BEHAVIOR CHARACTERIZATION AND PERFORMANCE EVALUATION OF A HYBRID P2P FILE SHARING SYSTEM

Juan Pedro Muñoz-Gea, Josemaria Malgosa-Sanahuja, Pilar Manzanares-Lopez
Juan Carlos Sanchez-Aarnoutse and Joan Garcia-Haro
Department of Information Technologies and Communications
Polytechnic University of Cartagena, Campus Muralla del Mar, 30202, Cartagena, Spain

Keywords: Peer-to-Peer systems, simulation tools, analytical models, performance evaluation.

Abstract: Peer-to-Peer (P2P) networks show a set of distinctive features which increase the need to previously simulate the new proposals. In order to perform an adequate evaluation of a P2P application, it is necessary a real characterization of the queries behavior and the node dynamism. After validating the new P2P algorithm by means of simulation, developers have the possibility to prove a real P2P application instance in an emulated environment. In each step, the detected errors or misbehaviors are appropriately corrected. The main contribution of this paper is twofold: first, to adequately characterize the real behavior of P2P overlay networks (including dynamic and static aspects) and second, evaluate a real P2P overlay network under the above constraints using one of the most popular (high level and user friendly) simulation frameworks.

1 INTRODUCTION

The cost associated with the implementation of a network application is too large in most cases. Therefore, to avoid a costly failure, it is recommended to simulate the proposed algorithms and specifications before. This methodology makes possible both to correct any kind of mistake and to validate the final design. In addition, peer-to-peer (P2P) networks (Steinmetz and Wehrle, 2005) show a set of distinctive features which increase the need to previously simulate the new proposals. First of all, these applications involve a massive number of nodes (1,000,000 or more), and they require to operate in large-scale environments, i.e. on Internet. Therefore, scalability is one of the most important characteristics that researchers have to prove. Secondly, P2P nodes are usually home computers. Therefore, they are prone to unpredictable disconnections and consequently, developers have to ensure that the new applications can appropriately adapt to the node failures and the continuous joining and leaving of nodes inherent to the P2P nets. Hence, node dynamism is another key factor to take into consideration.

On the other hand, in order to perform an adequate performance evaluation of a P2P application, it is necessary to program a realistic model of the system. Therefore, it is necessary to characterize the queries behavior, the node connection interarrival time, the distribution of the session length and the dynamism of the files.

Ideally, a simulator should implement all the mechanisms and procedures of the real algorithm/application, but coded in a completely different way: for instance, a P2P application may need to use the system call *socket()* or create a new thread, but usually a simulator of this application does not need to use this type of procedures to simulate the behavior of the application. On the other hand, normally a simulated application is a characterization of the real system and thus, more lightweighted than its corresponding actual software, because a real application is tied to heavy operating system resources such as processes and threads. In conclusion, the simulated experiments are developed in a complete different way to the real implementation.

After validating the new (under study) P2P algorithm, developers have the possibility to prove a real P2P application instance (natural programming) in an emulated environment. A network emulator is a program running in a set of computers (at least one) that appears to be a network; applications can be attached to the emulator and will behave as they are attached to a real network. The behavior of an emulated envi-

Pedro Muñoz-Gea J., Malgosa-Sanahuja J., Manzanares-Lopez P., Carlos Sanchez-Aarnoutse J. and Garcia-Haro J. (2008).
BEHAVIOR CHARACTERIZATION AND PERFORMANCE EVALUATION OF A HYBRID P2P FILE SHARING SYSTEM.
In *Proceedings of the Third International Conference on Software and Data Technologies - PL/DPS/KE*, pages 95-102
DOI: 10.5220/0001870300950102

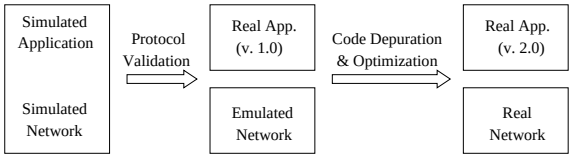

Figure 1: Procedure to develop an error-free network application.

ronment is exactly the same as the real world. This mechanism enables developers to test and improve their algorithms with a large number of nodes on a single computer or on few computers in a small lab (in that case the control messages are encapsulated and forwarded via UDP or TCP sockets between the computers).

Fig. 1 graphically summarizes the above discussion. Generally, the procedure to launch a new network application is first to simulate it, next to prove it in an emulated environment, and finally to issue the application to the market. In each step, the detected errors or misbehaviors are appropriately corrected.

