P2P File Sharing Analysis for a Better Performance

Martha-Rocio Ceballos
Doctoral student
Dept. of Telematics
Polytechnic University of Catalonia
+34 93 401 59 94
ceballos@entel.upc.es

Juan-Luis Gorricho
Associate professor
Dept. of Telematics
Polytechnic University of Catalonia
+34 93 401 68 30
juanluis@entel.upc.es

ABSTRACT
The so-called second generation P2P file-sharing applications have with no doubt a better performance than the first implementations. The most remarkable difference is due to the file division into smaller pieces, where a receiving peer of any piece automatically becomes a new source to other peers. But a new question arises on how we distribute all the pieces provided by a seed peer to minimize the global and presumably individual download times. In this paper we summarize part of the work we have developed up until now to answer this general question, in particular, we will analyze how close the present second generation P2P file-sharing applications remain from an ideal solution with the theoretical best performance, that is, where all peers are interconnected with each other and all peers have an altruistic behavior always uploading its contents at any chance. Successive modifications of the ideal solution will lead us to more realistic scenarios. We will estimate the performance on each case and finally present the current studies we are carrying out to improve the overall capacity.

Categories and Subject Descriptors
H.4.0 [**Information Systems Applications**]: General

General Terms
Algorithms, Experimentation, Measurement, Performance.

Keywords
File sharing, video streaming, network measurements, peer to peer applications, service capacity, performance evaluation.

1. INTRODUCTION
P2P file sharing applications have become very popular since the introduction of Napster, allowing users to share MP3 formatted music files, a few years ago. Independently of legal issues, the first P2P file sharing applications such as Napster, Gnutella and KaZaA were intended to satisfy the most relevant P2P properties: scalability, reliability and great efficiency on information delivery. Nevertheless, the free-riding phenomenon became an extended practice, peers downloading from other peers while not contributing to upload to others; and finally, in spite of the proclaimed P2P features, all these file sharing applications became the traditional client/server model, with only a few altruist peers as file servers and all the others as file requesters.

To avoid this undesirable situation, recent P2P file sharing applications such as BitTorrent [1] and eMule [2] have defined a new scenario with all peers forced to upload part of their received data if they want to download the complete file. The requested file is divided into chunks, so any peer receiving a chunk may be forced to upload it to other peers. These new P2P file sharing applications propose different algorithms to incentivate the peer collaboration, solving the way all peers establish and temporally renew its connections with other peers interested on the same file. A practical principle suggests that I will upload to you if you also upload to me, a tit-for-tat assumption [3].

In a more general sense, putting aside the free-riding phenomenon, the transmission of any information in smaller pieces always increases the system capacity when more than one node is involved, as re-transmitter or intended receiver. This is the case, because we don't wait for a completed file transmission to a particular peer, before this one begins the retransmission to the next peer. This feature is magnified if we deliver all chunks from the source peer, also called the seed peer, to the greatest variety of peers, this way we promote an increasing number of parallel transmissions among all peers.

The P2P networks are classified as structured and non-structured networks depending on the methodology used to organize the information search. On a structured network the resource location information is stored in a predefined way with the aid of hash tables, while on non-structured networks the resource location registration becomes an ad hoc process leading to a subsequent search procedure usually based on a flooding or a random-walk mechanism; in this case, to increase the search success the non-structured networks are usually organized in logically interconnected peers and super-peers, storing the resource location information only in the super-peers.

Independently of the P2P network type we can distinguish two phases: the resource search and the resource access or download, depending on the particular service. Considering a file sharing application the most relevant issue is the file download, not its location, due to the critical time delay we can experience if we have a high demand from the file requesters or a limited upload capacity from the file provider. In this respect it is crucial to achieve a reliable and scalable file delivery mechanism; this way we could even extend the file sharing application to implement the so many times named video streaming service without an IP multicast solution.

1.1 Relationship to previous work

Early studies on P2P file sharing have focused on traffic measurements for the most popular applications like: Gnutella, Napster and BitTorrent [4, 5, 6]. At present time more research has derived to the performance analysis of these applications from different approaches. In [7] a queuing model is proposed for an accurate study of the file sharing application performance. In [8] the study considers a transient regime to handle bursty traffic to demonstrate the exponential growth in service capacity and exhibits the sensitivity of this growth to system policies and parameters; later on, the same study considers a model for such systems in a steady state to show how the average delay seen by the peers would scale in the offered load and rate at which peers exit the system. In [9] a simple fluid model is presented to study the scalability, performance and efficiency of the file sharing mechanisms, including the evaluation of the BitTorrent built-in incentive mechanism on the network performance.

