A Simulative Framework to Estimate P2P Performance Indexes and Traffic Matrixes

Raffaele Bolla, Roberto Bruschi, Franco Davoli, Andrea Ranieri, Riccardo Rapuzzi, Michele Sciuto
DIST - Department of Communication, Computer and System Sciences
University of Genoa
Via Opera Pia 13, 16145 Genova, ITALY
raffaele.bolla, roberto.bruschi, franco.davoli, riccardo.rapuzzi, michele.sciuto@unige.it and andrea@reti.dist.unige.it

***Abstract* - Today, a large part of the traffic carried by the Internet is generated by file-sharing applications based on the P2P paradigm. Moreover, P2P traffic has particular features, quite different from the ones generated by "classical" data applications, and generally causes highly dynamic and distributed traffic matrixes that heavily stress network infrastructures. With this respect, it is reasonable to say that the analysis and study of the overall network performance have to take into account P2P traffic characteristics and behavior. To this purpose, the present paper approaches P2P traffic modeling activities, by describing a Gnutella simulator developed at the Telecommunication Networks and Telematics (TNT) Laboratory of DIST. The aim is well founded on the partial lack of simulators that effectively and completely describe the quantity, the characteristics and the impact of P2P traffic in networks.**

Keywords – Distributed systems; File-sharing applications; P2P modeling; Traffic simulation; Internet traffic.

I. INTRODUCTION

In the last decade, Internet technologies and infrastructures have experienced deep advancements and evolutions to meet and, somehow, to anticipate the increasing user requirements and support for new applications. Among the many topics, on which industrial and scientific research communities have concentrated their effort, Quality of Service (QoS), for example, has held a very important role: a large number of protocols and architectures have been proposed to extend the Internet capabilities beyond the simple best-effort service. Besides all these important efforts and works about the introduction of new network capabilities, also best-effort services experienced changes and developments with the introduction and spreading of new applications. Moreover, many measurement studies [1], [2], [3], [4], [5] report that, today, data traffic still represents the large part of the whole traffic carried by Internet. In this respect, it is reasonable to say that the analysis and study of the overall network performance have to heavily take into account data traffic characteristics and behaviors. It is well known that this significant traffic amount results to be fed only for a small percentage by "classical" Internet application data (i.e., Web, e-mail, FTP, etc.), but basically it is generated by file-sharing applications. Reference [6] outlines that this kind of applications, based on the Peer-to-Peer (P2P) technology and generally characterized by a fast and epidemic spreading and a bandwidth intensive nature, causes about 60-80% of the overall Internet traffic. P2P traffic has particular features, quite different from the ones generated by "classical" data applications, since, making any peer able to act simultaneously on both client and server sides (e.g., every peer can request or provide a service), P2P-based applications generally produce significant traffic flows that cross the network between very random and dynamic end-points. In fact, file-sharing applications realize, in every respect, distributed content databases, by interconnecting peers (generally more than 20,000 nodes scattered along the whole Internet with highly random lifetime) in huge and dynamic overlay networks, whose architecture and structure are completely defined by the specific P2P protocol adopted. The latter defines also the mechanism used to query the distributed database, and then, when a generic peer issues a query to request a content, the protocol fixes the scope of neighbor peers to which the request is sent and that can respond to it. In this respect, P2P traffic stresses network infrastructures with highly dynamic and distributed traffic matrixes, whose vertexes heavily depend on the overlay network and on the query mechanism implemented. Moreover, the success and the large-scale diffusion of file-sharing applications, like the well-known Gnutella [7], eDonkey2000 [8], eMule [9], Direct-Connect [10] and Kazaa [11] have already started to influence network planning and dimensioning: for example, the diffusion of broadband access in residential networks, where users require more and more bandwidth, is mostly due to the presence of P2P traffic and to its bandwidth intensive nature [12]. In such kind of environment, our main objective is to study, to characterize and to analyze the traffic produced by file-sharing applications and to understand how it can impact on underlying network infrastructures.

Many modeling and simulation works can be found about Gnutella network and, more in general, P2P overlay networks. The general context in which we intend to insert this research activity is founded on the effects of P2P traffic on both the transport and network layer; the most part of P2P simulators have been developed for studying the characteristics and the behaviors of such applications/protocols at the application layer only [13], [14], [15], [16], [17]. An interesting approach is the one presented in [15], where a framework for P2P simulation environments, that can be built on top of existing packet-level network simulators, has been developed; in this way the underlying layers seem to be deeply considered, but driving the study into a packet level approach, which could not scale to the size of a real P2P network. In this respect we intend to present an application layer simulative framework,

developed with the aim of studying the Gnutella query process, and, more in general, in order to extend the simulation study to the transport and network layers too. The future activity will consider the impact of P2P flows on the network, using TCP fluid simulative models, with an aggregate, and then more scalable approach.

The paper is organized as follows. Section II describes the proposed P2P simulation model, while the corresponding numerical results are discussed in Section III. Finally, in Section IV the conclusions are reported.

II. Model Description

A. Introduction

As previously sketched, our goal is to develop a tool to study and to analyze the impact of P2P traffic over underlying network infrastructures. To this purpose, we have designed a simulative tool that should effectively represent the overlay network dynamics and the query process of a common file-sharing application. The query process is a key element for identifying the end-points of P2P data transfer flows. To achieve a reliable model of the query process, we obviously have to take into account how peers are interconnected at the application layer, and how they can communicate. In other words, in order to model the query process, we first have to represent the overlay network behavior. Note that, the proposed simulative tool can be thought of as a mechanism to estimate P2P traffic matrices that can be easily integrated with classical transport/network layer traffic models as, for instance, that described in [18].

B. Overlay Network Architecture

We took Gnutella as reference protocol, which relies on a hierarchical unstructured overlay network. We decided to model the Gnutella protocol because it is a quite utilized one, it is well described and it has a deep analogy with eDonkey2000 [8] (which is the currently most popular P2P application).

In this Section we describe the most significant overlay network elements of a file-sharing application: peers, ultra-peers and files. Peers represent the leaves of the overlay network; ultra-peers have the same properties of peers, but they also have to satisfy the queries; files represent the content, which interests peers.