Some examples of specific P2P simulation frameworks are P2PSim (P2PSim, 2005) and OverSim (OverSim, 2007), and some of the most popular network emulators are ModelNet (ModelNet, 2005) and Overlay Weaver (Overlay Weaver, 2007). Simulation tools can operate with approximately ten thousands of nodes. Emulators work only with hundreds of nodes on a single PC.

The main contribution of this paper is twofold: First, to adequately characterize the real behavior of P2P overlay networks (including dynamic and static aspects) and second, evaluate a P2P overlay network designed and published by the authors (Muñoz-Gea et al., 2006) under the above constraints using one of the most popular (high level and user friendly) simulation framework. The results reported by the simulation model are evaluated in order to verify its correctness.

The remainder of the paper is organized as follows. Section 2 summarizes the most relevant properties of the well-known OMNeT++ simulation engine and the OverSim simulation framework. Section 3 briefly describes the most recent P2P applications models. Section 4 describes the proposed P2P simulation model, while the corresponding numerical results are discussed in Section 5. Finally, Section 6 concludes the paper.

2 P2P SIMULATION FRAMEWORK

We find interesting to remark and briefly describe the most relevant open-source P2P simulation frameworks available (a comprehensive survey of P2P simulation frameworks can be found in (Naicken et al., 2007)). P2PSim (P2PSim, 2005) is a discrete event simulator for structured P2P networks written in C++. However, it is largely undocumented and therefore hard to extend. Additionally, it has a limited set of statistics. PeerSim (PeerSim, 2006) uses query-cycle or discrete-event techniques to simulate unstructured P2P networks. However, the underlying TCP/IP network is not modeled and only the query-cycle simulator is documented. PlanetSim (PlanetSim, 2005) is a discrete event simulator written in Java. However, it has a limited simulation of the underlying TCP/IP network and it does not provide any mechanism to gather statistics.

Among all the available tools we have chosen OverSim because of its flexibility: Its modular design and the use of the Common API (Dabek et al., 2003) facilitate its extension with new protocols. In addition, its flexible underlying network scheme can be very useful during the application development. Another reason to select OverSim is that its code is well documented.

OverSim is built over OMNeT++. OMNeT++ is a well-known open-source network simulation engine and it is one of the most used simulation frameworks for the research community. Additionally, its coding procedure is well documented and it also offers a multiplicity of network simulation models already implemented.

The development of a simulation model with OMNeT++ has several steps, which we briefly point out:

1. Description of the system structure using the NED language.
2. Implementation of the simple modules in C++. The node functionality must be implemented in the *handleMessage()* method.
3. To obtain an executable, the modules have to be compiled and linked with the simulation library.
4. Configuration of the omnetpp.ini file.

The layered structure of OverSim appears in (Baumgart et al., 2007). It includes all the facilities about underlay and overlay networks and its primitives. Therefore, users only need to code the overlay application in the application layer.

The communication between the overlay layer and the application layer uses the API described and well

documented in (Dabek et al., 2003). The set of functions defined in this API are the same independently of the overlay network selected (Chord, Kademlia, GIA). Therefore, the same overlay application can be easily translated from one overlay system to another with a minimum effort.

However, although OverSim is a new overlay network simulation framework, developed to overcome several drawbacks of existing peer-to-peer simulators, it is necessary to introduce appropriate models to program the behavior of these applications in OverSim. Next section describes the most recent models for this behavior.

3 P2P MODELLING

In order to perform an adequate evaluation of a P2P application, it is necessary to program a realistic model of the system. Therefore, it is necessary to characterize the queries, the node connection inter-arrival time, the distribution of the session length, the files life-time, the files arrival rate and the number of peers receiving them. Next, we are going to briefly describe the most recent models for the previous parameters.

3.1 Queries Characterization

An accurate characterization of peer query behavior is absolutely indispensable. This characterization includes the time until the first query, the query interarrival time and the query popularity. In (Klemm et al., 2004a) the authors provide a model that is based on passive measurement of a peer in the Gnutella P2P system over a period of 40 days.

Specifically, the time until the first query can be modeled by a bimodal distribution composed of a Weibull distribution for the body and a log-normal distribution for the tail. In the same way, the query interarrival time is modeled by a bimodal distribution composed of a log-normal distribution for the body and a Pareto distribution for the tail.

