Chapter 16

VALIDATION OF RULES USED IN FOXY PEER-TO-PEER NETWORK INVESTIGATIONS

Ricci Ieong, Kam-Pui Chow and Pierre Lai

Abstract Rules have been specified for identifying first seeders in the Foxy peer-to-peer (P2P) network. However, these rules have not been validated due to difficulties in repeating download scenarios. This paper describes a rule validation scheme that uses a network simulation environment. The Type I and Type II error rates of Foxy network monitoring rules over 100 simulation experiments covering ten scenarios are measured and analyzed. The error rates reflect the limitations of the monitoring rules and demonstrate the importance of using network simulations for rule validation.

Keywords: Peer-to-peer networks, Foxy, first seeder, network simulation

1. Introduction

The Foxy peer-to-peer (P2P) network [8], which uses the Gnutella 2 protocol, was one of the most popular Chinese file sharing applications. It was designed to share files with traditional Chinese character keywords or file names in a search-based network. In 2009, the Government of Taiwan took down the publisher of the Foxy client application [8]; however, the Foxy network is still used for P2P file sharing. In fact, with the aid of the Foxy-specific GWebCaches server (Gnutella 2 web cache server for Foxy), Foxy clients can still connect to the Foxy network.

Some widely-publicized information leakage incidents led us to focus on the problem of identifying first uploaders (seeders) in the Foxy network [1]. In general, there are two types of P2P networks for file sharing: link-based networks such as BitTorrent, and search-based networks such as eDonkey, Gnutella, Gnutella 2 and Foxy. As it turns out, the identification of first seeders in search-based P2P networks is more complicated.

In a link-based P2P protocol, the preparation and publication of links and source files are interrelated. Only after the resource links are published can the associated files be downloaded. An investigator can immediately start monitoring the files to be shared based on the links provided, and can detect the users who possess 100% of the files from among the swarms of downloaders. Since an uploader is unable to share a file before the publication of the resource link, monitoring the first appearance of the resource link is sufficient for the investigator to detect the first seeder [1].

In 2007, we developed BTM, a tool for monitoring the BitTorrent protocol [1]. BTM monitored public web forums visited by BitTorrent users to collect torrent files (seed files for download in BitTorrent). By mimicking a BitTorrent client, BTM collected communications information from trackers and peers. Based on the analysis of the collected data, including the percentage of file content possessed by peers, it was possible to identify the users who possessed 100% of the files – these users could be assumed to be the first seeders.

However, not all P2P protocols allow a file requestor to determine the percentage of a file possessed by a peer. For example, the BTM strategy cannot be applied to the Foxy file sharing network. This is because the Foxy network client uses Gnutella 2, a search-based P2P protocol, to locate files.

A search-based P2P protocol [5] does not require a user to announce or perform any publishing prior to sharing a file. Also, it does not include a mechanism for a peer to determine the percentage of a file possessed by another peer. Indeed, a file can be published at any time, even before the keywords of the file are announced on the Internet. Because there is no direct link between publishing a file and announcing the keywords, a different strategy is required to identify first seeders in a search-based P2P network.

In 2009, we introduced heuristic rules for Foxy P2P network investigations [6]. One of the key rules is:

- **Rule 1:** If a seeding peer is found to be reachable and connectable during the slow-rising period, then the seeding peer is the first uploader.

To support this rule, we introduced two supplementary rules to determine the slow-rising period and two restriction rules to address failure situations [4]. These rules can be used to identify first seeders, but the accuracy of the rules was not validated.

The rules in question could not be validated because the platform for performing the validation was not defined. In a peer-to-peer file sharing environment, the first uploader would be unlikely to announce himself as the first seeder. Therefore, it would be difficult to confirm that the identified uploader is, indeed, the first uploader in the Foxy network, even if such an uploader has been identified.

In one experiment, we set up a controlled environment and published a seed file for download. However, we observed that unless the seed file was popular and attracted a swarm of downloaders, the file was not searched for, let alone downloaded.

Our download experiments performed on the actual Foxy network were well planned [4]. We designed the experiments by monitoring and selecting medium- to large-sized files of popular animation series videos. By selecting animation series with new episodes each week, we knew that the files for each episode would not be created or published before a particular time (i.e., screening time). Therefore, it was possible to reduce the waiting period for the uploaded file.