Peer: in the Gnutella network, peers are called leaves; they are the usual network clients, they do not have particular responsibilities, they do not provide interconnections in the overlay network and they have limited resources in terms of computational capacity and memory size. Every peer has a location in the network; this aspect will be important in the next evolution of this work, in which we will study the impact of P2P applications (file download) at the network/transport layer. In order to participate in the overlay network, every peer needs a connection (relationship) with an ultra-peer. For each relationship the peer keeps in memory the ultra-peer and edge-node identifiers, in order to maintain knowledge about its location with respect to both network and application layer. The main characteristics of a peer can be: connection bandwidth, download queue size and list of files. Connection bandwidth indicates the bandwidth of the access circuit (e.g., three different types of access can be represented as 10, 100 and 1000 in order to simulate dial-up, ADSL and Fast Ethernet LAN connections); as mentioned in [5], 10% of peers have dial-up connections, 60% ADSL and the remaining 30% have local area network accesses. Download queue size represents the number of download processes, requested to the peer having the desired file; this parameter, in conjunction with the connection bandwidth, is considered after a query process, in order to choose the best suitable peer for the file download (with the highest ratio bandwidth/number of downloads). The list of files is what the peers share.

Ultra-peer (called also Server in other contexts): they are the most important peers in the overlay network; they represent the nodes of the network at application layer. They are characterized by a list of connected peers and by a list of neighbors (at application level), which have one hop of distance among themselves (the neighbors are the ultra-peers to which queries are forwarded).

File: they are the “goods” requested from the peers; typical parameters associated to files are: number of replicas, size and popularity. The number of replicas is the number of times the file is replicated in the network; we assumed that every peer could just keep one replica of a file, so the number of replicas expresses the number of sources holding the file. Size indicates the file dimension; we divided files in three classes: 10, 100 and 1000. The percentage of files for every class respects the following proportion: 30% of files has a file size of 10, 60% has a size of 100 and 10% has a size of 1000. Popularity represents the level of interest for every file in the system; the greater the popularity is, the more the file is requested from peers.

C. Simulator general description

The developed simulator tries to represent two dynamic processes of a Gnutella P2P network: the overlay network evolution process and the query process. The first one models the ingresses and egresses of peers in and out of the overlay network. The arrival of a new peer causes the creation of new relationships in the overlay network and the potential associations of newly shared file copies to the peer. On the contrary, when a peer exits the overlay network, all its relationships and its shared files have to be erased from the system.

Two important aspects should be underlined before the detailed description of the tool. First, the data transfer is not simulated but, if the file has been found, a copy is considered immediately transferred to the requesting peer (and then it can be shared or not). Second, the number of different files in the P2P network is fixed at the beginning and it is not changed during the simulation.

The structure of the simulation tool can be roughly divided into three parts: the initialization, the overlay network dynamics description and the query process description.

D. Initialization

In this Subsection we describe how the following main quantities of the simulation are initialized:

- Total number of peers and ultra-peers.
- Total number of different files in the networks.
- Number of replicas.
- Distribution of the files in the peers.
- Popularity of the files.
- Peer and ultra-peer relationships.

The initial numbers of peers and ultra-peers are input parameters.

The number of replicas is a specific parameter associated with each different file. We assign randomly this parameter to each file, but according to the Zipf distribution [17]. It is assumed that each peer can maintain no more than one copy of each file; in this way, it is clear that the maximum number of replicas for every file is upper-bounded by the total number of peers.

Files are placed in the network according to the following allocation [5]: 25% of the total number of peers shares zero files (these peers are called "free-riders"), 70% of peers share up to 100 files and 5% of users share more than 1000 files. This type of subdivision is not a strict constraint, but it is a general objective. If the number of files is small, with respect to the number of peers, it is not possible to satisfy the above proportions. In this last case we maintain the percentages but we reduce the number of files, which characterize each category (e.g., 10 instead of 100 and 100 instead of 1000).

The file popularities also follow the Zipf behavior, according to this distribution; a parameter related to the level of interest in the network is assigned to every file, in inverse relation to the file IDs (i.e., the most popular file is the one with file ID 0). It is relevant to notice that the popularity and the number of replicas are not correlated at the beginning of the simulation process; we expect that the more the time advances, the more the two parameters become correlated, following the same Zipf behavior.

For what concerns the relationships, they are generated following the same rules that will be used during the simulation. Each peer has one relationship with an ultra-peer. This ultra-peer is selected randomly with uniform probability among the ones that have less than 150 connected leaves [7].

For what concerns the ultra-peer relationships, they are defined by using a mesh degree, which can assume a value in the range between zero (no neighbors) and 1 (fully-meshed overlay network). The number of relationships for each ultra-peer is obtained by multiplying the mesh degree for the total number of ultra-peers. Another constraint is considered in this case, i.e., the Gnutella protocol provides that every ultra-peer has a minimum of 5 to a maximum of 30 bidirectional relationship with other ultra-peers.

E. Overlay Network Dynamics

In this Section we describe the overlay network dynamics, in particular the events provided by the simulation process. The overlay network dynamics are caused by the login and logout processes, relative to ultra-peers and leaves. In this first version, we have used an exponential distribution to generate the inter-arrival time of these events, even if other authors (see [19]) suggest the usage of a power-law distribution.

An ultra-peer can enter in the system in case of a new ultra-peer birth or in case of a peer election to ultra-peer status. The model adds a new ultra-peer element in the overlay network, performing the creation of the neighbor list for the ultra-peer; the new ultra-peer joins the network, by contacting a random chosen ultra-peer. This ultra-peer forwards the list of its neighbors to the joining ultra-peer, which contacts immediately the ultra-peers in the list. Every neighbor's list gets an update, derived from the ultra-peer presence.

An ultra-peer logging out causes the assignment of its leaves to a randomly chosen ultra-peer; its neighbor list, too, is assigned to another ultra-peer.

A peer logging in provides the assignment of a number of files to share. The number of files is determined by following the previously defined percentages; as already mentioned, the model does not provide new file creation; the files assigned to the new peer are copies of those already present in the system. The peer joins the overlay network by establishing a relationship with an ultra-peer (chosen by following the procedure already described in the initialization sub-section). We assume that a peer joining the overlay network needs to search a file; this assumption implies that a query event always follows every peer login.

A peer logging out causes a decrement in the number of replicas, for all shared files. The ultra-peer (to which the peer was connected) needs to update its list of leaves.

F. Query Process

Query events are distributed in time following an exponential behavior, as suggested in [17].

Peers processing queries are chosen randomly among the ones joining the overlay network; each peer can perform a different number of queries from 0 up to 9 inclusive, in the same instant.

The sorting of files depends on the popularity of each file; the more a file is popular, the higher is the probability that it might be requested. When the file is chosen, two conditions must be satisfied: the file must not be present in the list of shared files maintained by the peer, and it must not have already been requested in the n previous queries (with $n \in [1,2,...,9]$). If these conditions are satisfied, the query process comes to a good end and the requesting peer can ask its ultra-peer to search the file.