On the other hand, the query popularity follows a Zipf-like distribution. Zipf's law states that the size of the rth largest occurrence of an event is inversely proportial to its rank r:

$$f_r \approx 1/r^{\alpha} (r = 1,2,3,...,N) \qquad (1)$$

where α is close to unity, and N is the number of distinct occurrences of an event. Under a Zipf-like distribution, the probability of requesting the ith most popular document is given by:

$$p_i = \frac{1}{K \cdot i^{\alpha}} \ \textit{where}, \ K = \sum_{j=1}^{N} 1/j^{\alpha} \qquad (2)$$

3.2 Dynamics of Peer Participation

The dynamics of peer participation, or *churn*, are an inherent and critical property of P2P systems. A peer joins the system when a user starts the application, contributes with some resources while making use of the resources provided by others, and leaves the system when the user exits the application. This join-participate-leave cycle is defined as a *session*. Thousands of independent sessions creates the collective effect called *churn*.

The two fundamental properties of churn are the following: (i) the inter-arrival time between sessions and (ii) the session length. Several simulation and analytical studies have typically assumed both distribution to be exponential, though some studies have modeled the session length distribution as Pareto. In (Stutzbach and Rejaie, 2006) authors found other distributions for inter-arrival time and session length.

Exponential distributions are typically used to model behavior resulting from a large number of independents events. However, peer arrivals are not completely independent. Users are less likely to be active during certain times of day (or during certain days of the week), and a surge of arrivals may occur when a link to a file appears on a popular website. In this scenario, Weibull distributions are a more flexible alternative to exponential distributions (in fact, exponential distributions are a special case of the Weibull distributions where the shape parameter is 1). Therefore, for inter-arrival times, a Weibull distribution provides a much better fit.

In the same way, session lengths do not follow the commonly-used exponential distribution. Several studies have reported that peer sessions lengths are heavy-tailed. However, neither the exponential nor heavy-tailed distributions proved consistent with recent observations. There are other two models that might provide a good fit: log-normal distributions and Weibull distributions. The reason is that while most session are short (minutes), some sessions are very long (days or weeks). This differs from exponential distributions, which exhibit values closer together in length, and heavy-tailed distributions, which have more pronounced extremes (years).

3.3 Files Characterization

Characterizing available files among participants peers is valuable because it reveals the properties, distribution and heterogeneity of the contributed re-

sources (i.e. storage space and available files) by individuals users in the system. The reference (Stutzbach et al., 2007) is the first study that has explored the dynamic characteristics of stored files in P2P systems. Among them, we can find the variations in the shared files by individual peers. There are two types of changes that may occur in the list of shared files at each peer: First, the user may add new files, either by downloading them from other peers or by manually adding them to the shared folder. Second, the user may remove files, by moving (or deleting) those files from the sharing folder. The total number of added and deleted files at a single peer is defined as the *degree of change*.

On the other hand, file replication can be modeled by a Zipf-like distribution. In (Klemm et al., 2004b) authors relate query popularity and file replication by characterizing the matching probability between queries and files. They showed that this matching probability is given by a probability function composed by a summation of an exponential and a normal distributions. The probability function depends on the popularity rank of the query, but it is independent of the replication rank of the file. Thus, the matching probability between queries and files is easily computable.

4 MODEL DESCRIPTION

4.1 Introduction

In (Muñoz-Gea et al., 2006) the authors proposed an unstructured overlay network which uses a structured lookup procedure to organize the peers, forming a hybrid P2P network. This network was evaluated without considering the requirements explained in Section 3 and using a custom simulator implemented in C language. Now, this network will be evaluated programming a realistic model of the system and using one of the most popular (high level and user friendly) simulation frameworks. First, subsection 4.2 summarizes the overlay network architecture. Second, its implementation in the OverSim simulator is briefly presented in subsection 4.3. Finally, the results reported by the simulation model will be evaluated in order to verify its correctness.

4.2 Overlay Network Architecture

We took (Muñoz-Gea et al., 2006) as reference protocol. In this proposal, peers are grouped into subgroups in an unstructured way. However, this topology is maintained by means of an structured lookup mechanism.

