LIP: A LIfetime and Popularity Based Ranking Approach to Filter out Fake Files in P2P File Sharing Systems*

Qinyuan Feng, Yafei Dai
(Department of Computer Science and Technology, Peking University, Beijing, China)
{fqy, dyf}@net.pku.edu.cn

Abstract

P2P file sharing systems often use incentive policies to encourage sharing. With the decrease of free riders, the amount of cheating behaviors has increased. Some users rename a common file with a popular name to attract the downloads of other users in order to gain unfair advantages from incentive policies. We call the renamed file a fake file. While techniques have been proposed to combat fake files, an effective approach to filter out fake files in existing systems is lacking, especially before a real file comes out. In this paper, we design two detectors to identify fake files by mining historical logs and find that fake files are indeed pervasive. We introduce a metric called lifetime, which is a file's average retention time in users' computers, and show that it can be used to distinguish between real and fake files. We then propose a lifetime and popularity based ranking approach to filter out fake files. Experiments are designed with the real and fake files collected by the two detectors, and the results show that this approach can reduce the download volume of fake files both before and after a real file comes out.

1. Introduction

P2P file sharing systems often use incentive policies to encourage sharing. Some incentive policies [1] are based on points where peers are rewarded points for uploading and spend points for successful downloading, and use service differentiation to reward and punish users for their behaviors. They give downloading preference to users with higher scores and sometimes apply a bandwidth quota to the downloads of users with lower scores.

With these incentive policies, the amount of free riders decreases, however the amount of cheating behaviors has increased. Some users rename a common file with a popular name to attract the downloads of other users in order to gain unfair advantages from the incentive policies, and we call the renamed file a fake file.

*Supported by NSFC(60673183) and Doctoral Funding of MOE(20060001044)

We design two detectors to identify fake files by mining historical logs and find that fake files are indeed pervasive. To make things worse, some users even publish fake files before a real file comes out.

Two kinds of reputation mechanisms have been proposed to resolve this problem. One evaluates a user's reputation from his behaviors [2], but a user's reputation is sometimes different from a file's reputation; whereas the other evaluates a file's reputation directly [3], it can reflect the file's essential attribute.

Users tend to retain a real file longer and delete a fake file more quickly. From this phenomenon, we introduce a metric called lifetime, which is a file's average retention time in users' computers, to analyze the file's quality. We show that it can be used to distinguish between real and fake files. Based on it, we propose a lifetime and popularity based ranking approach to filter out fake files.

Experiments are designed with the real and fake files collected by the two detectors, and the results show that this approach can reduce the download volume of fake files both before and after a real file comes out.

The road map of this paper is as follows. In Section 2, we cover the related works. We design two detectors and analyze the time characteristics of files in Section 3. We then propose a lifetime and popularity based ranking approach in Section 4 and design an experiment to evaluate this approach in Section 5. We draw a brief conclusion and identify some ways to expand in future research in Section 6.

2. Related Works

In pollution measurement aspect, J. Liang et al. [4] analyzed the severe pollution which is lunched by some companies to protect copyrights in P2P file sharing systems; We analyze the severe pollution which is lunched by some users to gain unfair advantages from incentive policies in P2P file sharing systems, and our approach can also be used in their circumstances; U. Lee et al. [5] analyzed the time interval between download and the quality checking; We find the same situation in our work; D. Dumitriu et al. [6] concluded that a user's behaviors such as willing to share and removing pollution quickly have a great

impact on the effect of file targeted attacks; Our work improves this results by quantitative analysis of users' behaviors with lifetime; N. Christin et al. [7] analyzed the differences between pollution and poisoning and their respective impact on content availability in P2P file sharing networks, they define a deliberate injection of decoys to reduce the availability of targeted files as poisoning and a accidental injection of unusable files as pollution; Our work finds that a user's deliberate behavior can also come from cheating incentive policies; J. Liang et al. [8] analyzed index poison, and our work is a complement of theirs.

In pollution identification aspect, D. Dumitriu et al. [6] discussed random algorithm; J. Liang et al. [8] recommended distributed blacklist; S. Kamvar et al. [2] proposed EigenTrust algorithm; K. Walsh et al. [3] proposed an object reputation system; J. Liang et al. [4] proposed an approach depending on whether a file is decodable and the warp of its duration to identify pollution automatically. But some of these approaches require users to participate actively; Some can't identify fake files before a real file comes out; Some need to download all or part of a file; Some need a lot of system resource; Some can only identify some types of files. Our approach calculates the retention times of files and filters out fake files automatically, so it is good at bypassing the limitations above.