However, there was considerable uncertainty regarding the experimental results. First, we suspected that a single observable seeder should be the first seeder, but we could not confirm that the seeder was, in fact, the first seeder. Because it is not possible for any one entity to monitor the entire Foxy network using current technology, the single observed seeder may not be the first seeder. Second, we observed that some factors (e.g., level-off effects at a Foxy hub) could not be evaluated in the real Foxy environment. In the Foxy network, hubs and swarms are defined based on the situation and the program settings. Thus, increases and decreases in the number of hubs connected to the Foxy network could not easily be achieved. Third, without control factors for defining the experimental environment (e.g., arrival times of incoming downloaders and the file transfer ratio of the uploader and downloader), it was impossible to repeat any experiments. Thus, instead of using a real environment to validate the rules, we decided to use a simulator that would faithfully reflect the behavior of the Foxy network environment.

This paper presents revised versions of the previously-specified heuristic rules for monitoring the Foxy P2P network [4–6]. A validation scheme is proposed for assessing the accuracy and error rates of the monitoring rules. The validation scheme is implemented using a Foxy P2P network simulator developed with the ns-3 network simulation environment.

2. Rule Validation

This section describes the new Foxy monitoring rules and the rule validation procedure.

2.1 Foxy Monitoring Rules

Before we proceed to specify the Foxy network monitoring rules, we list the notation used in this work:

- X: Number of seeders.
- T_i: Time when the i^{th} seeder appears.
- T_1: Time when the first seeder ($X = 1$) appears.
- $R_{j,i}$: Rate of change of the number of seeders ($= \frac{X_j - X_i}{T_j - T_i}$) .
- T_e: Time when the slow-rising period ends.
- T_R: Upper bound of the time when the second seeder appears (derived in [7]).

The following heuristic rule was proposed in [6]:

- **Rule 1:** If a seeding peer is found to be reachable and connectable during the slow-rising period, then the seeding peer is the first uploader.

Rule 1 is refined as the following Foxy monitoring rule:

- **Rule R1:** From T_1 to T_s, the first observed seeder is the initial seeder.

To satisfy the requirement of Rule R1, we define the end time of the simulation T_s based on two schemes, Scheme 1 and Scheme 2, where $T_s = min(T_R, T_e)$:

- **Scheme 1:** We use a fixed time T. The upper bound of $T_R = T_1 + \frac{filesize}{datarate}$ (from [7]).
- **Scheme 2:** We dynamically measure T_e based on the change of the slope of the increment of seeders over time. As shown in Figure 1, if $R_{e+4,e} > R_{e,1}$, then T_e should be the end of the slow-rising period. Note that T_e is the time when the curve enters the rapid rising period.

From the simulation results in Figure 1, $T_s = 110,304$ seconds and the number of seeders $X = 1$. According to Rule R1, the single observed seeder is the first seeder.

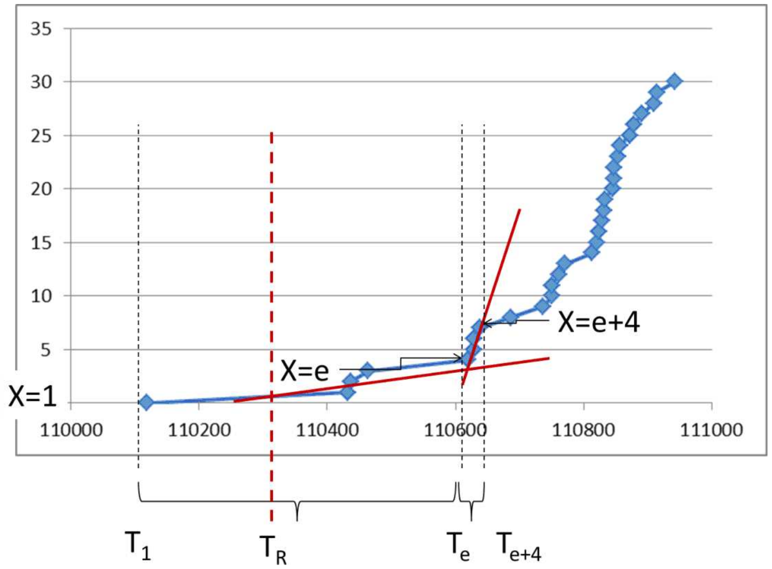


Figure 1. First 30 downloaders in the seeder curve.