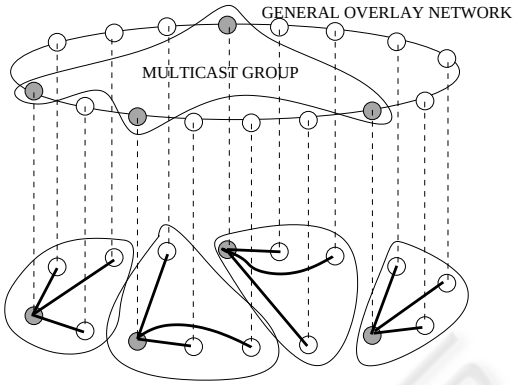

Figure 2: General architecture of the system.

Every subgroup is managed by one of the subgroup members, that we call super-peer. The super-peer is the node that has the best features in terms of CPU, bandwidth and reliability. When searching for a content, the user will send the lookup parameters to its local super-peer. The local super-peer will resolve the complex query, obtaining the results of this query.

To allow the super-peers to locate contents placed in remote subgroups, all the super-peers of the network are going to be members of a multicast group. The structured P2P network can be also used to implement an Application Level Multicast (ALM) service. Among all the possibilites, we have selected Chord-multicast (Stoica et al., 2003)(El-Ansary et al., 2003) as mechanism to send messages to all the members of the multicast group.

To implement the structured-based maintenance of the unstructured topology, all the nodes (peers) must be immersed into a a Chord structured network. Therefore, every node has an identifier (*NodeID*), and they have to contact with an existing node to join this network. Figure 2 describes the general architecture of the system.

The previous existing node also gives to the new node the identifier of the subgroup it belongs (*SubgroupID*), and using the structured lookup mechanism the new node will find the super-peer of that subgroup. This is possible because each time a node becomes a super-peer, it must contact with the node which *NodeID* fits with the *SubgroupID* and sends it its own IP address.

Initially, the new node will try to connect the same subgroup than the existing node. However, if there is no room, the new node will be asked to create a new (randomly generated) subgroup or it will be asked to join the subgroup that the resquested super-peer urged to create previously. The use of different ex-

isting nodes allows the system to fill up incomplete subgroups.

When a new node finds its super-peer, it notifies its resources of bandwidth and CPU. Thus, the super-peer forms an ordered list of future super-peer candidates: the longer a node remains connected (and the better resources it has), the better candidate it becomes. This list is transmitted to all the members of the subgroup.

4.3 Simulator Implementation

In this section the most relevant steps to configure a simulation in OverSim are discussed. First, it is necessary to implement the support for the joins and departures of the DHT service used by the applicaton software. As mentioned above, in OverSim the communication between the overlay and the application layers corresponds to the API presented in (Dabek et al., 2003). Specifically, the upcall *update(nodehandle n, bool joined)* is invoked by the overlay to inform the application that node *n* has either joined or left the neighbor set. In order to implement the mechanism that this function has to perform in our application, we have to include and code accordingly this function. Since our application uses the Chord overlay network, the actions performed by our *update()* function only have to take into account when a new node joins the network between the current predecessor node and itself (remember that Chord nodes are virtually organized in a ring). This is because in this case, the keys between the current predecessor and the new node have to be sent to the new node. These keys are delivered encapsulated in a message defined in our application and called *AppChurnMessage*. In addition, when a node leaves the network it has to send its keys to the first succesor node.

On the other hand, our application uses the *SubgroupID* as key and the IP of the super-peer as value, and both of them are organized in a DHT over a Chord structured network. In order to appropriately implement this functionality, our application must use the routing functions defined in the Common API (Dabek et al., 2003). These routing functions are the following:

- *void route(key k, msg m)*. This operation forwards a message, *m*, towards the responsible node of key *k*. This function is implemented in the overlay layer. The application layer offers the *callRoute(key k, msg m)* function in order to call the route-method in the overlay-layer.
- *void forward(key k, msg m, nodehandle nexthopnode)*. This function is implemented in the application layer. It is invoked from the overlay layer of each node that forwards message *m* during its routing.
- *void deliver(key k, msg m)*. This function is also implemented in the application layer. It is invoked from the overlay layer of the node that is responsible for key *k* upon the arrival of message *m*.

When the overlay layer facilities supply the IP addresses of the peers in our application, the communication between nodes is established by means of a UDP/IP socket. This activity requires two actions, first of all, nodes have to bind to a local port in order to allow the reception of messages (*bindToPort()* function); then, in order to send messages to a specific destination node (destAddr, destPort) we will use the *sendToUDP()* function.