3. Analysis of Fake Files

3.1. Term Specification

- **Title:** the common description of a particular content. Such as "Movie: The Matrix" and "Music: Yesterday Once More". We use T(Title) to express it.
- **File:** the sharing object in P2P file sharing system. We use F(File) to express it. A file is composed by two parts:
  - **Name:** the name of a file and it should belong to a title, we use N(Name) to express it.
  - **Content hash:** the hash value which is generated by digesting the content of a file with a hash function, it is independent of the name. We use H(Hash) to express it.
- **Fake file:** a file whose name does not match its content hash.
- **File Owner:** a user who has ever owned the file.

Figure 1 gives an explanation of these specifications. In figure A, U1 owns real files F1 and F2, U2 owns real file F1, and N1 belongs to T1 whereas N2 belongs to T2; in figure B, U2 changes F1's name to N3 which belongs to T2 and creates fake file F3; in figure C, U3 downloads F2 from U1 and U4 downloads F3 from U2. We can get the following conclusions: T1 has real file F1 whose owners are U1 and U2; T2 has real file F2 whose owners are U1 and U3; T2 has fake file F3 whose owners are U2 and U4; Content hash H1 corresponds to two names which are N1 and N3; Content hash H2 corresponds to one name which is N2.

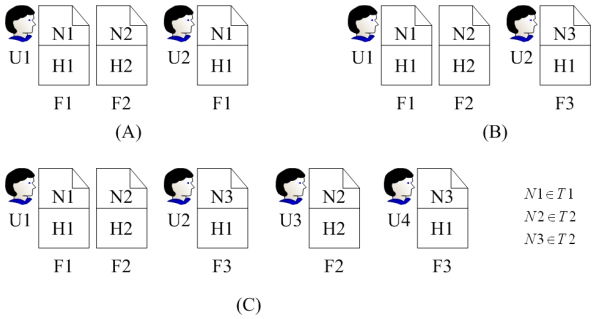

Figure 1: Specification sketch

3.2. Detection of Fake Files

Our historical logs are collected from Maze [9] which is a large deployed P2P file sharing system with more than 2 million registered users and more than 10,000 users online at any given time. A log server is used to record every downloading action and each log contains uploading peer-id, downloading peer-id, global time, file's content hash, and file's name. The logs from Oct 11, 2005 to Aug 11, 2006 are selected. Though our analysis is based on Maze, it is similar to many other P2P file sharing systems, so the approach and conclusion are universal.

We choose five representative popular titles and use T1, T2, T3, T4, and T5 to express them.

From users' feedbacks in Maze forum, we know there are some users who rename a common file with a popular name to attract the downloads of other users, so we get the hint to detect fake files from the change of names and the first publication time of a file.

When a user creates a fake file by renaming a file's name, it causes the file's content hash to correspond to at least two names. One belongs to the original title and the other belongs to the popular title, and the former one should appear earlier than the latter one. Furthermore, all the files of a title which are published before the title's real file comes out are sure to be fake files.

So, two detectors are designed to detect fakes files of a title.

- **Detector 1:** for each file of a title, the detector collects all the names that correspond to the file's content hash, if there exists a name which belongs to another title and appears before this file's first publication time, the detector will identify this file as fake.
- **Detector 2:** for a title, the detector gains the real

Table 1: Identification results

	Real file	Detector 1 only	Detector 2 only	Detector 1&2
T1	827(51.5%)	53(3.3%)	432(26.9%)	294(18.3%)
T2	409(83.3%)	12(2.4%)	21(4.3%)	49(10.0%)
T3	291(91.5%)	8(2.5%)	2(0.6%)	17(5.3%)
T4	411(68.3%)	42(7.0%)	49(8.1%)	100(16.6%)
T5	151(74.4%)	52(25.6%)	0(0.0%)	0(0.0%)

file's first publication time from Divx [10] and identifies the files published before this time as fake.