2.2 Experimental Set-Up

We developed FoxyNS3, a new Foxy simulator, to help validate the Foxy monitoring rules. Our previously-developed discrete event simulator SimFoxy was not used because it was unable to fully simulate and explain important aspects of network connectivity and congestion behavior that occur in real P2P networks.

Our SimFoxy simulator focused on discrete event simulation. All downloading events were simulated based on the estimated file packet transfer time. The transfer time was calculated based on the packet size and the bandwidths of the downloader and uploader. When additional downloaders appeared in the network, the available network bandwidth was assumed to be shared by the downloaders. However, in a real Foxy network, the bandwidth is not merely shared by the downloaders – traffic congestion in the network also affects the download time.

Another problem with SimFoxy was that the maximum number of network connections was not simulated. Thus, the effect on the download time induced by the limits imposed on uploader and downloader connections was not considered.

More importantly, SimFoxy was created to analyze only the file transfer protocol. The file searching mechanism and the Foxy network topology were neither considered nor simulated. However, because of the limits on the number of leaf nodes connected to a hub and the number of hubs connected to a leaf node, search query responses could not be fully recorded in any part of the real Foxy network. Clearly, this impacts the accuracy of the Foxy network simulation results.

Our new FoxyNS3 simulator was built using ns-3, a popular discrete-event network simulator. The ns-3 simulator, which is wrapped with Python, incorporates a C++ library that provides a set of network simulation models implemented as C++ objects. An important advantage is that ns-3 is able to simulate the Gnutella 2 protocol at the network level. In fact, network activities can be simulated in terms of file packet exchange.

FoxyNS3 was developed as a C++ program that interacts with the ns-3 simulation model library. The main simulation configurations, simulation network topology and network configuration parameters are defined in the file *test-gnutella.cc*, which controls the entire simulator.

We employed the CSMA network topology in our experiments. This topology has been used in network simulations of other P2P file sharing protocols. The FoxyNS3 `configure.txt` file defines all the node connection settings, including the connection set time, search time, node connection rate, network departure time and re-search time.

The FoxyNS3 simulator models a Foxy network using three key components: bootstrap server, hub node and leaf node models. The file `bootstrap.cc` defines the behavior of the GWebCache Server. The bootstrap program collects a list of available hub nodes and returns the list to a newly-connected leaf node. To reduce network connections and the memory used during a simulation, the bootstrap server disconnects from the leaf and hub nodes after connecting a leaf to a hub.

The hub node model (`hub-node.cc`) is used to simulate the behavior of hub nodes in a Foxy network. It has some features of a real Foxy hub node. In particular, it serves as a node that is connected to leaf nodes; however, the number of leaf nodes that can be connected to a particular hub node is limited. New leaf node connection requests are rejected after the maximum number of leaf nodes that can be connected to a hub node is reached. The available uploader lists are also maintained. When a connected leaf node or hub node submits a file request to a hub node, the node checks if the requested file is available at any of the nodes. Then, it relays and broadcasts a file request to its leaf nodes or hub nodes when the file request does not exceed the time-to-live limit.

In our simulation, we simplified hub searches by only returning the hubs that could be connected by leaf nodes instead of providing hubs that could not be accessed. In other words, a hub node cannot and will not participate in file downloading. This speeds up the initial setup of the network without affecting the topology.

The leaf node model (`leaf-node.cc`) is a simplification of a leaf node in the real Foxy network. A simplified version of the Gnutella 2 protocol was also implemented in the simulator. Although the ping/pong packets and other network communication packets were not coded as precisely as in the Gnutella 2 protocol, the behavior of the protocol was simulated adequately. All the search and download features are initiated from the leaf node model. After a leaf node obtains the hub list from the bootstrap server, it connects to three hubs. After the connection is established, the leaf node uses UDP packets as ping and pong to maintain connectivity. A leaf node assigned to participate in file sharing activities searches the connected leaf nodes. The file transfer feature was developed using a modified version of the HTTP Content-Range protocol; this simulates the download and upload behavior, albeit without certain download features.