The rest of the implementation is in the *handleMessage()* function. As already mentioned, this function is invoked when the module receives a message, and it has to implement the most important functionality of the simulator.

Regarding characterization of peer query behavior, we have followed the models presented in (Klemm et al., 2004a). Therefore, the time until the first query is modeled by a bimodal distribution composed of a Weibull distribution for the body, with parameters $\alpha = 0.9821$ and $\lambda = 0.02662$ and a log-normal distribution for the tail, with parameters $\sigma = 2.359$ and $\mu = 6.301$.

In the same way, the query interarrival time is modeled by a bimodal distribution composed of a log-normal distribution for the body, with parameters $\sigma = 1.625$ and $\mu = 3.353$, and a Pareto distribution for the tail, with parameters $\alpha = 0.90411$ and $\beta = 103$.

On the other hand, the query popularity follows a Zipf-like distribution with $\alpha = 1$ and $N = 5$. This is because the available contents have been divided in 5 different popularities.

For inter-arrival times, a Weibull distribution with scale parameter $k = 0.62$ is used, as it was presented in (Stutzbach and Rejaie, 2006). In the same way, the session length is modelled with a Weibull distribution with shape and scale paramters $k = 0.59$ and $\lambda = 41.9$. These parameters are introduced in a OverSim facility (ChurnGenerator) which manages the peers birth and death processes automatically.

Finally, regarding of the dynamic characteristics of stored files, (Stutzbach et al., 2007) is the first study that has explored it. However, it has not been implemented in our simulation model. The reason is that the previous work depicts the evaluated CDF for this characteristic but it does not present an analytical model to represent it.

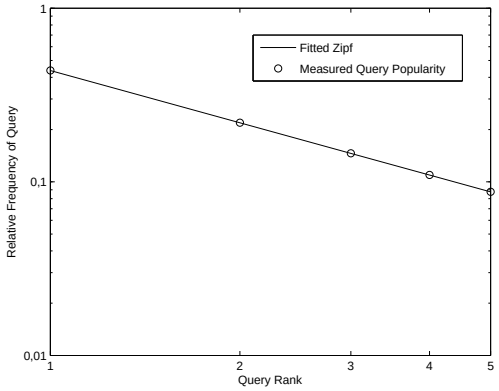

Figure 3: Query popularity.

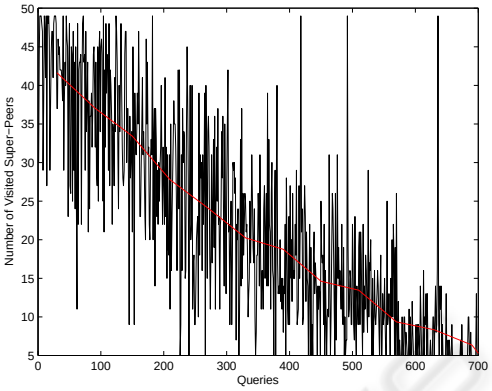

Figure 5: Number of visited super-peers.

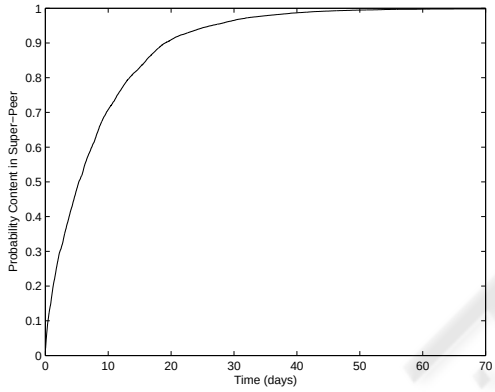

Figure 4: Probability that the content is in the super-peer.

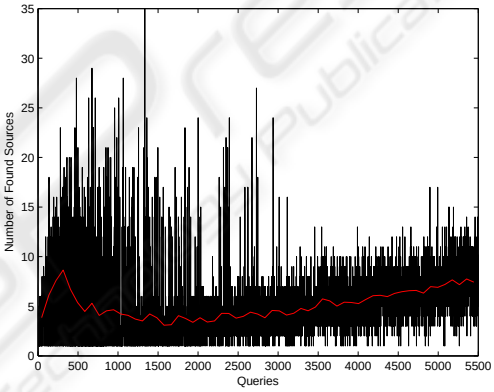

Figure 6: Number of found sources.