When we are identifying a title, we first collect all the files that belong to the title, then use the two detectors to identify them and mark them as real file(not identified by any of the detectors), detector 1 only(only identified by detector 1), detector 2 only(only identified by detector 2), detector 1&2(identified by both of the detectors). Table 1 shows the results with the pecentage of files belongs to each category. The five titles all have their own characteristics: T1 is the most popular title and it's fake files are the most pervasive; T2 has low-grade fake files; T3 has only published for a short time and it has the lowest grade of fake files; T4 has middle-grade fake files; T5's first publication time is earlier than the logs' collecting time and it represents the titles with incomplete logs.

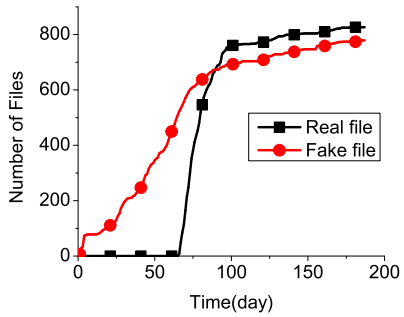

Figure 2: The change of cumulative numbers with time

Figure 2 shows the change of T1's real and fake files' cumulative numbers with time. The x-axis is the time, the y-axis is the cumulative numbers of real and fake files which have ever appeared (The other 4 titles' figures are similar to this one). It can be deduced from the figure that fake files can appear both before and after a real file comes out. So the approach to filter out fake files should have an effect in both conditions.

From the analysis above, we know that the two simple but efficient detectors can identify a lot of fake files, but there are still some questions: Why is it better to serve a fake file than the real one? Why not download a popular file and serve the real one? Doesn't this give the same benefits as serving a fake file? Why can't users use a completely new file as their fake files? The answer is same to all the questions: because it is the most simple way and it does work. What you should do is renaming a file, and especially if you rename it as a popular movie which has not appeared, all the users who want to download this movie will find you. In fact, we have tried some other ways to produce fake files, but this only makes things more complex and gains no more befinits.

Even so, we should recognize that the two detectors are not feasible in real systems, we need prepare a lot to analyze a title, this is the reason that we only choose five titles. Even if they are deployed, the users can also cheat the two detectors easily. But they collect the real and fake files from logs for our analysis. With the analysis below, a lifetime and popularity based ranking approach will be designed to filter out fake files automatically. Experiments are also designed with the the real and fake files collected by the two detectors to evaluate the effect of the approach.

3.3. Time Characteristic Analysis

Users tend to retain a real file longer and delete a fake file more quickly. From this phenomenon, we introduce a metric called lifetime, which is a file's average retention time in users' computers, to analyze the file's quality.

Definition of Lifetime and Popularity

If file F has n owners, and each owner's retention time of F is $t_i (1 <= i <= n)$, then define $L_F = \frac{\sum_{i=1}^{n} t_i}{n}$ as file F's lifetime and n as file F's popularity.

In order to calculate a file's lifetime, we check file and user's appearance time in the logs. For each file, its owners are gathered at first. Then each owner's first and last appearance times with this file are collected, the difference between the two times is treated as this owner's retention time of this file. For example, if U1 downloads F1 at 10:30 and uploads F1 at 14:00 and 16:30, we will calculate U1's retention time of F1 as $(16.5 - 10.5) \times 60 \times 60 = 21600s$. Some owners will close their client softwares after downloading a file or move the file out of the sharing directory, so the retention time calculated by this method will be smaller than the real retention time, but the warps are similar to all the files and we are mainly comparing lifetimes of real and fake files, it is therefore relatively reliable.

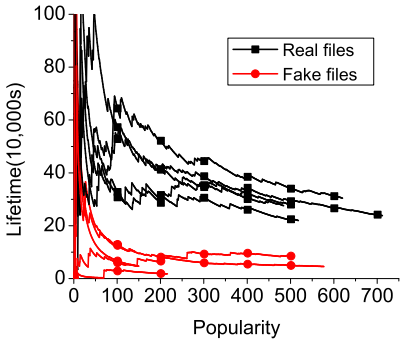

Figure 3: The change of lifetimes with popularity

Differentiation and Astringency of Lifetime

Figure 3 shows five real and five fake files belonging to five titles separately and the change of their lifetimes while their popularities increase. We can get the following conclusions. When the popularity is small, the lifetime fluctuates greatly. When the popularity reaches 100, the lifetime of a fake file converges below 100,000s, so we can distinguish fake files from real files then. Because the lifetime of a file with small popularity does not converge, we only analyze the files whose popularities are larger than 20 below.