During the searching process the ultra-peer receives the file ID, it searches the file at one-hop distance, contacting its leaves and the directly connected ultra-peers, the neighbor ultra-peers receive the file ID and forward it to their leaves. The process terminates when information is completely

received from the peer; the final result is a list of peers (at one-hop distance) sharing the requested file. If no peer is found, the process is repeated also for two-hop distance ultra-peers, which search the file in their attached leaves. When the list of peers sharing the file is received, the best peer must be selected; this choice is based on two important parameters, which describe the amount of bandwidth of the peer and the number of download processes the peer is satisfying. At this time, the second parameter is randomly assigned to the peers. The selected peer is the one having the largest ratio between the available bandwidth and the number of download processes. At this time the network level is not simulated, the time needed to download a file is not considered; we assumed that when the peer is selected, the file is transferred from the sharing peer to the searching peer. When the file is received by the peer, the user requesting it can choose whether to insert it in the list of shared files or not to share it (in this case the file is removed from the system); the number of replicas of the file is incremented or not, according to this choice. The decision to share a file is determined with a probability of 50% (sharing or not), this probability is zero in the presence of free-ride peers.

G. The detailed simulator

With the aim of validating our model, we have developed also a second simulation tool; we will refer to this second simulator with the term “Detailed”, and to the previous one with the term “Fast”. The difference consists in defining and implementing the specifications of the P2P protocol more accurately, in order to get a more precise representation. We described before that when the downloader peer selects the uploader one, it is not a must that the file is shared from the downloader; it can decide to share it or not. If the peer decides not to share the file, no information remains about the query and download processes; this means that, at the next step, the same peer can request the same file, letting the number of replicas unchanged. In the Detailed simulator, it is not allowed that the same file can be requested from the same peer (in this way the behavior is closer to the reality). In the Detailed simulator a list of non-shared files is added to every peer; the list is filled during the simulation, and allows a trace of the already requested files to build up, with the target of not requesting the same file twice or more. In this way also the query process gets a change: if the file is present in the not-shared file list, the query is not performed.

III. SIMULATION MODEL RESULTS

A. Introduction

In this Section we present some numerical results that try to show how precisely the simulator can represents the behavior of a Gnutella network. The obtained parameters are related only to the application layer and they represent the distribution of file replicas, the number of performed queries, and the number of visited ultra-peer in the function of time; also the behaviors of the number of queries and of the number of replicas, in the function of popularity, are reported.

We present results for different size overlay networks, in particular the number of peers for each reported simulation is 2500, 25000 and 250000. In order to present the application layer behavior during the simulation, we sampled the simulation time in fixed intervals, whose size depends on the complete simulation time.

No results are reported about the network/transport layer while the integration between the application and network layer simulators is a work in progress. The scalability has been specifically tested by considering increasing overlay network sizes and by reporting indication about computational and memory resources utilization. We have compared the two versions of simulator introduced in Section II, namely the Fast and the Detailed simulators.

For each selected overlay network size, we have fixed some of its most important parameters trying to match the Gnutella specifications and P2P measurement indications.

For example, consider the mesh index, Gnutella protocol suggests a number of ultra-peer neighbors between 5 and 30; the large part of simulations has run with an average number of 15 neighbors for every ultra-peer [7]. As sketched in Section II, we have also to take into account that only the 5% of peers shares more than 1000 files. This, especially in cases of little sized networks, may generate troubles due to the little number of files in the system: in such cases, we have decided to scale also the typical sharing dimensions by assigning to these classes of peers lower numbers of files. Peer birth and death inter-arrival time are generate by following a Poisson distribution.

In order to make the simulative results stable, the simulation time has been incremented according the raise of network size.

In the following sub-sections we intend to present simulation results for three different overlay networks: a small one (2500 peers, 650 files and 25 ultra-peers) and two wider ones, 25000 and 250000 peers, whose sizes are closer to real P2P networks. At the end of the chapter, in order to characterize the model scalability, considerations about the simulation time, computational and memory resource requests are reported.

B. The Smallest Overlay Network case (2500 peers)

As first studied case, we report the numerical results of overlay networks with only 2500 peers. In particular, for what concerns the overlay network structure, we have fixed the mesh index equal to 0.2, in other words we have imposed an average number of 5 neighbors for every ultra-peer. It is important underlining that, just for this type of small sized network, we have reduced the minimum number of neighbors from 5 to 2, exclusively to allow a correct simulation. Starting from 650 files, the total number of replicas results equal to 12437. In addition, we set an upper bound of 25 files to the 5% of peers, which have large shares. The simulated time is equal to 14515200 sec (i.e. 24 weeks).

In every shown simulation session birth and death events of peers and ultra-peers are considered, driving the analysis to the case of dynamic overlay networks closer to the reality; for every simulation also the static case (without births and deaths) has been run, although the resulting graphs are not

reported.

In dynamic simulations we have noticed a little increasing of the overall number of issued queries with respect to the static situation. This behavior can be explained by considering that, in the dynamic case, all the peers joining the network immediately execute a random number of queries (fixed to a value between 0 and 9). Moreover, it has to be considered that, when the peers in the system reach the saturation, they do not make any more queries, as they are sharing most of the files. This effect is mitigated by the overlay network dynamism, since this last introduces a replacement process of peers: in fact, while saturated peers dead, new and not saturated peers enter the system.

Some results versus file popularity

In this first simulation session, we are interested in studying the P2P system performance with respect to file popularity. It is well known that the number of queries issued to request a specific file is directly connected to its popularity. Thus, as the popularity has been assigned to every file according to Zipf distribution, it is reasonable for both the proposed tools that also the number of queries issued during a simulation has somehow to follow the Zipf popularity distribution.

Despite of all these considerations, Figure 1 reports a quite different behavior for both the Fast and the Detailed simulators. In fact, in such kind of environment (small network size, simulation time equal to 24 weeks and query inter-arrival time to 30 sec), the most popular files are spread in the whole network, reaching almost the large part of peers. In this way, after a short initial transient the most popular files become so replicated in the network to be the less requested ones (it is the case of "saturated" overlay network), since a large part of peers already has these specific files. With reference again to Figure 1, the Detailed simulator appears to reach faster the saturation than the Fast one. This is caused by the fact that while the first one allows each peer to issue a successful request for every file just once, with the Fast simulator the same content can be repeatedly requested and obtained by a peer until it is finally included in the list of shared files.

