An Experimental Study on Peer Selection in a P2P Network over PlanetLab

Fatos Xhafa
Department of LSI
Polytechnic University of Catalonia
Campus Nord, C/Jordi Girona 1-3
08034 Barcelona, Spain
fatos@lsi.upc.edu

Leonard Barolli
Dept. of Information and Communication Eng.
Fukuoka Institute of Technology (FIT)
3-30-1 Wajiro-higashi
Higashi-ku, Fukuoka 811-0295, Japan
barolli@fit.ac.jp

Raul Fernández
Open University of Catalonia
Department of Information Sciences
Av. Tibidabo, 39-43
08035 Barcelona, Spain
rfernandezco@uoc.edu

Thanasis Daradoumis
Open University of Catalonia
Department of Information Sciences
Av. Tibidabo, 39-43
08035 Barcelona, Spain
adaradoumis@uoc.edu

Abstract

Peer selection is an important aspect in many P2P applications requiring efficient assignment and execution of jobs to peer nodes and search and file transfer, among others. Due to increasing interest of using P2P systems for distributed computing, peer selection is taking relevance and several models have been proposed in the P2P literature. Yet, there are very few experimental studies for peer selection in P2P networks deployed in real large scale networks. In this work we present an experimental study that aims at revealing empirical information about the efficiency of selecting peer nodes in a P2P network when deployed in a real geographically distributed network. To this end, we have used a JXTA-based P2P platform deployed in PlanetLab, a planetary scale infrastructure, and have empirically evaluated several peer selection models. Our experimental study showed that in order to achieve efficient P2P applications, appropriate selection model should be used according to the characteristics of the application.

1. Introduction and motivation

P2P systems have become quite popular systems for file sharing among peers such as in Napster, Gnutella and FreeNet. Each time more, P2P systems are being used also as a new distributed computing paradigm for the development of large-scale distributed applications needing large computing capacity [10, 8, 3]. The advances in P2P systems, and particularly the improvements on P2P protocols are making possible P2P applications others than the well-known file-sharing applications. However, there is still few work to bring P2P system to real word applications, mainly due to the lack of robust P2P platforms that would allow the deployment of large P2P systems, in particular for efficiently discovering and selecting peers. Some advances are being done in this direction; for instance, the JXTA platform [2, 7] is making possible the development of P2P real-world applications.

Disposing large computing resources of millions of nodes of the P2P system, doesn't assure efficient P2P applications. Indeed, P2P networks join together very heterogeneous resources, interconnected by heterogenous networks. Therefore, while developing a P2P application the important issue of how to efficiently use nodes of the P2P network is raised. In most of the today's P2P applications this issue is dealt with in an *ad hoc* way. In fact, this could also explain why P2P systems remain difficult for many users [1].

In this work we address the issue of experimentally study peer selection in a P2P platform as a means to identify which peer selection models could be appropriate for which P2P applications. The experimental study thus aims at revealing real empirical data about the efficiency and intrinsics of selecting peer nodes in a P2P network when deployed in a real geographically distributed network. To this end, we have used a JXTA-based P2P platform, namely the JXTA-OVERLAY [5], deployed in PlanetLab [6] nodes –a planetary scale distributed infrastructure– and have empirically evaluated several peer selection models that include a scheduling-based selection model [4], a data evaluator model [11] and user's preference selection model. Our experimental study showed that in order to achieve efficient P2P applications, appropriate selection model should be used according to the type and characteristics of the application. Our approach is exemplified using the Sun's JXTA open protocols and has been validated using a P2P application for processing large size files of a virtual campus. JXTA has been chosen mainly due to its platform independence, transportability to any network, its basic functions that facilitate the implementation of the P2P platforms.

The rest of the paper is organized as follows. In Section 2 we give the peer selection models considered in this work. The architecture of the P2P platform used for the experimental study is presented in Section 3 and the evaluation results are given in Section 4. Finally, we conclude in Section 5 with some remarks and indicate directions for future work.

2. Selecting Peers in P2P-based Applications

The peer selection models considered for this experimental study include a scheduling-based model, a data evaluator model, and a user's preference selection model.