1.2 Our contributions

Our study on the performance characteristics of the present second generation P2P file sharing applications have been carried out in a similar fashion to [8,9], but in our particular case comparing the obtained results with a modified step by step semi-ideal application. Our study analyzes the chunk distribution with an ideal application, and how this distribution is deteriorated as we modify the best scenario to approach to the real situation of present applications. We intend to estimate the performance loss for the semi-ideal scenario, justify the losses with each new assumption and contribute with new ideas on how the present P2P file sharing applications could obtain a better performance.

2. An ideal P2P file sharing application

The characteristics of an ideal P2P file sharing application are those that maximize the throughput capacity of the network, achieving the lowest transmission delays for any peer trying to download a particular file. In our first simplified model for this ideal application we will assume all peers have the same transmission capacity for uploading and downloading. Also, to simplify the model we suppose any file is divided into a number of pieces, each one transmitted in a T interval time. At the beginning the seed-peer transmits the first chunk of the file to any peer; later on, the seed-peer continues transmitting consecutive chunks and simultaneously those peers which already have any chunk will try to retransmit one of them to other peers. In this ideal scenario, all peers are interconnected with each other, so a particular peer can deliver a chunk to any peer. The pieces delivery is done from the sources point of view, acting as peer re-transmitters, the goal is to upload as many chunks as possible. In the best case, when all peers have at least one chunk to upload, all peers but one will find a peer receiver, this is because we must count the seed-peer which is also a source peer. For this particular situation all peers are receiving a new chunk, so the system capacity is maximized. Let us see two examples of the pieces delivery among all peers with a different number of peers to check that we can easily find the best pieces distribution, we advance there is not a unique solution. Let us assume for this example the file has 9 pieces, and they are numbered from 1 to 9.

Table 1. Examples of chunk distribution with 3 and 8 peers

	P_1	P_2	P_3		P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
T_1	1				1							
T_2		1	2			1	2					
T_3	2	3	1					1	1	2	3	
T_4	3	2	4		2	2	1	3	4	1	1	1
T_5	4	5	3		3	3	4	2	2	5	2	2
T_6	5	4	6		4	4	3	5	3	3	6	3
T_7	6	7	5		5	5	6	4	7	4	4	4
T_8	7	6	8		6	6	5	7	5	8	5	5
T_9	8	9	7		7	7	8	6	6	6	9	6
T_{10}	9	8			8	8	7	9		7	7	7
T_{11}			9		9	9		8	8		8	8
T_{12}							9		9	9		9

On Table 1 we can see two different examples of chunk distribution with 3 and 8 peers as time goes by in the downward direction. On time T_1 the seed-peer delivers the first chunk to any peer, on time T_2 the seed-peer delivers the second chunk to another empty peer and the former receiver uploads its chunk to another empty peer. This initial phase lasts until all peers have at least one chunk to upload. The best results on chunk delivery are obtained precisely when the chunks are firstly delivered to empty peer. Later on, in the second phase, all peers receive a new chunk from another peer or the seed-peer. That means that all but one peer will upload a new chunk. The second phase lasts until the seed-peer delivers the last chunk: number 9. In the second phase the best results on chunk delivery are obtained when we proceed as follows:

1. - The peers try to upload the chunk with the largest index they have, beginning the chunk distribution from the peer with the shorter chunk index (the source peers are sorted from the one with the shorter of largest chunk index, min-max). In practice, all but one will upload its intended chunk. The receiving peers are asked on an indexed order.

2. - The seed-peer delivers the new next chunk to the only peer that did not receive any chunk from the other peers in step 1.

Following the same procedure as in second phase, although without the seed-peer participation, we arrive to the third phase where all peers finish downloading the remaining chunks.

We can say from Table 1 these are two examples of an ideal file sharing scheme as there is no hole, loss of transmission capacity, in the second phase, and the first and third phases are the best we can do.

We can appreciate on Table 1 for the 8 peers example how the chunks download follows an exponential law on the delivery of any particular chunk. The delivery sequence is always: T_n:1 (the seed-peer upload), T_{n+1}:1, T_{n+2}:2, T_{n+3}:4. And this pattern applies for any number of peers independently of its number; it follows a power of 2 law until the last step, when we deliver the intended chunk to the remaining peers.

We have checked the chunk distribution results following the procedure described in this section with an increasing number of peers, and we must say that quite frequently we achieve the best distribution. For those cases where there are inevitable holes in the mentioned second phase due to the applied algorithm, these represent less than 10% considering a 9 chunks file. These losses on transmission usually happen at the first steps on the second phase, and later on at isolated intervals. The mentioned percentage on lost transmissions becomes less than 4% considering files with more than 100 chunks.

Trying to diminish these percentages, we have proved different alternatives to modify the presented algorithm for step 1 on the second phase, asking the receiving peers on a random fashion and a min-max sort instead of the indexed order. We have also exchanged steps 1 and 2, so the seed-peer delivers the new chunk before the other peers ask the receiving peers to upload a chunk. Many resultant chunk distributions have been tested. We must say that in all cases the obtained results are quite similar to those mentioned in the previous paragraph.