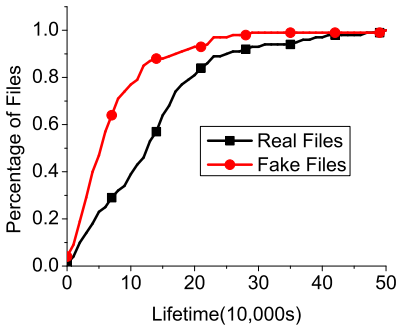

Figure 4: CDF of real and fake files' lifetimes

Distribution of Lifetime

Figure 4 shows T1's CDF of real and fake files' lifetimes. The x-axis is the lifetime, the y-axis is the percentage of real and fake files whose lifetimes are within x (The other 4 titles' figures are similar to this one). We can conclude from the figure that real files' lifetimes are holistically higher than fake files', which means compared to fake files, there are more real files with higher lifetimes.

4. Lifetime and Popularity Based Ranking Approach

We will describe this approach with some implementation details in DHT.

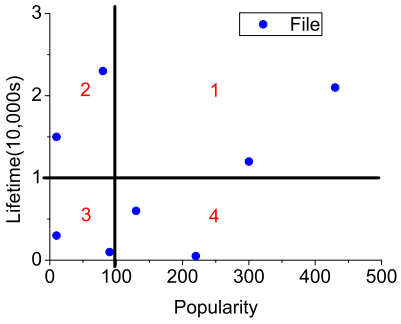

Figure 5: Classification of files

Lifetime and Popularity Based Ranking Approach

From the analysis above, we know lifetime can be used to distinguish between real and fake files. But when the popularity of a file has not reached a certain threshold, its lifetime does not converge, so we should combine popularity with lifetime. In figure 5, files are divided into four zones by their lifetimes and popularities. The files in zone 1, whose popularities are large and lifetimes are high, are more likely to be real files; the files in zone 4, whose popularities are large but lifetimes are low, are more likely to be fake files; the files in zone 2 and 3 should be judged later.

So we design a lifetime and popularity based ranking approach (LIP): it divides files into four zones with lifetime and popularity thresholds, filters out the files in zone 4 and recommends user to select the file with the highest lifetime in zone 1, 2 and 3. The settings of lifetime and popularity thresholds will be discussed in the next section.

Information Collection

This approach only needs the retention times of files. In DHT, the publication node periodically publishes its sharing information, such as a file's name and content hash, to an index node. When the index node first receives a node's sharing information, it does not only record the sharing information, but also records the publication time. When the index node receives the same sharing information from the same node, it will update this user's retention time for this file. For example, U1 publishes the information of F1 to U2 at 13:00 and 15:00 separately. U2 will calculate U1's retention time of F1 as $(15 - 13) \times 60 \times 60 = 7200$s.

Information Processing

When calculating a file's lifetime, it just sums all the owners' retention times up, and divides it by the file's popularity. In DHT, each index node records all the retention times within its responsible area, so it can calculate the popularity and lifetime by itself.

Information Feedback

It filters out fake files with lifetime and popularity thresholds, and sends files' lifetimes with search results to users. In DHT, each index node can do this by itself.

5. Experiment and Results

In this section. We first describe the experiment, and next use training set to choose lifetime and popularity thresholds. We then evaluate the effect of LIP.

In experiment, we simulate a sequence of downloading actions of a title. For each downloading action, we first find out all the files which still have replicas and calculate their numbers of replicas, popularities and lifetimes. Then, we use an approach to choose a file from them and calculate a retention time for this downloaded file.

We can collect the following contents from the logs.

- **File set for a title:** all the files of a title compose the file set for this title. With the two detectors in Section 2, we also label a file as fake or real.
- **Retention time set for a file:** all the owners' retention times of a file compose the retention time set for this file.

- **Downloading action set for a title:** all the downloading actions of a title compose the downloading action set for this title.