2.3 Simulation Assumptions

As mentioned above, the FoxyNS3 simulation model implemented Foxy clients in the CSMA network topology. To reduce network congestion, an Internet stack was used and the backbone network in the simulation was set to 1 Gbps.

The leaf and hub nodes were disconnected from the bootstrap server after the server lists for establishing connectivity were collected. In the real Foxy network, the bootstrap server sends the hub list to a leaf node whenever the leaf node issues a request. However, this activity is not performed frequently. Therefore, our simulator limited the request to one time only, assuming that the leaf node would only collect this information once after the connection was established – this reduces network traffic between the bootstrap and leaf nodes. Moreover, to streamline the Foxy network simulation, all the leaf nodes were pre-set to connect to the hub within the first 100,000 seconds. All file search and download activities were configured to start after 100,000 seconds.

Based on settings established by experiments on the real Foxy network, each leaf node was permitted to access and connect to no more than three hub nodes. Also, each hub node was permitted to connect to a maximum of 200 leaf nodes and three neighboring hub nodes.

To focus the simulation on file sharing activities between the leaf nodes, leaf nodes were permitted to depart after completing their downloads, but hub nodes were forced to be connected to the network. Thus, the scenario of a hub leaving the Foxy network was not simulated.

The upload and download connectivity were pre-set in the FoxyNS3 simulator. As in the real Foxy network, all Foxy clients were configured to support a maximum of three downloads and three uploads.

The file fragment size used in the simulated Foxy network was the same as in the real network. The transferred file fragments were set at 512 KB per packet, the packet size in a download packet request query in the real Foxy network. Thus, in our experiments, around 26 512 KB packets are downloaded by the leaf node for a 13 MB sample file.

2.4 Simulation Experiments

More than 100 simulations were performed using FoxyNS3. 50 of the simulation experiments were used for rule validation. The experiments were performed by varying four parameters:

- T_{arr-s}: Average inter-arrival time of search queries.
- N_p: Number of downloaders interested in a target file during the simulation period.
- N_h: Number of hub nodes in a Foxy network swarm.
- Upload and download data transfer rates for all the peers.

Certain parameters in FoxyNS3 were pre-set. The file size was fixed at 13 MB, the same size as in the experiment with the real Foxy network. The file packet size was set to 512 KB, the same size used in the Foxy network. The download slot limit in the uploader, which is the number of maximum downloading peers an uploader can handle simultaneously, was found to be and set to three. The number of simultaneous downloader connections to the uploader was also set to three.

The simulation experiments were divided into four sets:

- **Set 1:** This set of simulations focused on the sequential search time. In Sets 1(a), 1(b) and 1(c), experiments were performed by varying the inter-arrival time of search queries (T_{arr-s}). In Set 1(d), 40 out of 100 downloaders were submitted to the Foxy network in sequential order, while 60 out of 100 downloaders were submitted randomly.
- **Set 2:** This set of simulations focused on the random search time. In Sets 2(a), 2(b) and 2(c), the search query time entries were pre-generated, kept in the `configure.txt` file and used as the query time in the experiments. In Set 2(d), the simulation was conducted with Poisson time set to 100,000 seconds.

- **Set 3:** This set of simulations focused on the number of hubs present in the Foxy network. Sets 3(a) and 3(b) were performed using the same set of search query times as Sets 2(a) and 2(b), but were measured at the hub level instead of the overall view. In our simulation environment, leaf nodes were connected to hubs; also, according to the connectivity restriction, a leaf node can connect to no more than three hubs. Thus, the first seeder and other downloaders cannot connect to all ten hubs. The rule accuracy was evaluated by collecting the results from each hub during each simulation.

- **Set 4:** This set of simulations focused on the random data transfer rate. In the other simulation sets, the data transfer rates were pre-set to fixed rates in order to minimize randomness in the simulations. However, in the real Foxy network, data rates are not fixed and variations in the data rates affect the overall download speed for the Foxy swarm. In Sets 4(a) and 4(b), experiments were performed with 500 nodes, 200 downloaders and a mean data rate of 100 Kbps.

2.5 Simulation Results

Simulations were performed on two Linux machines running the ns-3.11 version. In each simulation experiment, the following information was collected from each inspected node with the downloader and uploader participating in file sharing:

- IP addresses of the uploader and downloader nodes.
- Start connection time, start search time, first packet received time, download completion time, download completion duration time, leave time, reconnection time and search time for each node.
- File download completion order.
- Number of successful and failed download requests (generated from an inspected node).
- Hub(s) to which each leaf node was connected.
- Number of file packets and data volume shared or uploaded by each node.