2.1 An increasing number of peers

In a more realistic scenario we are not going to have a constant number of peers participating in the file download since the very beginning. For this new approach we can establish an initial number of peers interested in a particular file, and new participants joining the peers community later on. We can assume the transmission time for any chunk small enough to admit individual incorporations at each round. The initial assumption of a complete meshed network implies that the increasing number of peers has no relevance to the chunk distribution. If the new peer is added at the first phase, this will be a new empty peer. If it enters the system in the second phase, there is no impact as we have mentioned earlier that in this phase there is always one peer which does not upload any chunk, this time it will do that to the new peer. And finally, the peer incorporation in the third phase has no impact to the remaining peers as they are in the final term of chunk delivery and there are more sources than requesters.

Table 2. Example with an increasing number of peers

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
T_1	1									
T_2		1	2							
T_3				1	1	2	3			
T_4	2	2	1	3	4	1	1	1		
T_5	3	3	4	2	2	5	2	2		
T_6	4	4	3	5	3	3	6	3	2	
T_7	5	5	6	4	7	4	4	4	3	
T_8	6	6	5	7	5	8	5	5	4	3
T_9	7	7	8	6	6	6	9	6	5	4
T_{10}	8	8	7	9		7	7	7	6	5
T_{11}	9	9		8	8		8	8	7	6
T_{12}			9		9	9		9	8	7
T_{13}									9	8
T_{14}									1	9
T_{15}										2
T_{16}										1

On Table 2 we can see the particular chunk distribution with initially 7 peers and the incorporation of P_8 at T_4, P_9 at T_6 and P_{10} at T_8. As we can appreciate, the new peers receive the chunk numbers distributed at the time they join the system, and from then on they participate on the chunks delivery dictated by the older peers. At the end, the new ones will download the remaining chunks until they complete the entire file.

3. A semi-ideal P2P file sharing application

There are two main assumptions made in the previous section which have defined an ideal P2P file sharing application. The most relevant one considers a complete meshed network with all the participants. This implies any new peer incorporation forces the establishment of a new connection from all the peers to the new one. The other assumption considers a chunk distribution carried out by all peers in the most altruistic manner. All peers try to upload the chunk with the largest index they have, as it is supposed to be the rarest chunk the peer can offer to the other peers, contributing this way to the intended exponential growth on the chunk distribution.

3.1 Chunk delivery on requesters demand

Let us suppose a more realistic scenario with a chunk delivery from the requesters' point of view. This time every peer in an indexed order asks another peer in a min-max order for a new chunk. The asked peer will first offer the largest index chunk it has, if not required, it will offer the next one, and so on, until the last one. If the requester peer has all the offered chunks, it will ask the next peer, and so on. On the same round, the seed-peer will upload the new chunk to the first unsuccessful peer from the previous procedure in an indexed order, with empty peer preference. Let us see an example of the resultant chunk delivery with an initial 7 peer network and an increasing number of participants, the same as on Table 2.

Table 3. Example of chunk delivery on requesters demand

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
T_1	1									
T_2		1	2							
T_3	2		1	1	3					
T_4	3	2		2	1	1	4			
T_5	4	3	3	5	2	2	1	2		
T_6	5	4	4	3	6	3	2	1	3	
T_7	6	5	5	4	4	4	3	3	7	
T_8	7	6	6	8	5	5	5	4	2	4
T_9	8	7	7	6	9	6	6	5	4	3
T_{10}	9	8	8	7	7	7		6	5	6
T_{11}		9	9		8	8	7	7	6	5
T_{12}				9		9	8	8	1	7
T_{13}							9	9	8	2
T_{14}									9	8
T_{15}										9
T_{16}										1

On Table 3 we can appreciate there is no great difference between the chunk distribution observed on Table 2. This is due to the relatively short number of nodes: 7 initial peers and 3 new incorporations. The difference is really at T_4, here there are 6 downloads, on Table 2 at the same round of time there are 8 downloads. This inefficiency will increase with the number of initial peers and it lasts for more than one T period. An extensive number of simulations have demonstrated the reduction of the transmission capacity is only observed at the initial phase, with the consequent delay spread as the number of peers increase. In fact, we have appreciated a certain loss of the exponential growth behavior on the chunks delivery. The second phase, where all nodes have something to upload, and the third one have demonstrated no remarkable degradations.

3.2 Chunk delivery with limited connections

The most problematic assumption made on the ideal application is the consideration of a complete meshed network. A real P2P file sharing application will work with a limited number of connections for any peer. In this scenario some questions arise: how many connections would be desirable? how do we choose the peers we will connect to? To overcome the free-riding phenomenon it's been said that I will upload to you if you upload to me, does it mean we must define complete meshed subsets of peers to promote this collaboration? When do we decide to renew the established connections? And how can we find the appropriate new peers?