2.1. Scheduling-based selection model

In this model [4] the idea is to find/provision as many as possible available idle peers to which the new incoming jobs can be allocated. This type of peer selection could be useful, for instance, in real life applications such as seti@home [9] to study of protein folding and disease by utilizing the new Cell processor in Sony's Playstation 3 to achieve performance previously only possible on supercomputers. Crucial to this model is the ready time of peers in order to plan in advance the allocation of jobs to P2P nodes. The estimated time is computed by the broker peers based on historical data kept for the peergroup. In case several peers are available candidates for executing the task, some additional data and criteria such as CPU speed are used.

2.2. Data evaluator model

This model can be seen as a *cost* model since a cost is assigned to each peer based on historical and statistical data for the peer. Using such data, different criteria can be applied to identify the peer having the best cost. The criteria can be global, namely, global data is used to compute the cost of the peer (e.g. percentage of successfully sent messages in the current session, percentage of successfully sent messages in all sessions (total), percentage of successfully sent messages during the last k-hours; number of messages in the outbox queue now, average number of messages in the outbox queue, number of messages in the inbox queue now, average number of messages in the inbox queue, etc.) or more task execution oriented criteria (e.g. percentage of successfully executed tasks in the current session, percentage of successfully executed tasks in all sessions (total), percentage of tasks accepted by the peer for execution in the current session, percentage of tasks accepted by the peer for execution in all sessions (total), etc.). Also, specific criteria for file request and file transmission can be considered (e.g. percentage of sent files in this session, percentage of sent files in all sessions (total), percentage of cancelled file transfers in the current session, percentage of cancelled file transfers in all sessions (total), number of pending transfers, etc.).

Each of the above criteria is given a certain weight (either user defined or pre-specified) meaning that some criteria are more important than others or even some are negligible (of zero weight); the best cost peer is then chosen.

2.3. User's preference selection model

In this model the peer is selected by the user according to his preferences and experience in using the peer nodes of the P2P network. Thus, this model is useful when the user knows the performance of some peers in advance, for instance, from previous submissions of the tasks. This model has a very low computational cost. Its main drawback is that it does not take into account the current state of the selected peer nor the current state of the network.

3. The P2P distributed platform

In this section we briefly present the P2P distributed platform we have used for the purposes of this experimental study; for more details and updated information on this platform, the reader is referred to http://jxta-overlay.dev.java.net. The JXTA-Overlay is essentially composed of three modules: the *Broker*, the *Primitives* and the *Client* module. Altogether this three modules form a new *overlay* on top of JXTA. We give next a short description of the three modules of the overlay.

Primitives: The overlay provides a set of basic functionalities, that we call primitives since they will be part of any P2P application, as regards the discovery and allocations of resources. Essentially, the primitives includes functionalities that allow: peer discovery, peer's resources discovery, peer selection, resource allocation, file/data sharing, discovery and transmission, instant communication, peer group functionalities.

An important place in the primitives is given to functionalities related to the management of executable tasks. These functionalities are intended to give service to users/applications on top of the overlay that submit executable tasks and receive results in turn. It should also be mentioned that the file sharing and transmission functionalities extend existing JXTA functionalities of sharing in P2P systems since an efficient file transmission is necessary for submitting tasks to resources. Resource statistics is another important interface in the overlay, and it is particularly useful for the selection of peers (statistics about the peers, the peergroups, the brokers and the clients.) Brokers act as governors of the P2P network and clients are edge peers (either SimpleClient –without GUI, or Client with GUI).

4. Experimental evaluation

4.1. Deployment of the P2P network

In order to evaluate the performance of the presented peer selection models, first we deployed the P2P network using nodes of the PlanetLab platform. PlanetLab [6] is an open platform for developing, deploying and accessing planetary-scale services (Figure 1). It is, at the time of this writing, composed up of 782 nodes at 382 sites. Each Planetlab node is an IA32 machine that must comply with minimum hardware requirements (i.e. 1GHz PIII + 1Gb RAM) running the same base software, basically a modified Linux operating system offering services to create virtual isolated partitions in the node, called slivers, which look to users as the real machine. Planetlab allows every user to dynamically create up to one sliver in every node, the set of slivers assigned to a user form what is called a slice. It is said that a Planetlab node can run up to 100 concurrent slivers.

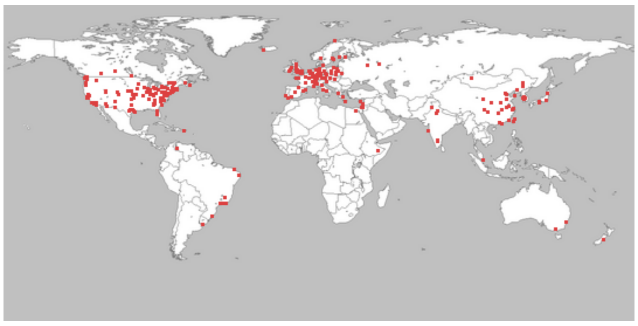

Figure 1. PlanetLab platform.