The results were stored in a text file (csv format). The seeder curves, representing the number of uploaders observed in the Foxy network, were created by analyzing the downloader completion time and file download

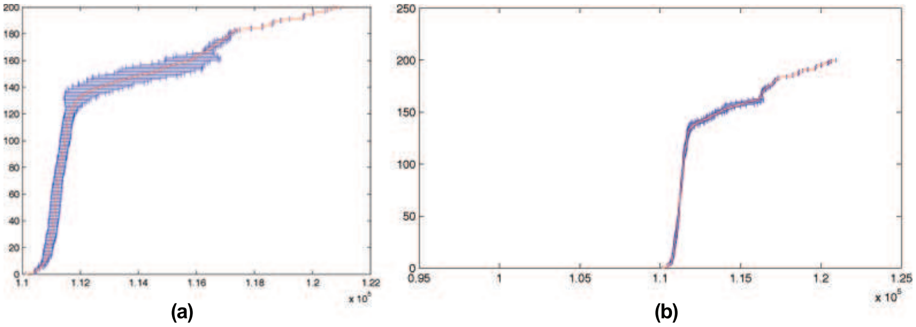


Figure 2. Mean and standard deviation values for Sets 2(a) and 2(b).

completion order. Although each set of simulation experiments was performed using the same values of node connection time and start search time, the network behavior and the selection criteria related to the order of the hubs varied randomly. Therefore, when congestion occurred in a simulation, a deviation from the expected mean was observed.

Figure 2 shows the mean and standard deviation values for typical seeder curves. Note that the graphs plot the seeder appearance time versus the number of seeders. The average within an experiment set is shown as a solid line while the deviations are shown as horizontal error bars. The results demonstrate that the deviations for Sets 2(a) and 2(b) are within a reasonable range.

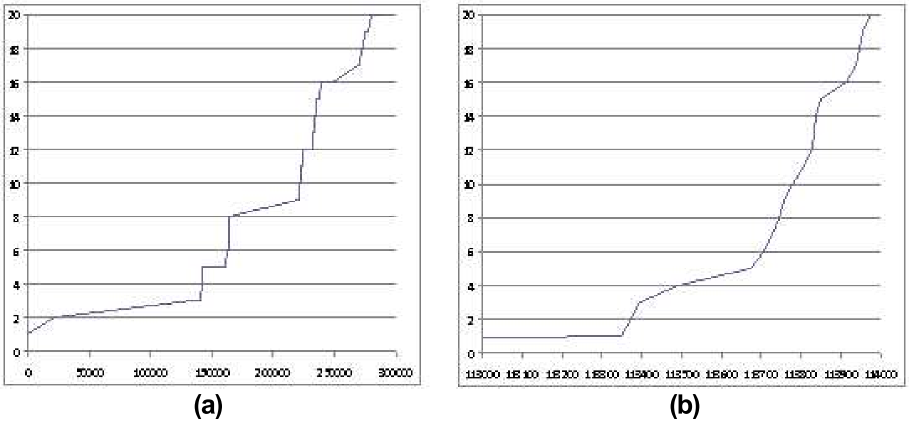


Figure 3. Seeder curves for the real and simulated Foxy networks.

Figure 3 shows the real and simulated seeder curves. In Figure 3(a), details of the first twenty incoming seeders in a slow incoming download scenario of a 100 MB file are recorded in the seeder curve. Figure 3(b)

Table 1. Validation results.

Experiment	Correct Seeder	Type I Error	Type II Error
Set 1(a)	0%	0%	0%
Set 1(b)	100%	20%	0%
Set 1(c)	100%	0%	0%
Set 1(d)	100%	20%	0%
Set 2(a)	100%	100%	0%
Set 2(b)	100%	100%	0%
Set 2(c)	100%	100%	0%
Set 2(d)	100%	0%	0%
Set 3(a)	30%	60%	0%
Set 3(b)	30%	50%	0%
Set 4(a)	100%	100%	0%
Set 4(b)	100%	0%	0%

shows that the simulated seeder curve reflects much of the detail in the real seeder curve. This demonstrates that the simulation replicates the behavior in the real Foxy network.

