has potentially large practical impacts. In this paper, we present THash, a simple scheme to implement an effective distributed network optimization for DHT systems. THash is based on standard DHT put/get semantics and utilizes a triple hash method to guide the DHT clients sharing resources with peers in proper domains. We have implemented THash in a major P2P application (PPLive) by using the standard ALTO/P4P protocol as the network information source. We conducted realistic experiments over the network and observed that compared with Native DHT, THash only generated 45.5% and 35.7% of inter-PID and inter-AS traffic, and at the same time shortened the average downloading time by 13.8% to 22.1%.**

I. Introduction

P2P platforms have been frequently considered as commercial platforms for content distribution. But one major obstacle to the further development of P2P system is the inability to control the spread of the traffic and the utilization of network resources, in particular costly inter-domain link capacities. For example, the BBC iPlayer [1] stepped back on P2P distribution mode in late 2008 because of the criticisms from the ISPs that the application inflicted major traffic increases to their inter-domain links. This is therefore mandatory for a content distribution P2P platform to provide solutions for controlling the spread of the load they generate and to maintain the locality of traffic. We are calling this problem P2P network optimization. The aim of this paper is to propose a practical scheme to address the network optimization problem in DHT-based P2P content distribution platforms.

Up to now, the focus of network optimization has been mostly on tracker-based centralized P2P applications (e.g., [2–6]). However, in recent years, multiple-DHT based schemes have been increasingly used to retrieve information in large-scale P2P networks due to their inherent scalability and fault tolerance. Noteworthy examples are Kademlia [7] (used in BitTorrent, eMule, and Thunder), Chord [8], and CAN [9]; the DHT mode of Thunder contributes as high as 3% of the total traffic of China Telecom; Vuze, a major client for BitTorrent, uses a DHT mode to achieve network optimization to bypass the deployment difficulty of modifying a large number of heterogeneous tracker implementations. Nonetheless, implementing network optimization for DHT-based P2P is more challenging than schemes that for centralized tracker based system because of the decentralized control of these systems. But finding a viable solution for this type of platforms has a potentially large practical impacts.

A major potential downside of network optimization is reduced user performance. Network optimization can limit the scope of data exchanges, forbidding the user to download from other network regions that have high bandwidth. Therefore there is a fundamental trade-off between network optimization and users performance.

Dealing with this tradeoff is much easier in a tracker based system, as the centralized tracker will consider both global bandwidth matching and locality when computing the neighbors for a peer. But in generic DHT systems, the problem becomes more difficult. The reason is that a generic DHT system supports only "put" and "get" semantics; a put simply appends value to a list and a get is typically simply to get the top elements (if the list is long). Therefore, it is mandatory to add a clever manipulation scheme on top of DHT or to extend the semantic of DHT to enable data transformation needed for network and bandwidth optimization.

In this paper, we present THash, a simple scheme that implements effective network optimization for DHT systems. THash is based on standard DHT put/get semantics and builds on top of it using a simple, triple-hash technique to conduct distributed network optimization for DHT-based systems.

We completely implemented THash over PPLive[10] and used ALTO/P4P [5] as the network information source. We conducted realistic network experiments with more than 20000 real users and showed the benefits of THash. In particular, we showed that compared with Native DHT, THash only generated 45.5% and 35.7% of inter-PID and inter-AS traffic, and at the same time shortened the average downloading time by 13.8% to 22.1%.

The rest of the paper is organized as follows. Section II introduces some background knowledge. Section III describes the design of THash. Section IV presents the results of real network experiments showing the performance gains achieved

by THash. Section V shows the related work and Section VI presents our conclusions.

II. Overview

This section provides some background information which is useful to better understand the content of other part of the paper.

A. DHT Model

We consider a DHT-based network using standard DHT semantics. Specifically, the DHT network is used for publishing and querying resource information. Peers in the DHT network can play three roles:

Publishing Peer (PP): the peer who wants to publish information to announce its ownership of a specific resource.

Requesting Peer (RP): the peer who sends a request to search for a specific resource.

Indexing Peer (InP): the peer who maintains the publication information (publishing peer list) of a specific resource.

In a traditional DHT network, a PP computes a hash using the resource file identifier (file-id), and then invokes the put method of the DHT to put its publication information into the corresponding InPs; a RP apply the same hash function as the PP on a given file identifier and then invokes the get method of the DHT to fetch (k elements from) the list of peers owning the resource from the InPs.

A major drawback of this query approach is that the returned peer list does not consider network optimization. Thus, the RP may download from peers that are not network efficient; where for network optimization, it is desirable that the RP connects to close-by PPs.

B. ALTO/P4P as Network Information Source (NIS)

The importance of controlling the locality of traffic in P2P network has been resulted in the proposition of an Internet Draft at IETF to achieve Application Layer Transport Optimization (ALTO). The proposed approach named ALTO/P4P [5] is a simple and flexible framework that enables ISPs and application developers to cooperate in order to optimize application communications and to reduce resource consumption.

Specifically, a network provider (ISP) implementing the ALTO/P4P network information framework, deploys a server (say iTracker) to provide its network information relative to topology, congestion status, cost, and routing policies through the ALTO/P4P protocol. This information is called "*my-Internet view*". The "my-Internet view" of an ISP consists of two maps: the Network Map and the Cost Map. The Network Map divides an ISP network into multiple regions called PID domains and each PID domain can be identified by a PID domain number (PIDN). The Cost Map defines the cost, referred as p4p-distance, between each pair of PID domains.

Fig. 1 shows an example DHT network where peers are distributed in two different ISPs, each identified by a unique autonomous system number (ASN). Moreover, peers in the DHT system are distributed at different PID domains of each one of the two ISPs. Each peer can query, for example at the time it obtains its IP address, the iTracker to map its IP address to its PIDN and ASN, and obtain the two maps (Network and Cost Maps).