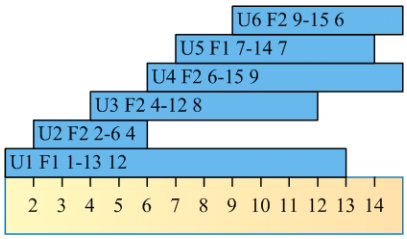

Figure 6: Log collection sketch

Figure 6 shows an explanation of how to collect these contents from the logs of a title. The x-axis is the time and each unit means one second.[1] Each rectangular interval is a user's retention time. User, file, first appearance time, last appearance time and retention time are written in each interval. We can see from the figure that, F1 has two owners who are U1 and U5, their retention times of F1 are 12 and 7 respectively, and the two retention times compose the retention time set for F1; F2 has four users who are U2, U3, U4 and U6, their retention times of F2 are 4, 8, 9, and 6 respectively, and the four retention times compose the retention time set for F2; the six users' downloading actions are (U1, 1), (U2, 2), (U3, 4), (U4, 6), (U5, 7), (U6, 9), they compose the downloading action set for this title.

We should also simulate the following contents.

- **File's retention time:** In simulation, when a user downloads a file, we select a retention time from the retention time set for this file randomly as this user's retention time of this file.

- **Five comparative approaches:**

  1. Replica priority (RP): select the file with the maximum number of replicas.
  2. Lifetime priority (LP): select the file with the highest lifetime.
  3. Select randomly (SR).
  4. Lifetime and popularity based ranking approach (LIP): select the file with the highest lifetime after filtering.
  5. Actual state (AS): it is the actual state of the logs.

Figure 7 shows a concrete example of the simulation with a title. It has simulated U1 to U5's downloading actions at time 1, 2, 3, 4, 8 respectively and is going to

[1]This is only a sketch, so the time begins from 1 and the range is very small. In real situation, it is the global time of log server and the range is very large.

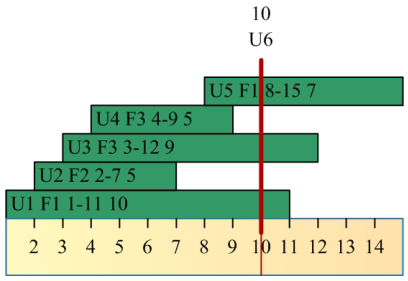

Figure 7: Simulation sketch

simulate U6's downloading action at time 10. It can be seen from the figure that F2's replica has deracinated at time 7 and only F1 and F3 still have replicas. Their numbers of replicas are 2 and 1, their popularities are 2 and 2, their lifetimes are $((10-1)+(10-8))/2 = 5.5$ and $((10-3)+(9-4))/2 = 6$. If we use RP, we should choose F1 and select a retention time from the retention time set for F1 randomly as U6's retention time of F1; If we use LP, we should choose F3 and select a retention time from the retention time set for F3 randomly.

5.1. Training Set

LIP needs two thresholds: lifetime threshold and popularity threshold. Different thresholds have different effects. If popularity threshold is too high, a fake file can only be identified after a lot of downloads. If popularity threshold is too low, the lifetime of a file may have not converged, so we may identify a real file as fake. If lifetime threshold is too high, we may also identify a real file as fake. If lifetime threshold is too low, we may identify a fake file as real.

Our approach should reduce the download volume of fake files both before and after a real file comes out. And we can see from figure 2 that T1 has a lot of fake files both before and after a real file comes out, so we use T1 as training set to choose lifetime and popularity thresholds.

From figure 3, we know a fake file's lifetime converges below 100,000s when its popularity reaches 100, so we choose popularity and lifetime thresholds near 100 and 100,000 respectively. We simulate with different threshold settings. And from the perspectives of reducing the download volume of fake files and guaranteeing the download volume of real files, we choose popularity threshold as 80 and lifetime threshold as 80,000 at last. It's interesting to see that 80,000 seconds is about one day, which means most real files' lifetimes are longer than one day, and most fake files' are shorter than one day.

5.2. Testing Set

We use T2, T3, T4 and T5 as testing sets to evaluate the effect of LIP.

Figure 8 gives the comparison of the download volume of fake files between AS and LIP. We can conclude that

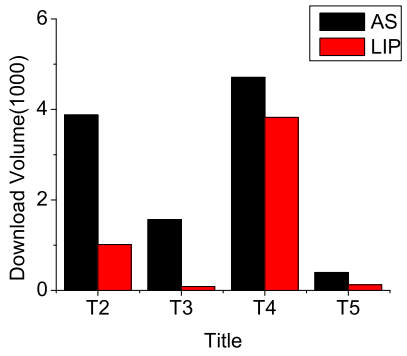

Figure 8: Comparison of the download volume of fake files

LIP can significantly reduce the download volume of fake files most of the time.