5 SIMULATION MODEL RESULTS

In this section the simulation results are presented. To better understand the performances, the results have been divided in two scenarios: In the first one, called static, the peers never die. In the second one, called dynamic, the peer birth and death processes are conveniently modeled.

The parameters of the network are: 2,500 peers, 25,000 files and 50 super-peers. At the end of the section, in order to characterize the model scalability, considerations about simulation time, computational and memory resource requests will be reported.

5.1 Static Overlay Network Simulations

First, we are interested in studying the P2P system performance with respect to query popularity. It is well known that the number of queries issued to request a specific file is directly connected to its popularity. Due to the popularity has been assigned to every query according to Zipf distribution, it is reasonable that the number of queries issued during a simulation has somehow to follow the Zipf popularity distribution. These results are presented in Fig. 3, which shows (in logarithmic scale) that the number of queries performed in the simulation perfectly matches with a Zipf distribution.

Fig. 4 shows the probability that the requested content is in the same requester's sub-group. It is observed that this probability grows as the time increases, but converging to a value of one, which assures that our system reaches a steady state. This also indicates that with the typical (real) parameters that model the queries behavior our architecture assures that, in one month, the contents will be almost equally distributed among all the sub-groups.

Fig. 5 reports the number of consulted super-peers for the queries that cannot be resolved in the requester's subgroup. In this case, a multicast process

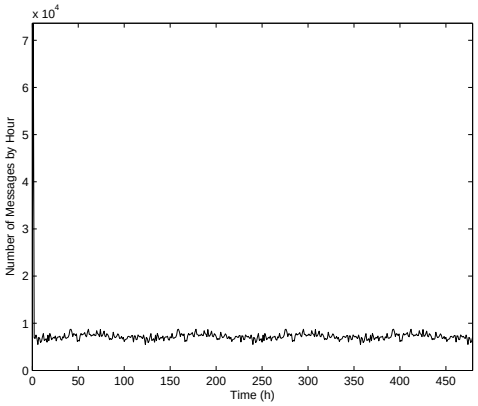

Figure 7: Number of control messages.

starts and tries to send the query to all the super-peers. However, when a super-peers has a reference to a peer which can resolve the query, it immediately stops the multicast process. As it can be seen, the number of consulted super-peers decreases when the number of queries increses, because the proposal is able to efficiently distributed the most popular contents among all the subgroups. This behavior can be viewed more precisely following the time behavior of the average value of this parameter which it is also represented.

Finally Fig. 6 represents, for each query, the number of found peers wich have the requested content. For the first queries, the number of peers increases because the multicast procedure is frequently used. However, when the number of queries is high enough they are resolved in the requester's subgroup. The figure also represents the average value, which shows that this parameters slowly increases and will stabilize around 50 peers when time tends to infinity, because every subgroup is composed by this number of peers.

5.2 Dynamic Overlay Network Simulations

In this simulation session, birth and death events of peers and super-peers are also considered, driving, in this way, the analysis to the case of dynamic overlay networks closer to the reality.

In the dynamic scenario it has been 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 peer joining the network immediately execute a query. 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 satured peers dead, new and non-saturated peers enter the system.

Table 1: Simulation Speed.

Number of nodes	Simul. sec. / Real sec.
500	112.861
1000	46.3345
1500	26.6125
2000	18.9614
3000	10.4441
5000	4.4518
10000	3.1518

Table 2: Memory Consumption.

Number of nodes	Memory [Mb]
500	30.72
1000	51.2
1500	73.72
2000	96.25
3000	141.31
5000	229.37
10000	315.39

Fig. 7 represents the average number of control messages in function of time. These messages are sent in order to update the overlay network architecture when a super-peer dies. In steady state the average number of control messages is under 10,000 messages by hour. Usually, control messages have a small payload (about 10 bytes) and a TCP/IP header (40 bytes). If we suppose each control message has a length of 50 bytes, control traffic supposes only a 1.1 kbps traffic rate.

5.3 Simulator Scalability

In this section we present the OverSim overall performance. Table 1 represents the ratio between the simulated seconds and the real seconds, for different number of nodes. For example, for 500 nodes in one real second OverSim is able to simulate 112.861 seconds. These results have been obtained using a Intel Core 2 architecture at 2.13 GHz with 2 GB of RAM memory. We observe that the simulation speed decreases in an exponential way when the number of nodes increases. This is due to the increment in the number of overlay connections among all the nodes in the network.

Table 2 represents the memory consumption as a function of the number of simulated nodes. In this case, the amount of memory increases linearly with the number of nodes.