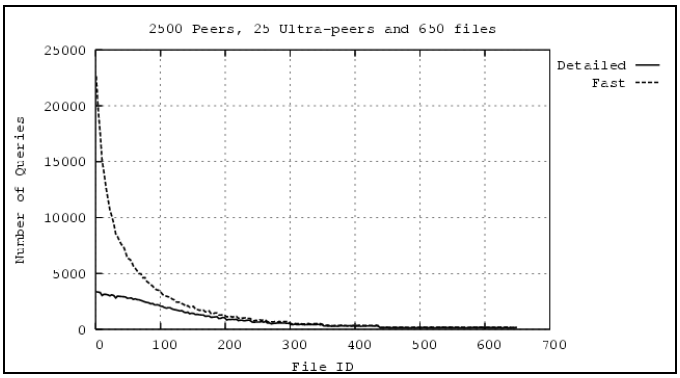

Figure 1. Number of Queries behavior versus popularity, at the end of the simulative process.

Moreover, since popularity is assigned to every file according to Zipf distribution and since it is directly correlated with the probability of requesting a file, it is reasonable to expect a similar behavior also for the number of replicas at the end of a simulation.

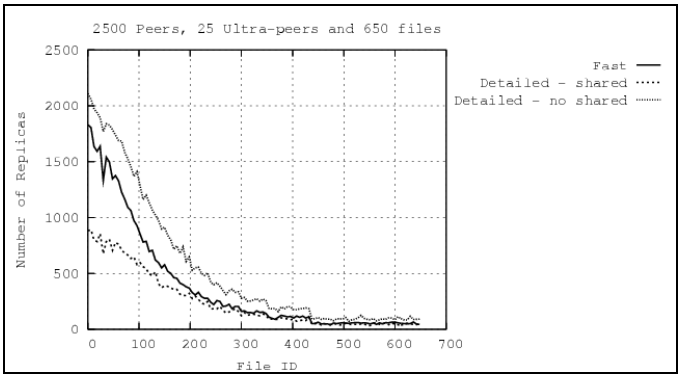

Figure 2. Number of Replicas behavior versus popularity (the difference in the number of replicas for each file, between the beginning and the end of the simulative process).

In Figure 2 the number of replicas versus popularity is reported for both the simulators; in the case of Detailed simulator both the numbers of shared and not-shared files are plotted (in the Fast simulation just the number of shared files exists). Note that, in order to underline the behavior of the file replicas growth, the difference in the number of replicas for each file, between the beginning and the end of the simulative process, is plotted.

Two important aspects can be considered:

- the sum of the shared and not-shared files, in the Detailed simulator, is a bit more of the total number of peers in the system, due to the overlay network dynamism;
- the number of shared files (for the most popular one) in the Fast simulator, is up to the 75% of the number of peers, which is the portion of peers that effectively shares some files (remind that the 25% of the number of peers are free-riders and does not share any file).

Some results versus simulation time

The following figures show some of the obtained results concerning the query process performance. The graphs clearly report that the overlay network reachs a saturation state, for the most popular files, in about 24 weeks; this means that the most popular file is kept by every peer, and therefore that the number of queries for that file keeps close to zero.

The number of queries, the number of replicas, the number of contacted ultra-peers and the number of found sources for some files (with different values of popularity) are plotted versus the (simulated) time.

As previously described, the query process of Gnutella is substantially composed by two different steps: in the first one, the ultra-peer receiving the query forwards the request to its neighbor peers and ultra-peers; if the query does not return any positive results, the request is forwarded to the neighbors of the neighbor ones. The Figure 8 clearly shows that, for less popular files, the number of visited ultra-peers has a relevant variance. This demonstrates that, if the requested file does not

appear in the cluster, a second query hop is needed.

Some differences between the Fast and the Detailed simulator can be observed: the number of performed queries is larger in the Fast simulator than in the Detailed one; considering the most popular file (ID 0), after 40 days, there is a difference of about 100 received queries. A similar gap can be found for the number of replicas in the system, after 40 days, the most popular file has about 900 replicas in the Detailed simulation and 1800 replicas in the Fast one. In the reported figures (Figure 3 and Figure 4, Figure 5 and Figure 6) the behaviors for the Fast and the Detailed are very similar, there is just a difference in the numbers, according to the features of the Detailed simulator, which performes a minor number of queries with respect to the Fast one (which does not keep memory about the not-shared files).

With reference to Figure 8 and Figure 9, a not-linear and interesting behavior is present in the time period before the 40th day; we suppose this behavior can derive from a discontinuity in the overlay network evolution, in terms of an unbalanced number of neighbors for some ultra-peers. We suppose that, at the 40th day, a critical ultra-peer birth or death happens, which changes the network meshing in a way that every query can satisfied in just one hop. Note that, for this overlay network size, the average number of ultra-peers is 25, which is just the amount of visited ultra-peers for each following query, as reported in Figure 8. After the step, the overlay network seems to appear in a sort of saturation state; the most popular file is kept by each peer, the few queries, performed by the new born peers, can find every replica of the requested file (after the 40th day the number of found sources in Figure 7 closely follows the number of replicas in Figure 5). This aspect is supported considering the static simulator (Figure 9), in which births and deaths are not performed; the number of found sources for the most popular file breaks down after about 50 days, giving reason to the fact that no more queries are performed by the existing peers.

These discontinuities can be considered a peculiar, and not so important, effect of small size networks, while wider overlay networks don't present such behaviors, as it can be verified in the following sections.

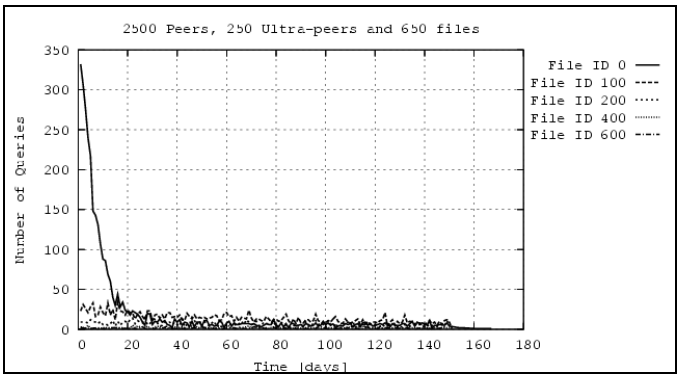

Figure 3. Number of Queries behavior (for some sample files) during the simulative process, for the Detailed simulator.