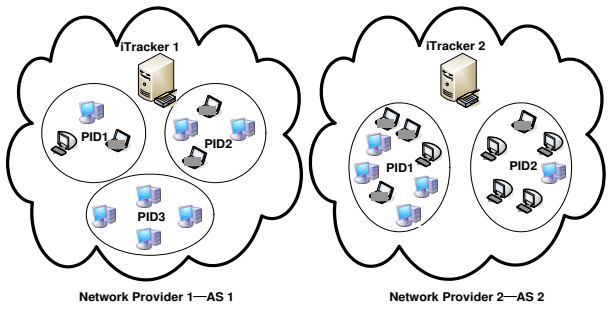

Fig. 1. An example DHT network where peers are distributed at two ISP networks.

C. PPLive

In order to evaluate our proposed scheme in a real environment we have implemented it into the PPLive platform [10]. PPLive is a leading online video service platform, founded in 2004 and offering more than 120 TV station live streaming and VoD of thousands of TV shows and programs. The PPLive company claims to have more than 200 million user installations and an active monthly user base of 104 million, i.e, a 43% penetration of Chinese internet users. The average viewing time per person per day over PPLive is more than 2.5 hours, the highest stickiness among all China websites. PPLive provides a hybrid CDN-P2P cloud platform and uses tracker based scheme as well as DHT for its content distribution.

III. THash Scheme

In this section we describe the THash scheme that implements the network optimization for DHT along with taking care of application performance.

A. Integrating Application Requirement and Network Information Using Peering Guidance Matrix (PGM)

The two network maps provided by the ISPs through ALTO/P4P are only from the network perspective. They do not consider application state and requirements. We need to integrate application state and requirements with them. For this purpose we define for each existing resource contents in a particular ISP a matrix called Peering Guidance Matrix (PGM). The PGM specifies the relative proportion of peers in each PID domain that should be used by a client in a given PID to download the content.

It is noteworthy that for fast downloading and in order to ensure system robustness, we should not restrict to choose peers only within a single domain and leave opportunities for clients to exchange data with peers outside.

Table I shows an example PGM for a specific resource for an ISP with 3 PID domains. Each row of the matrix encodes

TABLE I
AN EXAMPLE PGM

	PID1	PID2	PID3	Intra-AS Percentage
PID1	75%	10%	15%	90%
PID2	18%	70%	12%	85%
PID3	10%	10%	80%	90%

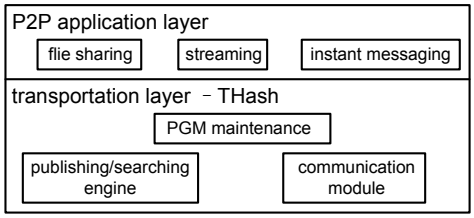

Fig. 2. System model of THash on DHT client.

the peer selection proportion for a single PID domain. For example a client at PID1, should choose 90% of its peers from inside the same ISP. The load balancing between the different PIDs of the AS are also given. For example, the client should select 75% of its peers in the AS from PID1, 10% from PID2 and 5% from PID3. Therefore, when selecting publishing peers for the resource transmission, a peer of PID1 should select up to *90%∗75%=67.5%* peers from its own PID, *90%∗10%=9%* peers from PID2 and *90%∗15%=13.5%* peers from PID3 and 10% peers from other AS.

The precise derivation of the PGM values will be described in section III-C.

B. Triple Hash Implementing PGM

In the classical DHT network, the resource publishing and lookup processes (DHT put/get semantics) are only based on the result of a hash of the resource file identifier (file-id). However in our scheme, we have to follow the indications of the PGM. This is achieved by the THash scheme we are defining in this section.

The THash scheme is based on a triple hash method that uses the standard DHT put/get semantics for publishing and searching resources. The triple hash method in THash adheres with the indications of PGM, and therefore implements the network optimization but taking care of application performance. In addition, THash avoids creating bottlenecks at NIS servers. The system model of THash is depicted in Fig. 2. In the forthcoming, we describe the details of the operations of the THash scheme for publishing and retrieving information.

1) Resource publishing in THash

When a PP wants to announce its ownership of a resource file in THash, it has to compute the following three hash values:

(a) Hash value on the file identifier.

$$Key1 = Hash(\textit{file-id}) \tag{1}$$

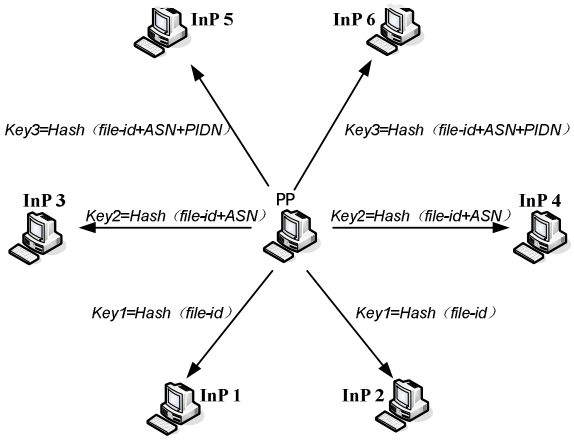

Fig. 3. Triple hash for resource publishing in THash.