6 CONCLUSIONS

The main contribution of this paper is twofold: First, to adequately characterize the real behavior of P2P overlay networks (including dynamic and static aspects) and second, evaluate a hybrid P2P overlay network proposed by the authors under the above constraints using one of the most popular (high level and user friendly) simulation framework.

Among all the available tools we have chosen OverSim because of its flexibility. Its modular design and the use of the Common API facilitate the extension with new protocols. We have also described the most recent models for the characterization of the queries, the dynamics of peer participation and the dynamics characteristics of stored files.

The overall performance figures of the simulation are also presented and discussed. They reveal that the exposed methodology can be easily applied and used by the research community to obtain interesting and useful R&D results.

ACKNOWLEDGEMENTS

This research has been supported by project grant TEC2007-67966-C03-01/TCM (CON-PARTE-1) and it is also developed in the framework of "Programa de Ayudas a Grupos de Excelencia de la Región de Murcia, de la Fundación Séneca, Agencia de Ciencia y Tecnología de la RM (Plan Regional de Ciencia y Tecnología 2007/2010)". Juan Pedro Muñoz-Gea also thanks the Spanish MEC for a FPU (AP2006-01567) pre-doctoral fellowship.

REFERENCES

Baumgart, I., Heep, B., and Krause, S. (2007). Oversim: A flexible overlay network simulation framework. In *Proceedings of 10th IEEE Global Internet Symposium (GI '07) in conjunction with IEEE INFOCOM 2007*, Anchorage, AK, USA.

Dabek, F., Zhao, B., Druschel, P., Kubiatowicz, J., and Stoica, I. (2003). Towards a common api for structured peer-to-peer overlays. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS03)*, Berkeley, CA.

El-Ansary, S., Alima, L. O., Brand, P., and Haridi, S. (2003). Efficient broadcast in structured p2p networks. In *IPTPS*, pages 304–314.

Klemm, A., Lindemann, C., Vernon, M. K., and Waldhorst, O. P. (2004a). Characterizing the query behavior in peer-to-peer file sharing systems. In *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 55–67, New York, NY, USA. ACM.

Klemm, A., Lindemann, C., and Waldhorst, O. P. (2004b). Relating query popularity and file replication in the gnutella peer-to-peer networks. In *Proc. 12th GI/ITG Conf. on Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB)*, Dresden, Germany.

ModelNet (2005). Modelnet. http://modelnet.ucsd.edu.

Muñoz-Gea, J. P., Malgosa-Sanahuja, J., Manzanares-Lopez, P., Sanchez-Aarnoutse, J. C., and Guirado-Puerta, A. M. (2006). A hybrid topology architecture for p2p file sharing systems. In *Proceedings of the 1st International Conference on Software and Data technologies (ICSOFT 2006)*, Setúbal, Portugal.

Naicken, S., Livingston, B., Basu, A., Rodhetbhai, S., Wakeman, I., and Chalmers, D. (2007). The state of peer-to-peer simulators and simulations. *SIGCOMM Comput. Commun. Rev.*, 37(2):95–98.

Overlay Weaver (2007). Overlay weaver: An overlay construction toolkit. http://overlayweaver.sourceforge.net/.

OverSim (2007). The oversim p2p simulator. http://www.oversim.org/.

P2PSim (2005). P2psim: A simulator for peer-to-peer protocols. http://pdos.csail.mit.edu/p2psim/.

PeerSim (2006). Peersim: A peer-to-peer simulator. http://peersim.sourceforge.net/.

PlanetSim (2005). Planetsim: Object oriented simulation framework for overlay networks. http://planet.urv.es/trac/planetsim/.

Steinmetz, R. and Wehrle, K., editors (2005). *Peer-to-Peer Systems and Applications*, volume 3485 of *Lecture Notes in Computer Science*. Springer.

Stoica, I., Morris, R., Liben-Nowell, D., Karger, D. R., Kaashoek, M. F., Dabek, F., and Balakrishnan, H. (2003). Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Trans. Netw.*, 11(1):17–32.

Stutzbach, D. and Rejaie, R. (2006). Understanding churn in peer-to-peer networks. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 189–202, New York, NY, USA. ACM.

Stutzbach, D., Zhao, S., and Rejaie, R. (2007). Characterizing files in the modern gnutella network. *Multimedia Syst.*, 13(1):35–50.