The sample set of PlanetLab's machines forming the slice is about 25 nodes shown in Table 1.

Moreover we used the cluster nozomi.lsi.upc.edu (a main control node + five computing nodes). The main node was used as one the brokers of the P2P network. For the purposes of this work, we will present the computational results of the experimental study for a group of 8 geographically distributed machines in seven EU contries that were used as SimpleClient (abbreviated SC) peer nodes, that is peer clients without GUI, consisting of:

Table 1. Nodes added to the PlanetLab slice.

ait05.us.es	planet01.hhi.fraunhofer.de
planet1.cs.huji.ac.il	planet1.manchester.ac.uk
system18.ncl-ext.net	planetlab1.net-research.org.uk
planetlab01.cs.tcd.ie	planet2.scs.stanford.edu
planetlab01.ethz.ch	planetlab1.ssvl.kth.se
planetlab1.esi.ucm.es	planetlab1.csg.unizh.ch
planetlab1.poly.edu	planetlab1.cslab.ece.ntua.gr
planetlab2.ls.fi.upm.es	planetlab1.eecs.iu-bremen.de
planetlab2.upc.es	planetlab1.hiit.fi
lsirextpc01.epfl.ch	planetlab5.upc.es
ricepl1.cs.rice.edu	planetlab1.itwm.fhg.de
planet2.seattle.intel-research.net	planetlab1.informatik.unierlangen.de
edi.tkn.tu-berlin.de	

- SC1: ait05.us.es
- SC2: planetlab1.hiit.fi
- SC3: planetlab01.cs.tcd.ie
- SC4: planetlab1.csg.unizh.ch
- SC5: edi.tkn.tu-berlin.de
- SC6: lsirextpc01.epfl.ch
- SC7: planetlab1.itwm.fhg.de
- SC8: planetlab1.ssvl.kth.se

4.2. Computational results and evaluation

One of the functionalities in the jxta-overlay is the file transfer. File transfer is among the most common operation in distributed P2P-based application, where a peer could just need to send a file to another peer or a file associated to a task should be send to a peer to execute the task. In such scenarios the efficiency of file transmission is very important. The aim of the experiment in this case was to reveal how efficient was the file transmission by using geographically distributed peers of a real P2P network, namely the jxta-overlay deployed over PlanetLab. In the following we consider essentially two scenarios: in the first, just file transmission time is empirically studied, while in the second we consider both file transmission time and processing in peer nodes of the PlanetLab.

File transmission. In order to obtain empirical data on file transmission, large files were considered. A file was split into many parts of a fixed size such as 50Mb, 100Mb,... and such parts were sent to peers. As soon as a peer receives the part, it should confirm correct reception of the file and its availability to receive another part. The experiment was repeated 5 times to get significant (averaged) results.

We give in Figure 2 the time in receiving the petition for a file transmission by a peer. As can bee seen by this figure, the peer node SC7 (`planetlab1.itwm.fhg.de`) takes a larger time just to receive the petition for file transmission.

We give in Figure 3 the transmission time of a file of size 50Mb and in Figure 4 the transmission time of receiving the last Mb. As expected, peer SC7 was the latest in completing the file transmission.

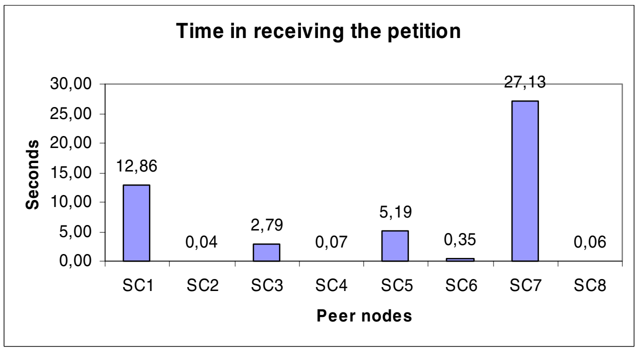

Figure 2. Time in receiving the petition time for file transmission.