(b) Hash value on the combination of the file identifier and the PP's AS domain number.

$$Key2 = Hash(\textit{file-id} + ASN) \tag{2}$$

(c) Hash value on the combination of the file identifier, the PP's AS domain number and the PP's PID domain number.

$$Key3 = Hash(\textit{file-id} + ASN + PIDN) \tag{3}$$

This is the three values that are going to be used by an RP to query the DHT to find the resources. Indexing Peers (InPs) also should store and manage these three keys. Each particular DHT network has its own method, mainly based on recursive searching to search for the InPs [7–9]. Once the InPs are found, the PP is added to the publishing peer lists, which are maintained by these InPs. As shown in the example in Fig. 3, the publication information for each resource is stored in three groups of InPs: one group for each hash key. For example InP1 and InP2 are the InPs for *Key1*, InP3 and InP4 for *Key2*, and InP5 and InP6 for *Key3*. It is easy to see that although THash scheme triggers three separate recursive searching for the InPs relative to each key, the complexity of the resource publishing scheme remains in the same order as in native DHT, $O(\log N)$, where N is the number of peers in the DHT network.

2) Resource searching in THash

The above triple hash method for resource publishing in THash enables us to control the P2P traffic over two layers: PID domain and AS domain layer. If a RP wants to download a file from the peers in the same PID domain, it can use its *Key3* to search the publication information in the DHT network. Similarly, if a RP wants to download within the same AS domain, it uses *Key2*. And if a RP has no limitation on the positions of the peers, it uses *Key1*. To extend this further, if a RP wants to download a file from a specific domain, it only needs to add the ASN and PIDN of this domain into the hash operation and search by the resulting key in the network accordingly. Therefore, a RP can easily follow the direction given in the PGM to download the resource of interest from its own PID, AS or outside and do the network optimization.

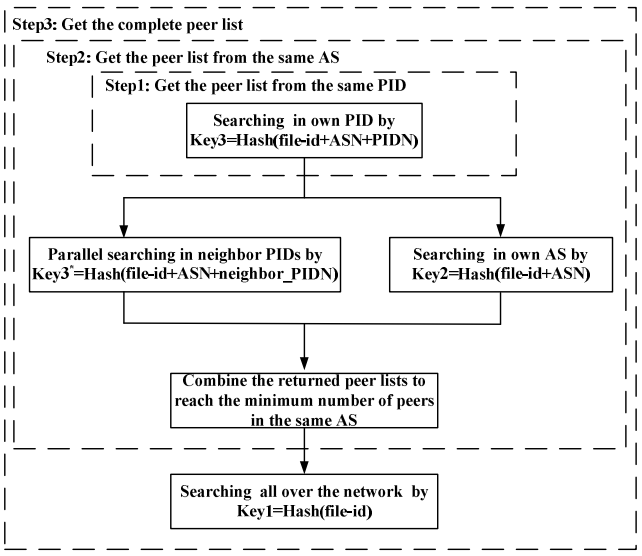

Fig. 4. Resource searching process for RP.

The peer selection proceeds in the following way: Assume that a RP needs *k* peers. The RP will receive these *k* peers in three stages. First, the RP calculates *Key3* and search it. Thus in this stage the peers in the same PID domain as the RP can be derived. Second, the RP launches a round of parallel searches with the other PID numbers given in the PGM and it own AS number by computing *Key3** with these values. These queries return some extra peers in other PIDs but the same AS with RP. Note that the PGM indicates the intra-AS peer percentage. If the above procedures do not meet this ratio, the RP will also launch a query with *Key2* and find some additional peers in the same AS. Finally, in the last stage, the RP launches a query with *Key1* that returns it the list of peers regardless of the AS and PID domains. How many peers are derived in each stage depends on the proportions specified in the PGM. Each InP for the same resource maintains the corresponding PGM, thus the RP only needs to tell the value of *k* (total number of peers) in its query message and InPs can calculate how many peers they should return according to the instructions in the PGM.

The above description shows that the complexity of the searching process in THash is $O(\log MN)$, where *N* is the number of peers in the DHT network and *M* is the number of PID domains of interest. According to the trace data by PPLive, the value of *M* remains negligible compared to *N* (usually 4–5 orders smaller than *N*) and thereby the complexity stays $O(\log N)$, the same as that in native DHT. The resource searching process of RP in THash is depicted in Fig. 4.

C. Derivation and Distribution of PGM

The preceding section presented the THash scheme. This was dependent on the availability of the PGM. In this section we will describe how to compute and distribute this value.

1) Basics of PGM

As illustrated in Section II.B, the p4p-distance in the Cost Map expresses the costs and distances between peers in different domains from the network point of view and can be used to capture a number of interesting network metrics, such as peak backbone utilization and preferred inter-domain links. Thus it can be made to contain important information that should be considered in network optimization and enable the derivation of the PGM that is used by nodes to modulate their connectivity. Let's describe how this distance can be derived.

Generally speaking, ISP can assign the p4p-distance in a variety of ways [6], for example deriving it from OSPF weights and BGP preferences or assigning it according to a financial cost. It this paper we consider the approach presented in [6] that uses the dual (shadow price) variables of optimization decomposition.

In this approach the ISP computes the p4p-distance using optimization decomposition by minimizing the maximum link utilization. This is the distance that is returned back in the Cost Map. As our approach is rather independent of the algorithm used to derive the p4p-distance we are not giving more details about that part here and refer the reader to [6].