3. Validation Results

After performing the simulations, we applied Rule R1 in an attempt to determine if the first seeder could be identified. The rule validation scheme involves three steps:

- **Step 1:** If at time T_s, a single seeder is observed and the seeder is the first seeder, then the finding is considered to be correct.
- **Step 2:** If at time T_s, more than one seeder is observed, then the finding is classified as a Type I error.
- **Step 3:** If at time T_s, the seeder is rejected, then the finding is classified as a Type II error.

3.1 Rule R1 Results

Table 1 presents the rule validation results. Several observations can be made based on these results.

Effect of T_{arr-s}: In Sets 1(a) through 1(d), when the inter-arrival search times (T_{arr-s}) were shorter than the download time for a single file, Rule R1 could identify the first seeder. When the T_{arr-s} times were much longer, Rule R1 failed to identify the first seeder. The rule also

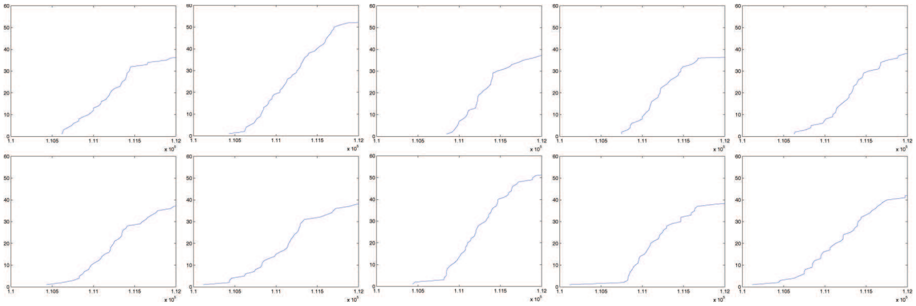


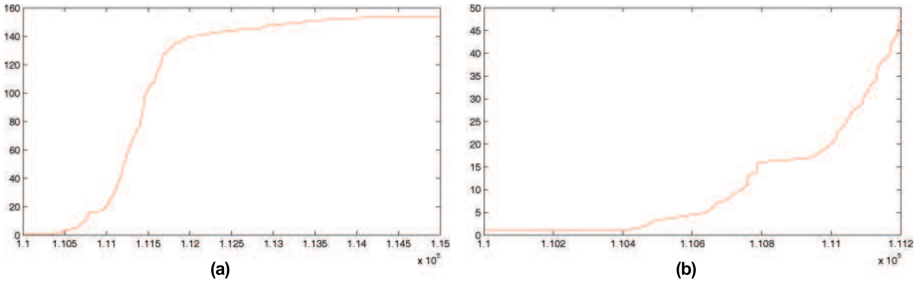
Figure 4. Seeder curves collected at ten hubs in Set 3(b).

incorrectly classified 20% of downloaders as the first seeder. Thus, when the time T_{arr-s} is shorter, there is a higher likelihood of identifying the first seeder in the swarm.

Effect of Random T_{arr-s}: In Sets 2(a) through 2(d), the inter-arrival search times were changed from a fixed search schedule to random search times across the 100 downloaders. In all cases, it was possible to successfully identify the first seeder using the monitoring rule. However, the higher Type I errors are observed compared with Sets 1(a) through 1(d). With the change in search start time, the download completion time affected each downloader and, thus, more downloaders were incorrectly classified as first seeders.

Observations of Hub Effect: In the case of Sets 3(a) and 3(b), we observed that the seeder curve at each hub node shown in Figure 3 (corresponding to Set 3(b)) differed from the overall seeder curve in Figure 2(b). When more hubs were involved, the shape of the seeder curve deviated more from the single hub seeder curve. This caused difficulties when using Rule R1 to identify the first seeder. Without the overall view of the download, it was possible to simply rely on Rule R1 and its supplementary rules to determine the first seeder. In the Set 3 simulations, because seven out of ten hubs did not connect to the first seeder, simply relying on Rule R1 could incorrectly identify a normal downloader as the first seeder. Although the Type I errors shown in the validation results are just over 50%, when more hubs were involved in a download, the likelihood of incorrectly identifying a downloader as the first seeder increases.

