Enhancing Peer to Peer Parallel Data Access with PeerFecT

Laurent Dairaine, Laurent Lancérica, Jérôme Lacan

{Laurent.Dairaine, Laurent.Lancerica, Jerome.Lacan}@ensica.fr
Tel : +(33)561618683 / Fax : +(33)561618688
ENSICA / TéSA, 1, place Emile Blouin 31056 Toulouse cedex 5, France

Abstract. Data access performances over a Peer-to-Peer (P2P) network is a critical issue since downloading time remains the longest phase of P2P usage by end-user. In the present effort, we propose an innovative approach called PeerFecT (*Peer-to-peer with Fec dissemination Technique*) based on an erasure code, often called Forward Error Correction (FEC) code, to speed-up data access into P2P networks. The main idea is to use the erasure code to dilute the information over the peers. This dilution allows a greater and flexible choice among all the disseminated encoded data and then statistically enhances the overall throughput of the transfer. This paper focus on the performance gain this technique can provide. A performance evaluation based on an original model using the results of a measurement campaign of sequential and parallel downloads in a real P2P network over Internet is presented. The results show that for the same amount of information disseminated into the P2P network, our solution is more efficient compared to classical file replication approaches.
Keywords. P2P (Peer-to-peer), Erasure codes, parallel data access.

Introduction

The popularity of Peer-to-peer (P2P) file sharing applications offers new prospects to Internet end-users. The main benefits of P2P networks are to enhance the utilization of information, bandwidth, and computing resources. As a result, numerous research level issues are related to the conception of such a system. Scalability, fault tolerance, authentication, routing or data localization are examples of hot topics concerning this domain. Nevertheless, the performance of data access and downloading time over the P2P networks is the crucial aspect in such systems. The contribution of this paper concerns these last issues.

A classical approach to cope with data access performance consists in enhancing the localization of replicated copies [1]. The end user then simply downloads one of them from the closest peer. Typical enhancement of this scheme consists in getting various blocks from different peers in parallel. In the present effort, we propose a complementary approach based on the use of an erasure code, also called Forward Error Correction (FEC) codes, to speed-up the data access into the P2P network. We named

PeerFecT (Peer-to-peer with FEC dissemination Technique) this approach. This paper focus on the gain in terms of downloading time provided from the use of erasure codes in P2P networks. The FEC usage allows the information to be diluted over a set of different data blocks. Those blocks once disseminated over the P2P network, FEC properties induce that only a subset of the total set of blocks is necessary to reconstitute the original information. Due to the universal value of each encoded block, any client will be statistically situated closer to the minimal necessary amount of blocks among the multiple FEC blocks copies than in the case of classical approaches. Then, the overall downloading performance will be enhanced. This paper shows that for the same amount of information disseminated into the P2P network, this solution is statistically more efficient in terms of data access performance compared to classical file replication. This property is verified whatever the downloading technique used, i.e., sequential or parallel access to remote peers.

The paper is structured as follows. The first section describes the PeerFecT idea, explaining how the FEC are used into the P2P network to enhance data access time. The second section presents and analyses the technique used into parallel data access context, context largely implemented into today's P2P systems. The last section considers main issues for implementing PeerFecT. Finally, concluding remarks and future work are given.

Using Erasure Codes to Enhance Data Access Performance in P2P System

A P2P system uses a peering architecture that offers the support for various P2P services such as applicative multicast communications or file sharing between peers. Examples of implemented file sharing P2P systems are Napster or Gnutella. Using these systems, each node is able to determine its peers and to localize data over the peer network. Furthermore, the peering algorithms allow data dissemination and cost determination in terms of bandwidth, for peer to peer data transfers. Concerning the data localization among peers, several efficient techniques have already been proposed (see e.g.[2]). Recall that erasure codes are a mechanism classically used to protect information against errors or losses in transmissions (see e.g. [6]). The principle is to add redundancy to the transmitted information, permitting receivers to recover the whole original data, even after experiencing transmission errors.