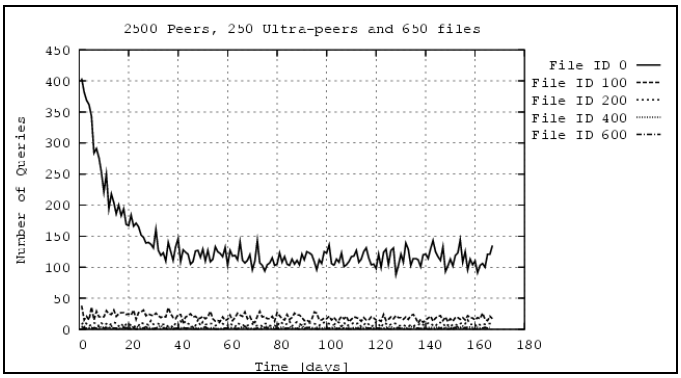

Figure 4. Number of Queries behavior (for some sample files) during the simulative process, for the Fast simulator.

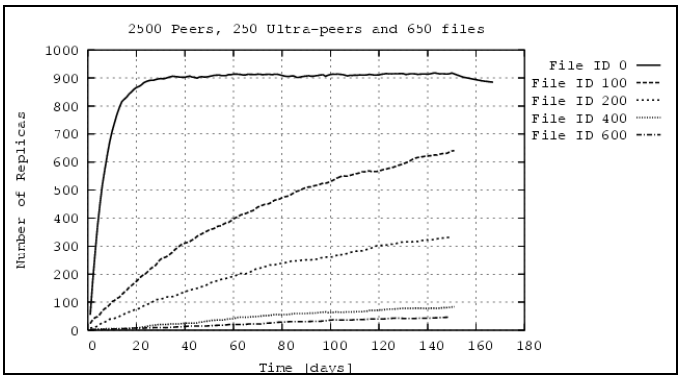

Figure 5. Number of Replicas behavior (for some sample files) during the simulative process, for the Detailed simulator.

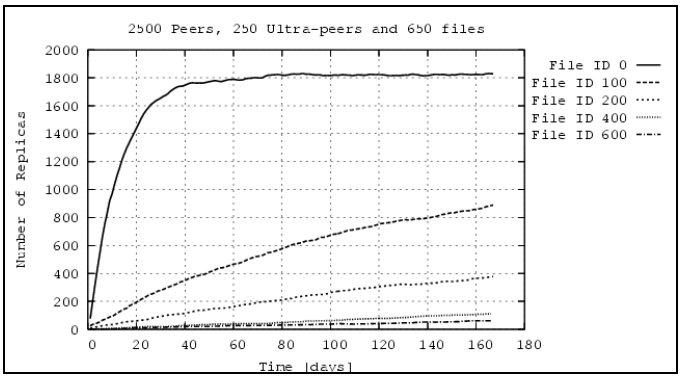

Figure 6. Number of Replicas behavior (for some sample files) during the simulative process, for the Fast simulator.

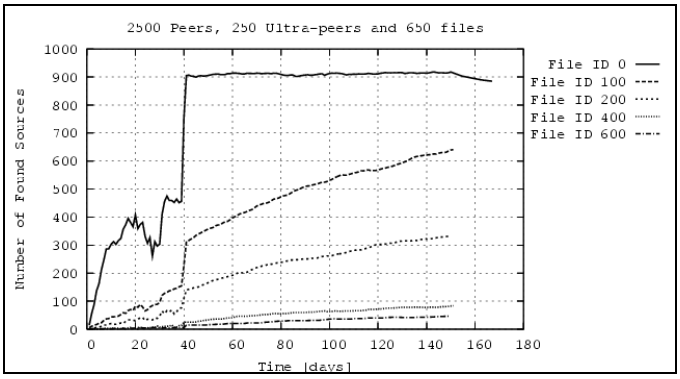

Figure 7. Number of Found sources behavior (for some sample files) during the simulative process, for the Detailed simulator.

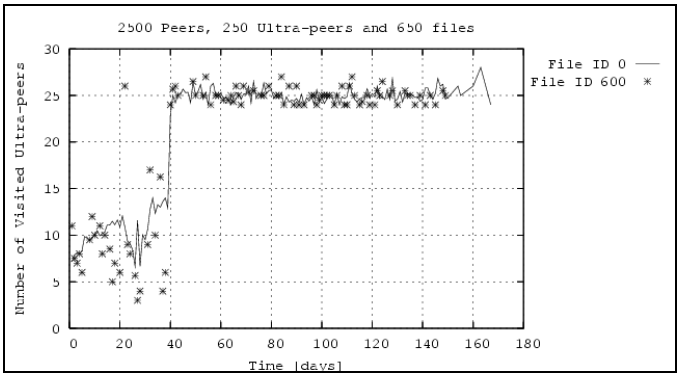

Figure 8. Number of Visited ultra-peers behavior (for some sample files) during the simulative process, for the Detailed simulator.

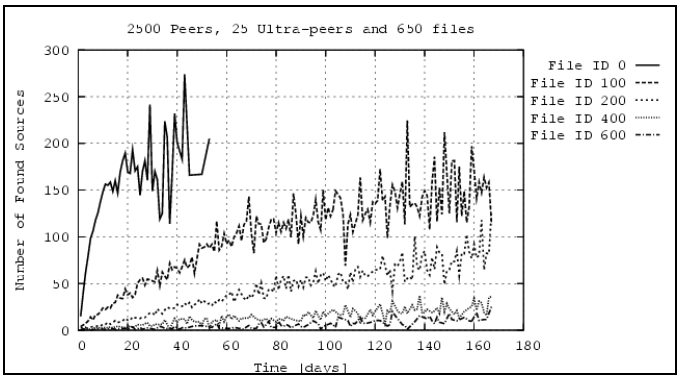

Figure 9. Number of Found sources behavior (for some sample files) during the simulative process, for the Detailed simulator, in the STATIC case.

C. The Medium Size Overlay Network case (25000 peers)

In this case, we report the numerical results of a wider overlay network, with 25000 peers. The mesh index has been fixed equal to 0.06, in other words we have imposed an average number of 15 neighbors for every ultra-peer. Starting from 6500 files, the total number of replicas results equal to 483283 at the beginning. In addition, we impose for the 5% of peers, which have large shares, a number of files between 100 and 150. The simulated time corresponds to 96 weeks, which do not seem to be enough for reaching a saturation state, as reported in Figure 13. It is important to consider the linear behaviors in the number of found sources and visited ultra-peers during the simulation; with respect to the previous overlay network (2500 peers) some discontinuities, typical of small overlay networks, are no more present.