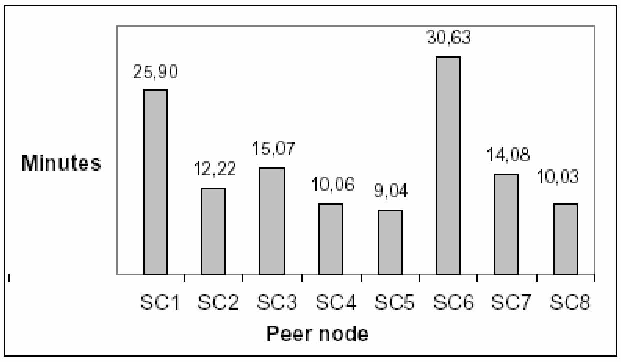

Figure 3. Transmission time for a file of 50 Mb.

As can bee seen from Figure 4, the time in completing the reception of the last Mb for peer SC7 is from 2 to 4 times slower than the rest of the peers. This empirical evidence shows that, in any P2P application involving file transmission, peers cannot be used in a blind way, rather, selecting them carefully is very important to achieve efficient P2P distributed applications.

In our next step in this experimental study, we considered the setting where the file is sent as a whole or it is divided into parts. The idea is that, by using characteristics of the P2P network, different granularity, that is, splitting the file into smaller size parts, in file transmission can be used to achieve an efficient overall file transmission time. The computational results are shown in Figure 5, the transmission time when the file was sent as a whole (100Mb) or was divided into parts (4 and 16 parts). As can bee seen from this figure, the transmission time of the file as a whole it's not worth! On the other hand, when the file is sent by smaller parts (in the case of dividing into 16 parts, the resulting files have size of 6.25Mb), the transmission time is in average 1.7 minutes, which is much smaller than the transmission time of the file as a whole and even when the division into 4 parts is considered.

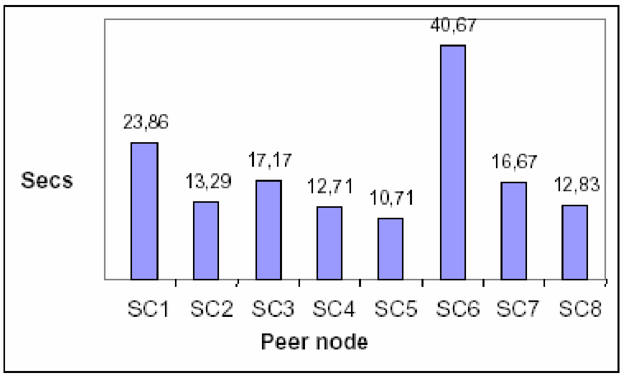

Figure 4. Transmission time of the last Mb.

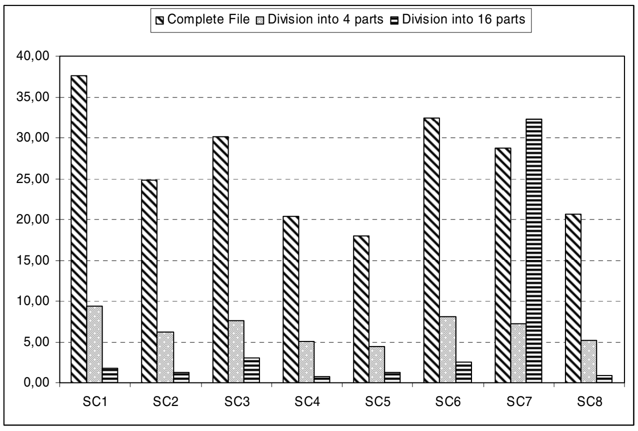

Figure 5. File transmission time (X-axis represents time in minutes; Y-axis represents 8 different peer nodes).

It should be noted the in the above experimental study, all peers were equally considered, that is no peer selection is done. But, as mentioned above, by not selecting the peer, large transmission times can occur since some peer node can worsen the overall time. Therefore, we considered the file transmission when the peer is selected according to three peer selection models (see Section 2). The transmission time when the peer was selected according to three peer selection models, namely economic scheduling model, data evaluator (same priority mode) and user's preference (quick peer mode) is shown in Figure 6

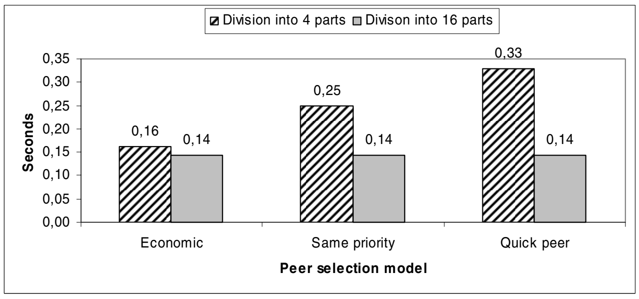

Figure 6. File transmission time according to three peer selection models.