When we use other titles in our experiments, there are also some instances that the download volume of real files reduces or the download volume of fake files increases. There may be three factors that influence the effect of LIP.

1. **The quality of fake files.** If a fake file's quality is better than the real file, which means the original movie is more attractive, users may choose to retain the fake file longer than the real file.

2. **The percentage of low quality files in real files.** Because LIP may filter out some real files of low qualities.

3. **Real file's first publication period.** Before a real file's lifetime converges, LIP can't confirm whether it is real, so it may filter out some doubtful real files.

5.3. Comparison of Different Approaches

We continue to use T1 to compare five different approaches and analyze their effect on the download volume of fake files both before and after a real file comes out. From the downloading action set for T1, we know T1's real file first comes out at the 5837th downloading action, which means during the previous 5836 downloading actions, there are only fake files.

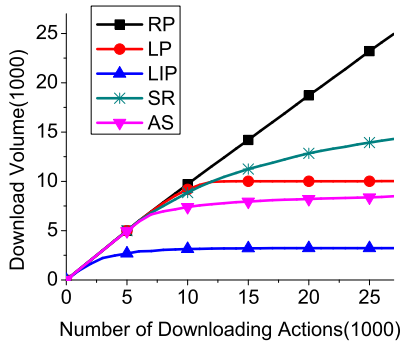

Figure 9: Comparison of five different approaches

Figure 9 shows the download volume of fake files with five different approaches. The x-axis is the number of downloading actions that have happened, and the y-axis is the download volume of fake files. It can be concluded from the figure that RP is the worst, so is SR, LP can significantly reduce the download volume of fake files after a real file comes out, but only LIP can reduce the download volume of fake files both before and after a real file comes out.

6. Conclusion and Future Work

Some users create fake files to gain unfair advantages from incentive policies and it can make fake files pervasive. Two detectors are designed to collect data from logs for our analysis and experiments. We show that lifetime can be used to distinguish between real and fake files, and propose a lifetime and popularity based ranking approach to rank the reputation of files and filter out fake files in P2P sharing systems. The experimental results show that this approach can reduce the download volume of fake files both before and after a real file comes out.

There are many aspects to improve LIP, such as security considerations, the trust of feedback, how to choose lifetime and popularity thresholds.

LIP is not only a binary value to identify real and fake files, it can also reflect a file's quality. We will realize it and see how it works in real situation.

References

[1] Q. Lian, Y. Peng, M. Yang, Z. Zhang, Y. Dai, X. Li, Robust Incentives via Multilevel Tit for tat, In Proc. of IPTPS, Santa Barbara, CA, 2005.

[2] S. Kamvar, M. Schlosser, and H. Garcia-Molina, The Eigentrust Algorithm for Reputation Management in P2P Networks, ACM WWW'03, pages 640-651, 2003.

[3] K. Walsh and E. G. Sirer, Experience with an Object Reputation System for Peer-to-Peer Filesharing, USENIX NSDI, 2006.

[4] J. Liang, R. Kumar, Y. Xi, and K. Ross, Pollution in P2P file sharing systems, Proc. IEEE INFOCOM'05, Miami, FL, 2005.

[5] U. Lee, M. Choi, J. Cho, M. Y. Sanadidi, M. Gerla, Understanding Pollution Dynamics in P2P File Sharing, In Proc. of IPTPS, Santa Barbara, CA, 2006.

[6] D. Dumitriu, E. Knightly, A. Kuzmanovic, I. Stoica, and W. Zwaenepoel, Denial-of-service resilience in peer-to-peer file sharing systems, Proc. ACM SIGMETRICS'05, Banff, AB, Canada, 2005.

[7] N. Christin, A. S. Weigend, and J. Chuang, Content Availability, Pollution and Poisoning in Peer-to-Peer File Sharing Networks, In Proc. of the Sixth ACM Conference on Electronic Commerce (EC'05) Vancouver, BC, Canada, 2005.

[8] J. Liang, N. Naoumov, and K.W. Ross, The Index Poisoning Attack in P2P File-Sharing Systems, IEEE Infocom 2006, Barcelona, Spain, 2006.

[9] M. Yang, B. Zhao, Y. Dai and Z. Zhang, Deployment of a large scale peer-to-peer social network, In Proc. of the 1st Workshop on Real, Large Distributed Systems, San Francisco, CA. USA, 2004.

[10] Divx Database, http://divx.thu.cn, 2006.