Nonetheless, p4p-distances are from the ISP perspective. To compute the PGM, the application perspective should also be considered. In particular, the total number of Publishing Peers (PPs) in one domain should be considered. The more copies of the interested resource exist in one domain, the higher is the probability of the RP selecting peers in this domain. Let's suppose p_{ij} denoting the p4p-distance from PID *i* to PID *j* calculated by the ISP and n_j denotes the total number of the PPs for the interested resource in domain *j*. Then a simple method to compute the entry w_{ij} in the PGM is a weighted average given in Eq. (4), where coefficients c_{i1} and c_{i2} specify the relative weights for the two factors (p4p distance p_{ij} and the total number of the PPs n_j)

$$w_{ij} = c_{i1} \cdot \frac{\frac{1}{p_{ij}}}{\sum_j \frac{1}{p_{ij}}} + c_{i2} \cdot \frac{n_j}{\sum n_j} \tag{4}$$

Eq. (4) indicates that a direct relation of w_{ij} with the total number of the PPs in domain *j* and an inverse relationship with the p4p-distance from PID *i* to PID *j*. One can derive the content of the i^{th} row in the PGM by normalizing w_{ij} (Eq. (5)).

$$a_{ij} = \frac{w_{ij}}{\sum_j w_{ij}} \tag{5}$$

From Eq. (4) and (5), we can conclude that a RP in PID *i* will more likely connects to peers in the PIDs which have more copies of the resource and are near PID *i*.

The final remaining parameters to determine are weighting coefficients c_{i1} and c_{i2} in Eq. (4). These can be assigned using a series of different strategies. The network operator can decide to statically assign these values according to its experience; or the coefficients can be set dynamically according to the variable network conditions.

We propose to set these values using the following rationale.

1) The coefficients are weighting parameters and should satisfy $0 \leq c_{i1}, c_{i2} \leq 1$, $c_{i1} + c_{i2} = 1$.
2) If all the candidate PIDs have the same value in one attribute, this attribute is inefficient for differencing PIDs. Therefore the corresponding coefficient for this attribute should be set to 0.
3) If an attribute has large variation in its values and supplies more information it should be given a higher weight. Therefore the corresponding coefficient should be large.

To obtain coefficients that validate the above constraints we use the following approach. Let's assume there are altogether *k* PIDs in the same AS domain and each of them has the above two attributes: distance from other PIDs and number of PPs. We have therefore for each PID two vectors of length *k* that can be put in a *k×2* matrix. Take PID *i* as an example, then the matrix can be expressed as $B_i = (b_{mn})_{k\times 2}$, where b_{m1} denotes p_{im}, the p4p-distance from PID *i* to the PID *m* and b_{m2} denotes n_m, the number of PPs in PID *m*.

Now let's derive the distance between a vector *X* and the uniform vector *U* of dimension *k* using the Kullback-Leibner distance defined as:

$$\begin{aligned} D(X||U) &= \log k + \sum\nolimits_x p(x) \log p(x) \\ &= \log k(1 - H(X)) \end{aligned} \tag{6}$$

where *H(X)* is the entropy of *X*. If we assign weights to the two factors proportional to their Kullback-Leibner distances to uniform vector we will have

$$c_{i1} = \frac{1 - H_{i1}}{2 - H_{i1} - H_{i2}} \quad c_{i2} = \frac{1 - H_{i2}}{2 - H_{i1} - H_{i2}} \tag{7}$$

where H_{ij} is the entropy of the j^{th} column of matrix B_i relative to node *i*. These above defined coefficients will have the needed desirable features.

2) Computation and Distribution of PGM

As described above the PGM is derived using the network costs provided by ISP server and the numbers of PPs in the different domains of the network. However this last input is available only at the Indexing Peers (InP). Therefore, one way to compute the PGM is to ask the Indexing Peers to do it. However, this is not really suitable as InPs can be unreliable.

To deal with these issues, we rather propose to use an additional server named the Application Optimization Engine (AOE) to compute the PGM when distributing a resource. The AOE collects network information from the ISP servers as well as the number of PP from InPs and calculates the PGM. It thereafter distributes the PGM to InPs.

Another challenge is that the DHT network may have a large number of InPs. If a large number of InPs all interact with the AOE for the computation of PGMs, the AOE can become a bottleneck. Moreover, synchronizing the content of the matrix among the different InPs for the same resource will be difficult.

To avoid making the AOE a bottleneck, we strictly limit in THash the number of the InPs that can interact with the AOE. Recall that we had three types of InPs (InP for *Key1*, *Key2* and *Key3*). Particularly, *Key1* is the hash value of the file-id of the resource. InPs for *Key1* maintain the most complete publication information for all the DHT network. Therefore, we require only InPs for *Key1* to be able to communicate with the AOE. If an InP for *Key1* finds that the distribution information of a specific resource varies drastically, it would initiate a connection with the AOE and upload the current distribution information of the resource (how many PPs of the resource exist currently in each domain). The AOE computes the new PGM according to the current network information and resource distribution information. Then the AOE only replies with the updated matrix to the InPs for *Key1*. In addition, while the AOE has updated the PGM due to the significant variation of the ISP network state, it can also actively initiate the connections with *Key1*-level InPs. Moreover, we assume that the PGM has a validity lifetime. This means that *Key1*-level InPs should query periodically the AOE to ensure PGMs' freshness.

While assessing the variation of the network information (say p4p-distance) is relatively easy using the information provided by the NIS, deciding on the variation of the resource distribution is more subtile. Several approaches can be used for this purpose. For example using correlation coefficient [11] or even mutual information [12]. However, in this work we have used a simple technique. Let's suppose the two vectors $X=\{x_1, x_2, \ldots x_n\}$ and $Y=\{y_1, y_2, \ldots y_n\}$ respectively denote the old and new numbers of PPs in different domains in the network. We first normalize these values by

$$x_i^* = \frac{x_i}{\sum_{j=1}^{n} x_j} \qquad y_i^* = \frac{y_i}{\sum_{j=1}^{n} y_j} \tag{8}$$

and then define the distance between the two vectors as

$$distance(X, Y) = \max_{i=1\ldots n} |x_i^* - y_i^*| \tag{9}$$

Whenever the above defined distance exceeds a threshold (eg. *distance(X,Y)* $> 5\%$), the PGM should be updated and diffused to InPs.