File transmission and processing. In this setting, we measured the time needed when file transmission and processing takes place in peer nodes *versus* just processing time. This setting is very interesting when P2P application involve both file transmission and processing because, it very important to observe whether the processing time is superior or not to the processing time. Moreover, careful peer node selection should be done to avoid including peer nodes (such as peer node SC7 in our experiment). We show in Figure 7 the computational results.

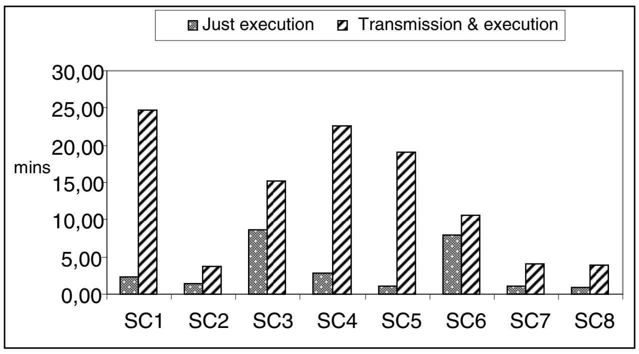

Figure 7. File transmission time according to three peer selection models.

5. Conclusions and future work

In this work we have presented an experimental study for peer selection on a P2P network deployed in PlanetLab –a real planetary-scale infrastructure. The experimental study revealed that using peer nodes in a "blind way" without taking into account the characteristics of the peers could yield to applications that do not benefit efficiently the large computing capacity of peer nodes. In our study we have used file transmission, which is an important feature in many P2P applications, as well as file transmission and file processing in peer nodes. By considering a small subset of peer nodes of PlanetLab geographically distributed, we could identify peers that if considered would be the "bottleneck" of the P2P application due to the large time needed in time of receiving petitions and file transmission. We considered then three peer selection models and presented a first evaluation of them, namely, the economic scheduling model, data evaluator model and user's preference peer selection model.

In our future work we would like to extend the empirical study of this work to study the performance of the proposed peer selection models by using a larger number of peer nodes. Also, we plan to measure the peer selection effect on real P2P large scale applications deployed in PlanetLab.

Acknowledgements

This work has been partially supported by the Spanish MCYT project TSI2005-08225-C07-05 and Project FP6-2004-IST-FETPI (AEOLUS).

References

[1] H. Bal, H. Casanova, J. Dongarra, and S. Matsuoka. Application-Level Tools. In Foster et al., eds, *The Grid: Blueprint for a New Computing Infrastructure*, chapter 24, 463–489. Morgan Kaufmann, 2003.

[2] D. Brookshier, D. Govoni, N. Krishnan, and J. Soto. *JXTA: Java P2P Programming*. Sams Pub., 2002.

[3] J. Crowcroft, T. Moreton, I. Pratt, and A. Twigg. Peer-to-Peer Technologies. In Foster and Kesselman, eds, *The Grid: Blueprint for a New Computing Infrastructure*, chapter 29, 593–622. Morgan Kaufmann, 2003.

[4] C. Ernemann, V. Hamscher, and R. Yahyapour. Economic scheduling in grid computing. In *The 8th Int. Workshop on Job Scheduling Strategies for Parallel Processing*, 128–152, UK, 2002.

[5] Jxta-Overlay. http://jxta-overlay.dev.java.net

[6] Planet Lab. http://planet-lab.org/.

[7] S. Oaks, B. Traversat, and L. Gong. *JXTA in a Nutshell*. O'Reilly, 2003.

[8] A. Oram, ed. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly, 2001.

[9] Seti@Home. http://setiathome.berkeley.edu/.

[10] C. Shikey. What is P2P ... and what isn't. O'Reilly Network, November 2000.

[11] J. Yu, M. Li, Y. Li, F. Hong, and M. Gao. A framework for price-based resource allocation on the grid. *Proc. of Parallel and Distributed Computing: Applications and Technologies*, vol. 3320 of *LNCS*, 341–344. Springer, 2004.