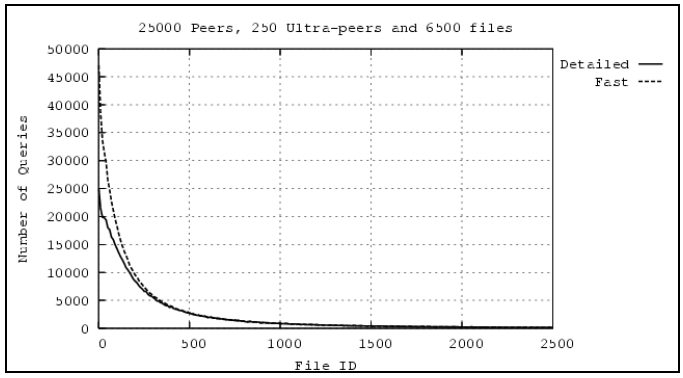

Figure 10. Number of Queries behavior versus popularity, at the end of the simulative process.

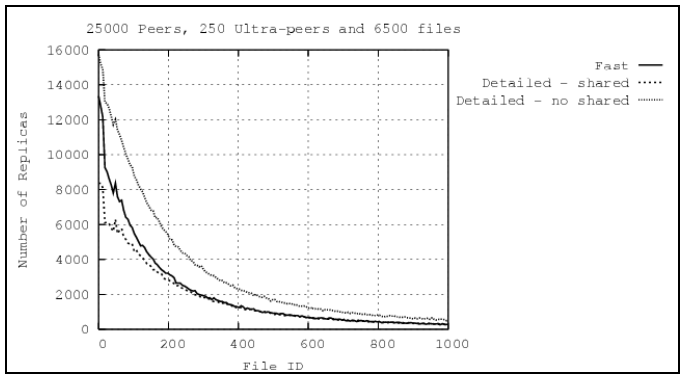

Figure 11. Number of Replicas behavior versus popularity (the difference in the number of replicas for each file, between the beginning and the end of the simulative process).

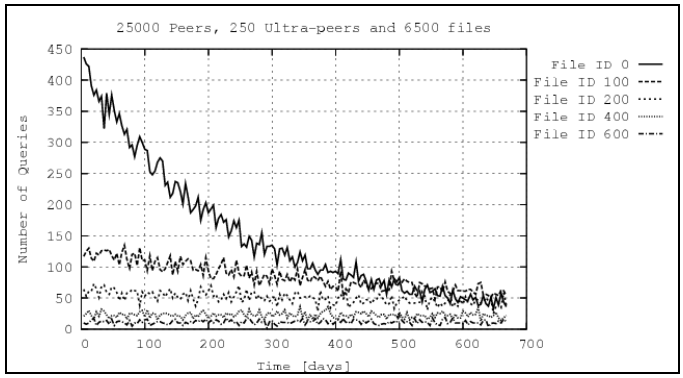

Figure 12. Number of Queries behavior (for some sample files) during the simulative process, for the Detailed simulator.

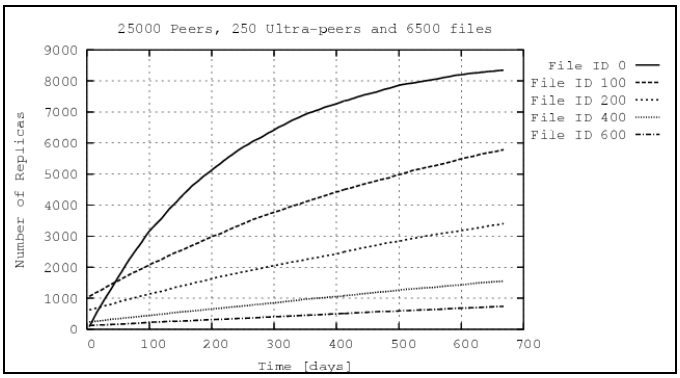

Figure 13. Number of Replicas behavior (for some sample files) during the simulative process, for the Detailed simulator.

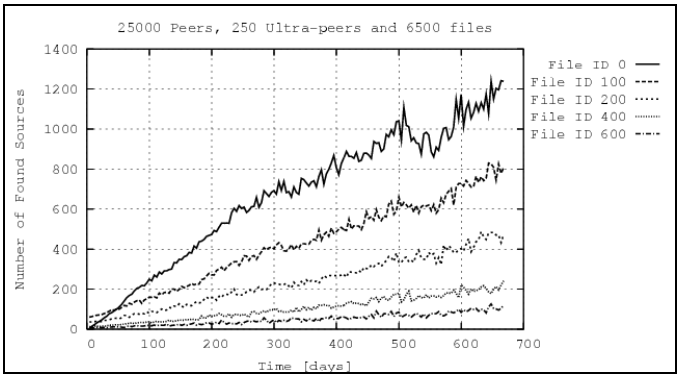

Figure 14. Number of Found sources behavior (for some sample files) during the simulative process, for the Detailed simulator.

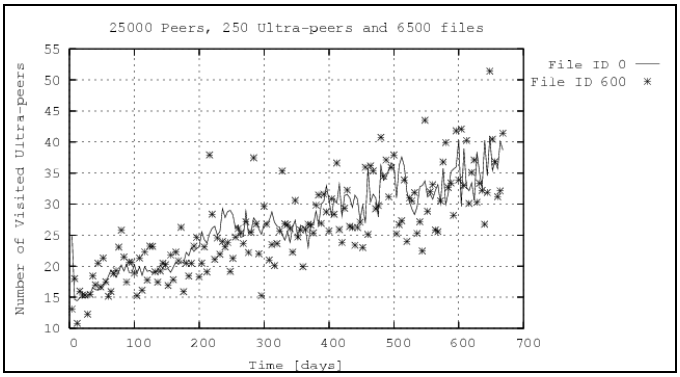

Figure 15. Number of Visited ultra-peers behavior (for some sample files) during the simulative process, for the Detailed simulator.

D. Wide Overlay Networks (250000 peers)

In this section we want to present results about wide overlay networks, in terms of peers, ultra-peers and files. We have simulated a network with 250000, which is a size not so far from reality, considering that the actual Gnutella network size is about 2000000 peers, and that 250000 peers was the total number of peers at May 2004 [1].

The mesh index has been fixed again to 0.06 (15 neighbors for every ultra-peer in average). Starting from 100000 files, the total number of replicas results equal to 30224244. In addition, we set an upper bound of 1600 files to be assigned to the 5% of peers, which have large shares. The simulation time is equal to 192 weeks, we decided to perform such a so long simulation in the hope of finding the saturation state, even if graphs show that, at the end of the simulation, the steady state, also in respect to the most popular files, is far yet.

In the following figures the behavior of the number of queries and the number of replicas is reported in the function of popularity; after, the same results previously shown in the case of smaller size networks are reported.