The last remaining action is defining how *Key1*-level InPs that have got the latest PGM, can transmit the matrix to other InPs managing the same resource. Basically, there are two ways.

The first approach is to ask *Key1*-level InPs that receive the matrix directly from AOE to share it with other InPs managing the publication information of the same resource at *Key2* and *Key3*-level. For this purpose, the *Key1*-level InPs derive the corresponding *Key2* and *Key3* by Eq. (2) and (3) for all available PID and AS domains. Then with these keys, the *Key1*-level InPs can launch searching requests in the DHT

network to find the relevant *Key2* and *Key3*-level InPs for these keys and transfer the new matrix to them.

The second approach consists of asking each InP to share the matrix only with InPs maintaining the same key value. Since these InPs have anyway to exchange periodically with their peer list, we can take benefit from these opportunities to exchange the PGMs simultaneously.

IV. Performance Evaluation

In other to validate our proposed scheme and to assess its performance in real operation we have deployed THash with the help of 3 major Chinese ISPs (China Telecom, China Unicom, CERNET) and over a real P2P network (PPLive). We present in this section the results of this deployment.

A. Experiment Deployment

As stated above, we have conducted experimental deployment inside the P2P network of a major actor of P2P video streaming in China, PPLive. The P2P network was studied in the network of the three major Chinese ISPs (China Telecom, China Unicom, CERNET). These three operators have implemented an ALTO/P4P service that is accessible from an ALTO/P4P server based at Yale University. The nodes in the system, which did not belong to any of the three ISP networks, were all categorized into the "other networks" category while computing the PGM.

We selected the Kademlia protocol for the DHT implementation. The original implementation of Kademlia is referred as Native DHT as it is used for comparison here. We added THash approach as described into Kademlia. We have also implemented THash and Native DHT schemes as plugins to the PPLive clients. PPLive released new version of their software to support our experiment. All users who had updated their clients joined our experiment. According to our statistic data, more than 20,000 nodes joined our experiment platform.

The experiment lasted for 6 hours from 15:30 to 21:30 on May 26th, 2011. We categorized the nodes in the DHT network into two groups. One group consisted of nodes with odd identifier numbers (odd Node ID) and were using THash scheme for resource publication and searching. The other group consisted of nodes with even identifier numbers (even Node ID) and used Native DHT scheme for resource publication and searching.

PPLive assigned for our experiments five test channels. Five separate files were distributed on these channels in the DHT network. The sizes of these files were respectively 413MB, 210MB, 226MB, 529MB and 222MB. Users in these five channels used DHT approaches, either Native DHT or THash, for resource sharing without the help of PPLive trackers.

We deployed a server for the computation of PGMs for the five resource files. The computation details are described in Section III.C, considering not only the p4p-distance but also the distribution of the resources. We obtained the Cost Map from the ALTO/P4P server at Yale University [13] and obtained the resource distribution information from the *Key1* Indexing Peers in the network.

To collect performance statistics we also deployed a log server. Every node in the DHT network maintains a simple log and a complete log. The simple log records some basic data of the node such as the publishing and searching delay, the replied peer list for a resource and the amount of traffic exchanged with each peer. The complete log keeps more detailed information of the node such as debugging logs. During our test, we required every node to submit its simple log to our log server every 15 minutes. In this manner we were able to have the complete information of the resource sharing by peers in the DHT network.

B. Performance Metrics

The following performance metrics were considered:

- *Publishing delay:* The time interval between a peer sending a publication request for a resource and receiving the confirmation from the corresponding Indexing Peers (InPs). Usually, a Publishing Peer records its publication information on a series of InPs.
- *Searching delay:* The time interval between a peer sending a query for a resource and receiving the peer list from the corresponding InPs. Generally, the searching process is recursive, and for the THash parallel queries are run.
- *Intra-domain peer ratio:* The ratio of the number of intra-domain peers to the total number of the peers in the peer list. More specifically, this metric can either be intra-PID peer ratio or intra-AS peer ratio. For network efficiency consideration, we should increase the value of this metric.
- *Inter-domain traffic:* This metric, measuring the total P2P traffic traversing different domains for transmitting the five resources in our test, is another metric of network efficiency and greatly influences the scalability of the system. More specifically, this metric can either be inter-PID traffic or inter-AS traffic.
- *Downloading performance speedup:* This metric measures the downloading performance enhancement by using our THash scheme. It is defined as the ratio of the average downloading time of THash to the average downloading time of Native DHT for a specific resource.
- *Peer reselection ratio:* This metric measures the stability of the system and is defined as the ratio of the number of peers updated to the total number of transmission peers in each periodic updating interval.
- *PGM update overhead:* We use PGM update frequency to estimate the overhead for PGM updating.

C. Experiment Results

Fig. 5 depicts the variation of the average publishing delay for THash and Native DHT calculated every 1 hour. As can be seen the publishing delay difference between the two schemes is not very large (the relative ratio the publishing delay of the two schemes varies between 88%$\sim$102%). The overall average

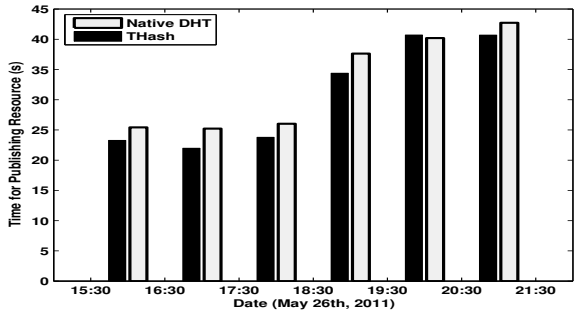

Fig. 5. Average publishing delay.