We will go a little bit further in our desire to stay as close as possible to the ideal file sharing application. It seems that, in spite of a mandatory limited number of connections, the idea of a complete meshed subset of peers will not contribute to the exponential growth on chunk delivery. In that respect, let us propose the establishment of disjoint peer connections, that is, any peer will randomly connect to a limited number of peers with upload purposes. If all peers act the same way we will create a meshed, although not complete, network, but again it will presumably work fine if all peers have an altruistic behavior, just the opposite to the tit-for-tat principle mentioned above. Nevertheless, this is a more realistic system implementation as we are already working with a limited number of connections. Let us see an example of the resultant chunk distribution with an initial and constant number of peers, all of them working with 4 connections. The total system transmission delay is less than the value obtained in the previous two examples because this time all peers are present at the simulation start.

Table 4. Example with a limited number of connections

	P_1	**P_2**	**P_3**	**P_4**	**P_5**	**P_6**	**P_7**	**P_8**	**P_9**	**P_{10}**
T_1	1									
T_2		2		1						
T_3			3			1			1	2
T_4	4	3		2	1		1	1	2	1
T_5	5	1	1	4		2	2	2	3	3
T_6	2	6		3	2	5	3		4	
T_7	3	7	2	5	3	3	4	4	6	5
T_8	6	8	7		4	4	5	5	5	4
T_9	7	5	4	6	5	9	6	3	8	
T_{10}	9	4	5	7	7		8	6	7	6
T_{11}	8		8	9	6	6		7	9	7
T_{12}		9	6	8	8	7	7	9		8
T_{13}			9		9	8	9	8		9

On table 4, all peers establish 4 connections at random; these connections have only an upload purpose. The simulation runs considering that first of all the peers in a min-max order try to upload a chunk to one of their 4 connected peers, offering the largest index chunk it has; if no one is interested, it will offer the next chunk, and so on; if, at any round this peer can not upload any chunk to the connected peers, we consider a loss in transmission and it renews the less used connection with a new peer in a random fashion for the next round. On the same round, the seed-peer uploads a new chunk in an indexed order to any free peer, choosing only empty peers when available.

We have made many simulations with a different number of peers and a different number of simultaneous connections. In the worst case, working with 3 or 4 connections per node and more than 100 peers we can appreciate a reduction on the system capacity compared to the ideal situation where all peers are interconnected of about 30%. This capacity reduction diminishes rapidly as we increase the number of connections per node.

4. The research plan

In this section we will present the last results in our research. In particular, we will describe the most remarkable features of the BitTorrent and E-mule applications. From them, we will inherit the relevant features useful to define an application model for the intended simulations. We will explain with a few illustrative examples the obtained chunk distributions with this model. We will present in detail the most remarkable results obtained from the comparison on system performance between the semi-ideal application model we have been working on and the second generation file sharing application model. This will conclude with some ideas on how the present P2P file sharing applications could obtain a better performance.

Finally we will describe our future research plan consisting on the implementation of a file-sharing P2P network to be deployed and evaluated on the Internet; this way we will achieve an empirical comparison with other alternatives, and presumably extend the file-sharing P2P application to a multimedia-streaming P2P service.

5. ACKNOWLEDGMENTS

This work was supported by CICYT Grant TIC2003-08129-C02.

6. REFERENCES

[1] Cohen B. *Incentives build robustness in bittorrent*, May 2003. http://bitconjurer.org/BitTorrent/bittorrentecon.pdf.

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

[3] Hales D. and Patarin S. *Computational sociology for systems "in the wild": the case of BitTorrent*. IEEE Distributed Systems on-line, Vol. 6, n. 7, July 2005.

[4] Ripeanu M. *Peer-to-peer architecture case study: Gnutella network*. Technical report, University of Chicago, 2001.

[5] Ripeanu M., Foster I. and Iamnitchi A. *Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design*. IEEE Internet Computing Journal, 6(1), 2002.

[6] Andrade N., Mowbray M., Lima A., Wagner G. and Ripeanu M. *Influencies on cooperation in BitTorrent communities*. SIGCOMM'05 Workshops, August 22–26, 2005, Philadelphia, PA, USA.

[7] Ge Z., Figueiredo D.R., Jaiswal S. *Modeling peer-peer file sharing systems*. In Proceedings of IEEE INFOCOM, 2003.

[8] Yang X. and de Veciana G. *Service Capacity of Peer to Peer Networks*. In Proceedings of IEEE INFOCOM, 2004.

[9] Qiu D. and Srikant R. *Modelling and performance analysis of BitTorrent-like peer-to-peer networks*. SIGCOMM'04, Aug. 30–Sept. 3, 2004, Portland, Oregon, USA