We want to highlight that large differences with respect to smaller overlay networks, are not present. Increasing the network size, the simulator seems to be coherent and to fit well the expected behaviors. Saturation effects are obviously not preeminent, in wide overlay networks, the file displacement is not so dense thus, the number of performed queries, in the Fast and Detailed simulators, have very close behaviors (Figure 16).

Note that the number of replicas in the Fast simulator is very close to the number of shared files in the Detailed one; as we could expect, growing the overlay network size, the gap among the curves decreases (Figure 2, Figure 11 and Figure 17). In fact, in the Fast simulator, the probability that the same peer queries an already requested file, decreases with the increasing of the number of files and peers in the system.

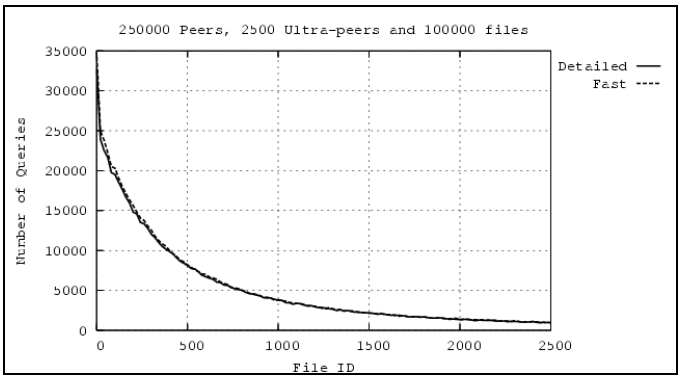

Figure 16. Number of Queries behavior versus popularity, at the end of the simulative process.

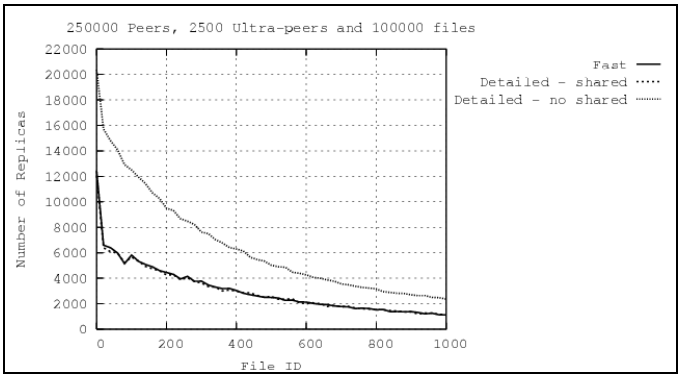

Figure 17. Number of Replicas behavior versus popularity (the difference in the number of replicas for each file, between the beginning and the end of the simulative process).

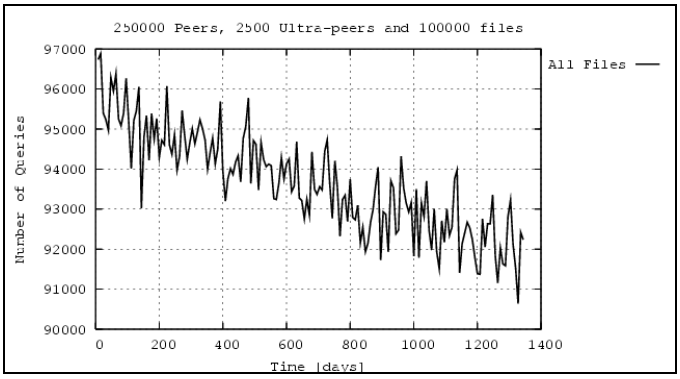

Figure 18. Number of Total Queries behavior during the simulative process, for the Detailed simulator.

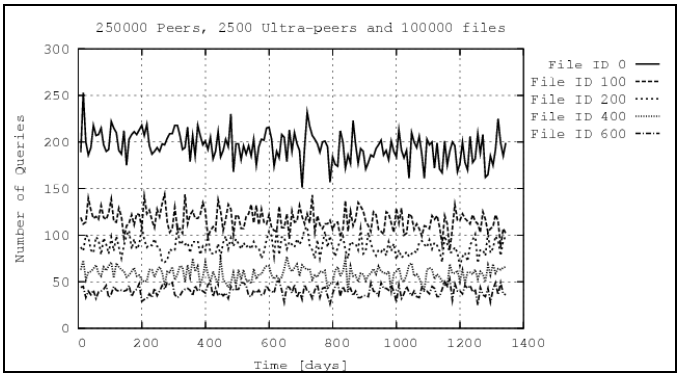

Figure 19. Number of Queries behavior (for some sample files) during the simulative process, for the Detailed simulator.

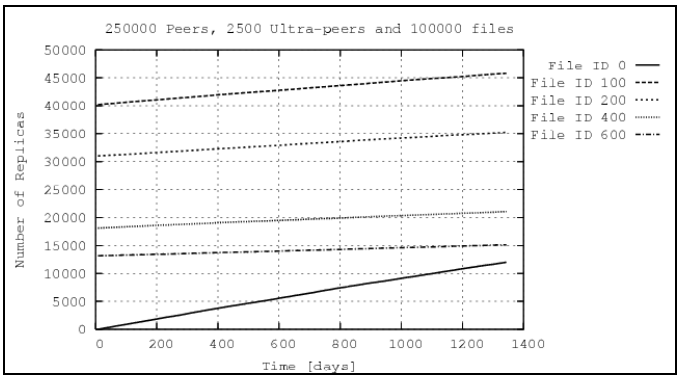

Figure 20. Number of Replicas behavior (for some sample files) during the simulative process, for the Detailed simulator.

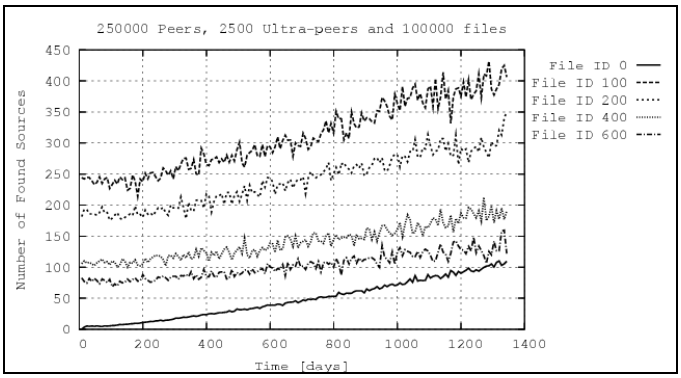

Figure 21. Number of Found sources behavior (for some sample files) during the simulative process, for the Detailed simulator.