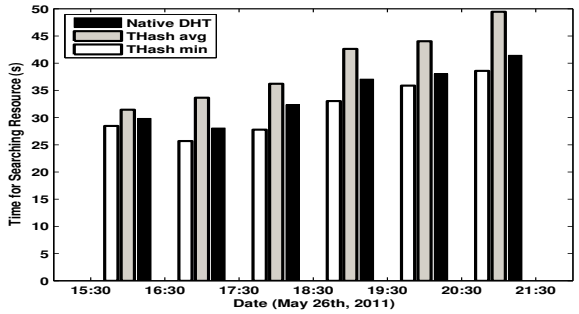

Fig. 6. Average searching delay.

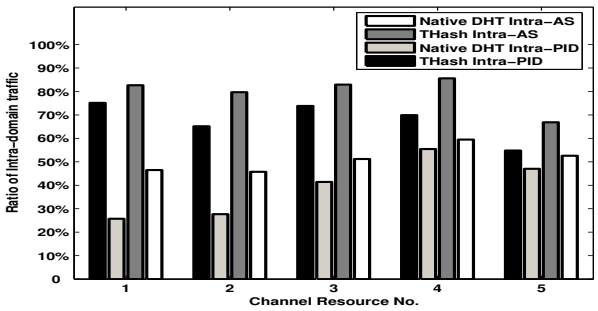

Fig. 7. Intra-domain peer ratio.

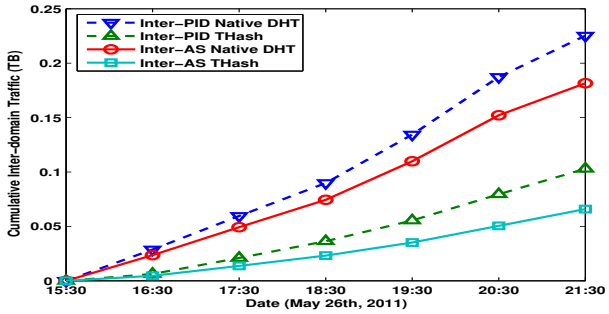

Fig. 8. Inter-domain traffic.

delay for THash was 30.9 sec where for Native DHT it is 31.8 sec. Therefore, we can conclude that THash does not have a negative impact on this important DHT network performance metric. The small differences in different hours are related to load variations that affect the same way THash and Native DHT.

Fig. 6 plots the variation of the average searching delay for THash and Native DHT. We observe that the use of THash causes an increase in the average delay for searching resources. The overall average searching delay for THash was respectively 39.6 sec and for Native DHT 34.5 sec. The reason for the increase is that in THash a Requesting Peer may send a series of parallel queries for searching in different domains simultaneously and this increases the total number of resource requests in the DHT network prolonging message queuing and processing delays. But fortunately, in THash we do not need to wait for all the reply messages arriving before we can initialize the session. Once the first reply message arrives, the RP can initiate the data transmission immediately with the peers contained in that reply message. In Fig. 6, the bars in white indicate the delay of the first reply for THash, and the average value of this metric is 31.6 sec, even lower than the average delay for Native DHT. Overall, the ratio of the searching delay of THash average to Native DHT average varies from 105%∼119% and the ratio of the searching delay of THash minimal to Native DHT average varies from 83%∼94%. The two ratios are stable over time.

Fig. 7 shows the intra-domain peer ratio averaged for each of the five resources for THash and Native DHT. We can clearly see that by introducing the network optimization in THash we remarkably limit inter-AS and Inter-PID traffic. In average the intra-PID and intra-AS peer ratios of THash are 36.3% and 34.6% higher than those of Native DHT. We have observed that in every 1 hour time interval the Native DHT scheme generated in average respectively 2.2 and 2.8 times of inter-PID and inter-AS traffic compared to THash scheme. Over the six hours test and only for these five resources, THash saved respectively 0.122 TB inter-PID traffic and 0.116 TB inter-AS traffic, see Fig. 8.

Fig. 9 compares the downloading performance speedup of THash relative to Native DHT. What is noteworthy is that different resources have different distributions in the DHT network. Therefore the PGMs for these resources are different and that results in diverse sharing peer behaviors in THash scheme. Fig. 7 showed that for Resource 1, 2, 3, and 4 RPs using THash are more likely to share resources with peers within the same domain, while for Resource 5 RPs connect to more peers outside. We see the effect of this fact in Fig. 9, where downloading performance enhanced by THash for resource 1, 2, 3 and 4 are more obvious than that of 5. However, as seen from Fig. 9, even for Resource 5, the average downloading time with THash is only 86.2% of that of Native DHT. Thereby, THash significantly improves the downloading performance of the applications.

In order to guarantee the quality of service, PPLive client updates its peers periodically (peer reselection process); new peers replace bad performance peers with poor uploading capacities. We have depicted in Fig. 10, the ratio of peer reselection in Native DHT to that in THash. It can be seen that for most of the time THash scheme performs better in this metric. Averaging the results of all the intervals, we can observe that Native DHT generated 117% peer reselections of that in

THash. This can be explained by the fact that peer selection in THash considers both link utilization in p4p-distance and load balancing in resource distribution information, thus the uploading performances of these peers are not badly impacted by the variations of the network conditions, resulting in higher stability and efficiency for systems.