Observations of Transfer Rate: Figure 4 shows the overall and zoomed-in portions of the seeder curve corresponding to Set 4(b). The



(a)

(b)

Figure 5. Overall and zoomed-in portions of the seeder curve in Set 4(b).

variation in the data transfer rate adds more stepwise details to the seeder curve. The validation results indicate that the error rates are not affected by a data transfer rate variation.

3.2 Observations

Several observations can be made based on the simulation results.

- When no nodes are permitted to leave the network, Rule R1 can identify the first seeder when the download time of a file is longer than the inter-arrival time of search queries. In other words, if the swarm of downloaders is interested in a file and the file size is sufficiently large, then there is a higher likelihood of identifying the first seeder.

- If the inter-arrival time of search queries is shortened, then the Type I error increases. That is, more downloaders are incorrectly identified as the first seeder.

- In contrast, if the inter-arrival time of search queries is lengthened, then the Type II error increases and the first seeder is likely to be rejected. This is because no slow-rising period is observed in the seeder curve. Thus, the seeder curve cannot be used to determine if the single uploader observed is the first seeder or just a downloader who has remained in the network.

- Most search-based P2P networks such as Gnutella 2 and Foxy P2P client rely on hub nodes to separate a swarm into islands. With these hubs, monitoring the first seeder based on the number of seeders leads to high Type I errors. For example, the results of the simulations involving ten hubs demonstrate that more than 50% of the first seeders were incorrectly identified.

4. Conclusions

Identifying first seeders is critical in P2P network investigations of illegal file sharing. Monitoring rules can help identify first seeders. However, the rules must be validated in order to estimate the likelihood that an identified first seeder is, in fact, the first seeder.

It is well known that the error rates of P2P monitoring rules are typically high and that the error rates are even higher when data is collected from a fragment of the entire network. Since it is not possible to collect all the data from a real P2P network, rule validation based on data collected from an extensive network simulation using a simulator such as FoxyNS3 is a promising approach.

Our future research will focus on identifying new attributes to enhance the P2P monitoring rules. We will also extend our experiments to incorporate more hubs and permit uploaders to leave the network during the slow-rising period. These extensions will help refine the monitoring rules to better accommodate real-world P2P network scenarios.

References

[1] K. Chow, K. Cheng, L. Man, P. Lai, L. Hui, C. Chong, K. Pun, W. Tsang, H. Chan and S. Yiu, BTM – An automated rule-based BT monitoring system for piracy detection, *Proceedings of the Second International Conference on Internet Monitoring and Protection*, p. 2, 2007.

[2] K. Chow, R. Ieong, M. Kwan, P. Lai, F. Law, H. Tse and K. Tse, Security Analysis of the Foxy Peer-to-Peer File Sharing Tool, Technical Report TR-2008-09, Department of Computer Science, University of Hong Kong, Hong Kong, China, 2008.

[3] B. Fan, D. Chiu and J. Lui, Stochastic differential equation approach to model BitTorrent-like P2P systems, *Proceedings of the IEEE International Conference on Communications*, pp. 915–920, 2006.

[4] R. Ieong, P. Lai, K. Chow, M. Kwan and F. Law, Identifying first seeders in Foxy peer-to-peer networks, in *Advances in Digital Forensics VI*, K. Chow and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 151–170, 2010.

[5] R. Ieong, P. Lai, K. Chow, M. Kwan, F. Law, H. Tse and K. Tse, Forensic investigation and analysis of peer-to-peer networks, in *Handbook of Research on Computational Forensics, Digital Crime and Investigation: Methods and Solutions*, C. Li (Ed.), IGI Global, Hershey, Pennsylvania, pp. 355–378, 2010.

[6] R. Ieong, P. Lai, K. Chow, F. Law, M. Kwan and K. Tse, A model for Foxy peer-to-peer network investigations, in *Advances in Digital Forensics V*, G. Peterson and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 175–186, 2009.

[7] P. Lai, Profiling Internet Pirates, Ph.D. Dissertation, Department of Computer Science, University of Hong Kong, Hong Kong, China, 2011.

[8] Wikia, The Encyclopedia of Virtual Communities in Hong Kong on Foxy, San Francisco, California (`evchk.wikia.com/wiki/Foxy`).