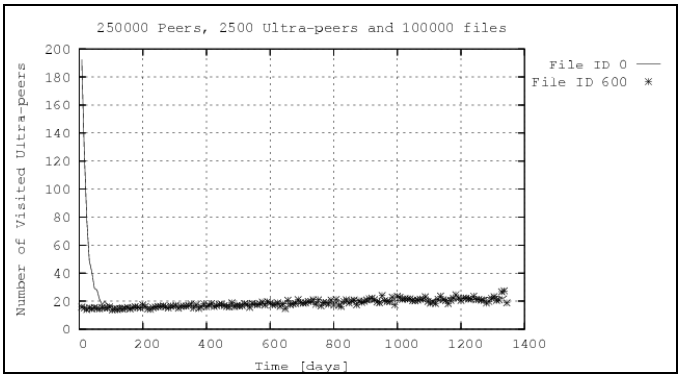

Figure 22. Number of Visited ultra-peers behavior (for some sample files) during the simulative process, for the Detailed simulator.

E. What about scalability?

Finally, we report the computational time (Figure 23) and the memory utilization (Figure 24) values needed by the Fast and the Detailed simulators in dynamic environments. Both the simulations have been run with different overlay network sizes and, for each size, in the same initial conditions (e.g., the same overlay networks). The considered simulation time has been fixed to 10 weeks.

It is clear that developed simulators report very close behaviors for what concerns scalability issues; both the simulators give good results also in very wide overlay networks, such as the one with 250000 peers. Indeed we have noticed that the Detailed simulator, initially developed just for validation reasons, results to fit better the waited results, not increasing the computational time.

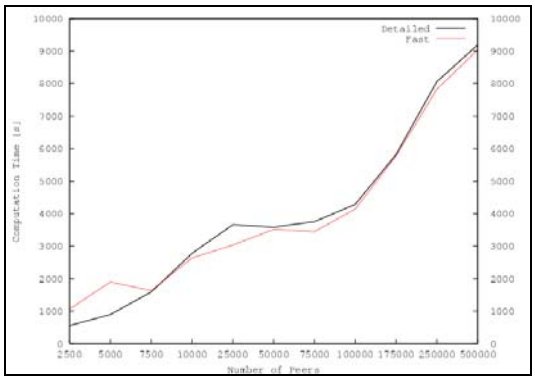

Figure 23. CPU (Intel Xeon, 3 Ghz) Utilization during the simulative process

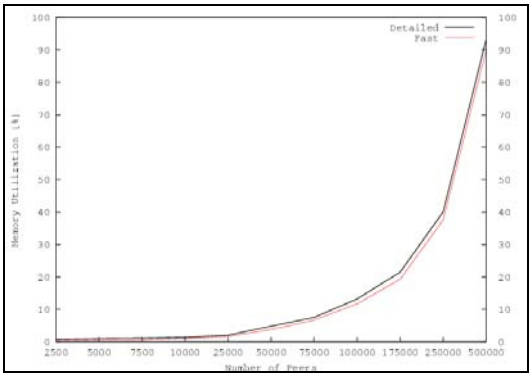

Figure 24. RAM (2 GBytes) Utilization during the simulative process

IV. CONCLUSIONS

The proposed paper has presented a P2P traffic simulation model and it has described some results obtained by Telecommunication Networks and Telematics (TNT) Laboratory of DIST with this tool.

In particular, for what concerns the simulation model development, two models (called Fast and Detailed) with different detail levels have been proposed and analyzed; then, a set of results, about the application layer performances and regarding different network environments, has been shown and discussed. We can say that the developed simulators result very scalable, giving good results also in very wide overlay networks, such as the one with 250000 peers. Indeed we have noticed that the Detailed simulator, initially developed just for validation reasons, results to fit better the waited results, not increasing the computational time; in this way the Detailed one is the best candidate to be integrated with a TCP simulator, in order to study the impact of such P2P traffic at the lower layers.

REFERENCES

[1] slyck.com. http://www.slyck.com, 2005

[2] D. Stutzbach and R. Rejaie. Capturing Accurate Snapshots of the Gnutella Network. In Global Internet Symposium, 2005.

[3] T. Karagiannis, A.Broido, M. Faloutsos, and kc claffy. Transport layer identification of P2P traffic. In ACM/SIGCOMM IMC, 2004.

[4] Matei Ripeanu, Ian Foster, Adriana Iamnitchi. Mapping the Gnutella Network. IEEE Internet Computing, Volume 6, Number 1, January/February 2002.

[5] Stefan Saroiu, P. Krishna Gummadi, Steven D. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In Proceedings of Multimedia Computing and Networking, 2002.

[6] S. Sen and J. Wang, Analyzing Peer-to-Peer Traffic Across Large Networks. In ACM Transactions on Networking, Volume 12, Number 2, April, 2004, pp. 219—232.

[7] Gnutella. http://rfc-gnutella.sourceforge.net/.

[8] eDonkey. http://edonkey2000.com/

[9] eMule. http://www.emule-project.net.

[10] Direct Connect. http://www.indx.f2s.com/.

[11] Kazaa. http://www.kazaa.com.

[12] Louis Plissonneau, Jean-Laurent Costeux, Patrick Brown. Analysis of Peer-to-Peer Traffic on ADSL. In PAM, 2005.

[13] Jonathan Hess, Benjamin Poon. Improving Performance in the Gnutella Protocol. Available online at http://www.cs.berkeley.edu/

[14] Murat Karakaya et Al. GnuSim: A General Purpose Simulator for Gnutella and Unstructured P2P Networks. Available online at http://www.cs.bilkent.edu.tr

[15] Qi He, Mostafa Ammar, George Riley, Himanshu Raj, Richard Fujimoto. Mapping Peer Behavior to Packet-level Details: A Framework for Packet-level Simulation of Peer-to-Peer Systems. In MASCOTS 2003.

[16] Tobias Hoßfeld, Kenji Leibnitz, Rastin Pries, Kurt Tutschku, Phuoc Tran-Gia, Krzysztof Pawlikowski. Information Diffusion in eDonkey Filesharing Networks. ATNAC 2004, Sydney, Australia, December 2004.

[17] Schlosser, Mario; Condie, Tyson; Kamvar, Sepandar. Simulating A File-Sharing P2P Network. 1st Workshop on Semantics in Grid and P2P Networks.

[18] Raffaele Bolla, Roberto Bruschi, Matteo Repetto. A Fluid Simulator for Aggregate TCP Connections. In Proceedings of SPECTS, 2004.

[19] Daniel Stutzbach, Reza Rejaie. Towards a Better Understanding of Churn in Peer-to-Peer Networks. Technical Report CIS-TR-04-06, University of Oregon, November 2004.