PGM is a new concept introduced in this paper. Thus, the cost of updating and distributing PGM is a special overhead for THash scheme. Since the size of the PGM is fixed, this overhead can be estimated by looking at the PGM updating frequency. The PGM updating was triggered using the distance described by Eq. (9) and the threshold of the distance was respectively set to 0.05 and 0.1. In addition, the validity lifetime of the PGM was set to 10 min. We show in Fig. 11 the hourly PGM updating frequency. Note that the data in Fig. 11 is the sum of updating numbers for all the five resources. This figure shows that a stricter threshold (the black bars in Fig. 11 with the threshold set to 0.05) incurs more updates. We can observe that the number of PGM update decreases sharply, from more than 400 in the first hour to about 50 in the last hour. This is caused by the fact that initially there are not many PPs in each domain. The distribution of resources is therefore very sensitive to the dynamics in the P2P system. However with the experiment going on, sufficient copies for the resources appear in each domain. Therefore, a single peer's coming/leaving has little impact on the entire distribution of the resources and the resulting PGM becomes stable. Another interesting outcome of Fig. 11 is that the frequency of the PGM updating is not related to the intensity of traffic in the system. For example, the PGM updating frequency remained very low even during the period 20:30∼21:30 when the system was heavily loaded with the highest average resource publishing and searching delay. Moreover, as the average size of a PGM in our test is only 3.89 KBytes, when the system enters into a stable state, the overhead of updating and distributing the PGM is negligible compared with the data traffic transmission in the system (several TBytes of data).

D. Summary of Results

The experiment results illustrated from Fig. 5 to Fig. 11, enable us to state that: compared with Native DHT, THash scheme is able to reduce the Inter-AS and Inter-PID traffic (the most important performance metrics from ISP viewpoint) by a factor larger than two (Fig. 8) and ensure that more than 80% of traffic remain in the AS (Fig. 7). This happens surprisingly even with a decrease in downloading delay (at least 13.8%) and better downloading performance (Fig. 9). As these parameters are the major metrics of the efficiency and scalability of the P2P network, the THash scheme does improve the efficiency and scalability of the P2P network. In addition, THash also brings an improvement in peer reselection ratio (Fig. 10) which has a direct impact on the stability of the P2P system.

Indeed this comes with some costs. We need now to update and transmit PGMs and that incurs an overhead that has been shown to be negligible compared with the data traffic in the system (Fig. 11). Another cost of the scheme is relative to the increase in average searching delays. This is caused by the parallel searches that are needed. However, as the sessions can be initiated immediately after the first reply of these parallel search processes, the session startup time can even be reduced (Fig. 6).

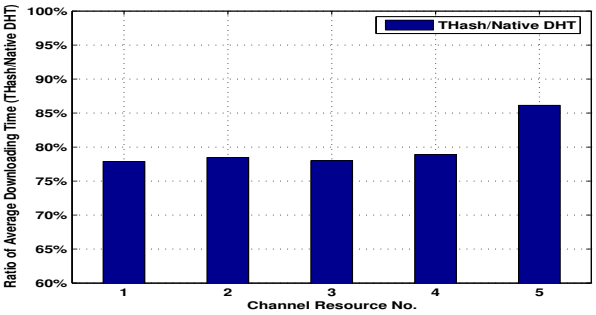

Fig. 9. Downloading performance speedup.

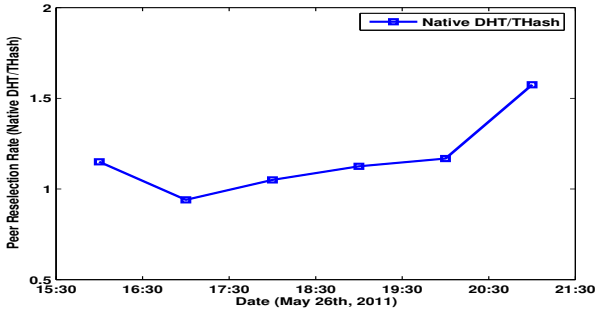

Fig. 10. Peer reselection rate (Native DHT/THash).

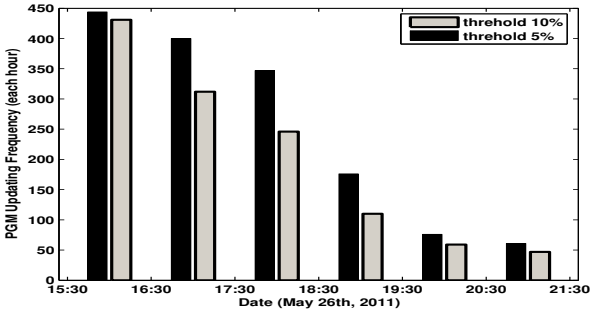

Fig. 11. PGM updating frequency (times in each hour).

V. Related Work

ISP Approaches: The traditional ISP approaches for network optimization include P2P caching and P2P shaping. References [14–16] proposed to deploy P2P caching devices to cut down the bandwidth occupation by P2P applications. Caching devices are often deployed at the edge of the ISP domains to provide service for the nearby peers. Combined with the redirection technology, resource requests from the peers can be redirected to the caches in the same domain. Thus, P2P caching can significantly reduce inter-domain P2P traffic and enhance downloading performance of the peers. However, P2P caches need to be designed for specific applications and utilize

the appropriate protocols, which limit their generality and applicability to proprietary protocols. In addition, ISPs may not want to bear the costs of caches. Furthermore, caching contents may lead to legal liability.

Another widely used ISP approach [17, 18] is to use traffic shaping to reduce the bandwidth cost of P2P applications. Traffic shaping devices can identify P2P traffic through DPI (Deep Packet Inspection) or DFI (Deep Flow Inspection) technologies and then impose additional delay on the transmission of this traffic in order to prevent the network from being overloaded. The disadvantage of the P2P traffic shaping is that the end-to-end performance of the P2P applications may be seriously deteriorated [18]. In addition, the difficulty of identification of P2P traffic will significantly increase if the P2P applications use encryption and dynamic ports.

P2P Approaches: P2P application providers also devoted themselves to reducing P2P traffic in the network and improving downloading performance of their applications. Most of these approaches focus on centralized P2P applications. The basic idea of these approaches [3, 4] is that the AppTracker (centralized server) selects the nearby nodes to share the resource with the requesting node. This traffic localization can indeed improve network efficiency. However, the main disadvantage of these P2P application provider approaches is that they all rely on the fact that the applications can probe the network and infer various types of network information such as topology, congestion status, cost, and policies. Reverse engineering of such information by P2P application providers is challenging and the accuracy of their results is questionable.

Ono [3] and Vuze [19] also use DHT based component. Specifically, Ono uses CDN redirection behaviors to approximate network information, and Vuze DHT is based on a network coordinate system. But such information sources are mostly driven by latency or CDN server status, without consideration for other network information such as congestion status, cost and routing policy. In contrast THash is a generic network optimization scheme for DHT systems, that uses ALTO/P4P standardized as the standard network information source. Also, THash utilizes Peering Guidance Matrix as a generic mechanism to avoid considering network locality only.

Similar Experiments: It is noteworthy that in 2010 China Telecom launched a large-scale experiment [20] similar to our experiment using ALTO/P4P but with Thunder P2P application. The main difference between their test and ours is that they estimated the tracker-based application performance and we focused on DHT system. In addition, the PGM introduced in this paper not only considers the p4p-distance from ISP perspective but also resource distribution from application perspective. Therefore, the locality of the traffic and the user downloading speed can both be improved in our THash.

VI. Conclusion

In this paper, we presented THash, a simple scheme to conduct effective network optimization for DHT systems. THash is based on standard DHT put/get semantics to conduct distributed network optimization for DHT-based systems and utilizes a triple hash method to guide the DHT clients sharing resources with peers in proper domains. Our large-scale experiments show that THash can be as a solution for enabling DHT-based P2P platform to have a better control of the load they generate over ISP network, opening the way for a larger deployment of these systems for content diffusion.

Acknowledgment

This work is supported by National Basic Research Program of China (2012CB315802, 2011CB302702) and the Natural Science Fundation of China (61003266, 61100177). The authors wish to acknowledge R. Alimi at Yale University and G. Yang at Institute of Computing Technology for their contributions to the design and implementation of THash. The authors also wish to acknowledge Prof. L. Mathy at Lancaster University UK for his comments on the paper.

References

[1] iPlayer. http://www.bbc.co.uk/iplayer/.
[2] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, and A. Krishnamurthy. iPlane: An Information Plane for Distributed Services. In Proc. of OSDI2006, Seattle, Washington, Nov. 6–8, 2006.
[3] D. R. Choffnes and F. E. Bustamante. Taming the Torrent: A Practical Approach to Reducing Cross-isp Traffic in Peer-to-peer Systems. In Proc. of CCR, Vol.38 No.4 2008.
[4] S. Tang, H. Wang, and P. V. Mieghem. The Effect of Peer Selection with Hopcount or Delay Constraint on Peer-to-Peer Networking. Springer-Lecture Notes in Computer Science Vol.4982 2008.
[5] R. Alimi, R. Penno, and Y. Yang. ALTO Protocol, IETF draft draft-ietf-alto-protocol-09.txt. Jun. 2011.
[6] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu and A. Silberschatz. P4P: Provider Portal for Application. In Proc. of ACM SIGCOMM, Seattle, Washington, USA, Aug 17–22, 2008.
[7] P. Maymounkov and D. Mazieres. Kademlia: A Peer-to-peer Information System Based on the XOR Metric. InLNCS2429, pp. 53–65, 2002.
[8] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In Proc. of ACM SIGCOMM, San Diego, California, Aug. 27–31, 2001.
[9] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker. A Scalable Content-Addressable Network. In Proc. of ACM SIGCOMM, San Diego, California, Aug. 27–31, 2001.
[10] Y. Huang, T. Fu, D. Chiu, J. Liu, and C. Huang. Challenges, Design and Analysis of a Large-scale P2P-VoD System. In Proc. of ACM SIGCOMM, Seattle, Washington, USA, Aug 17–22, 2008.
[11] Correlation. http://www.mega.nu/ampp/rummel/uc.htm.
[12] T. M. Cover and J. A. Thomas. Elements of Information Theory (2nd ed.). Wiley, New York, pp. 19–30, 2006.
[13] P4P maps. http://p4p.cs.yale.edu/files/doc/latest/p4p-common-cpp/html.
[14] M. Hefeeda, B. Noorizadeh. On the Benefits of Cooperative Proxy Caching for Peer-to-Peer Traffic. In IEEE transactions on Parallel and Distributed Systems, Vol.21 No.7 pp.988–1010, 2010.
[15] J. Dai, B. Li, F. Liu, B. Li, and H. Jin. On the Efficiency of Collaborative Caching in ISP-aware P2P Networks. In Proc. of INFOCOM, Shanghai, China, Apr. 10–15, 2011.
[16] E. Rosensweig, J. Kurose, and D. Towsley. Approximate Models for General Cache Networks. In Proc. of INFOCOM, San Diego, CA, USA, Mar. 15–19, 2010.
[17] C. Wang, N. Wang, M. Howarth, and G. Pavlou. A Dynamic Peer-to-Peer Traffic Limiting Policy for ISP Networks. In Proc. of NOMS. OSaka, Japan, Apr. 19–23, 2010.
[18] M. Marcon, M. Dischinger, K. P. Gummadi and A. Vahdat. The Local and Global Effects of Traffic Shaping in the Internet. In Proc. of ACM SIGCOMM, Seattle, Washington, Aug. 17–22, 2008.
[19] Vuze. www.vuze.com.
[20] K. Lee and G. Jian. ALTO and DECADE service trial within China Telecom. IETF draft draft-lee-alto-chinatelecom-trial-02.txt. Apr. 2011.