The PeerFecT approach is based on the following method. When a file must be published into the network, it is firstly cut into k blocks which are encoded using an erasure code to obtain a set of n blocks. The last phase of publication is the dissemination stage of the various blocks over the P2P network (briefly discussed in Implementation Issues Section).

Considering the various encoded blocks disseminated into the network, and thanks to the erasure code properties, downloading a complete file is equivalent to downloading any k blocks among the total n ones. Hence, compared to classical approaches, there are greater choices for the sets of k blocks to get, to reconstitute the original data. Note that the availability and the robustness of the system are also increased (for a fault-tolerant point-of-view of similar systems, see e.g. [9]). When these blocks are disseminated over the P2P network, searching and throughput measurement services help to determine the closest ones by considering a certain cost function (e.g., the largest throughput). Then, the k closest blocks are downloaded in a classical way. The original data file can be finally obtained from the decoding of those k blocks.

An evaluation of the PeerFecT performance was proposed in [4] for sequential downloading, i.e. when the blocks are downloaded one after the other. Three dissemination strategies were studied: The first approach, *a) Entire File Replication*, consists in simply replicating the entire file into the network. The second approach, *b) no-FEC Block Dissemination*, splits the file into several blocks and distributes them. The last approach, *c) FEC Block Dissemination*, disseminates the blocks previously encoded with an erasure code. Note that the three approaches disseminate the same amount of data in the network. The first result shown in [4] is that no performance gain is obtained when using a simple *No-FEC Block dissemination* compare to *Entire File Replication*. This result is not amazing and the equivalence of the two approaches has been mathematically proved. The second important result was that PeerFecT approach was always more efficient than Entire File Replication and *No-FEC Block dissemination*.

Parallel Data Access Performance Evaluation

It should be noted that the context of sequential downloading is not realistic. Indeed, one major characteristic of recent P2P systems is to support a parallel access to several peers [1] (i.e., several blocks are downloaded at the same time). This parallel strategy drastically improves the downloading time. Those gains are evaluated in the context of mirror sites in [7].

Problem Modeling

We propose a simple model allowing computing the cost to get a file with the three previously considered approaches.

For the first strategy, *(a) Entire File Replication*, we consider that the r file replicas are downloaded independently with the bandwidths $Bw_1, Bw_2, ..., Bw_r$. By opening r parallel connections towards the r peers and assuming that each connection downloads different parts of the file, it could be obtained a total bandwidth up to $Bw_1 + Bw_2 + ... + Bw_r$. This maximum is reached if all the connections are bottleneck-disjoint i.e., without any side-effect between the various connections. In a P2P network, where the connections can be established among several thousands different peers, it is very difficult to evaluate the presence of bottlenecks due to parallel-downloads of different parts of the same file. Thanks to the over-provisioning of the network infrastructure, we consider that the bottleneck are necessary situated near the client, at access network, due to bandwidth limitation. This assumption is strongly confirmed by the campaign of measures presented in the next part of this section.

Let us denote this access bandwidth by Bw_0. By using this new parameter, we can state that the available bandwidth is now equal to $\min(\sum_{i=1}^{r} Bw_i, Bw_0)$. The cost to get the entire file is then $C_{(a)} = S_F / \min(\sum_{i=1}^{r} Bw_i, Bw_0)$. For the second strategy, (b) no-FEC Block Dissemination, the file is segmented into k blocks of size S_b, and each block is replicated r times in the network. Let us denote by $Bw_{i,j}$ the available bandwidth for the download of the i^{th} replica of the j^{th} block. This block can be downloaded for a cost equal to $S_b / Bw_{i,j_i}$ where $Bw_{i,j_i} = \max_{j=1..r}(Bw_{i,j})$. Note that parallel accesses to several replicas of the same block could be opened, but this would be equivalent to have a larger number of blocks and could also decrease performance due to the added load. So we only consider one connection per block. With this hypothesis, k parallel connections are opened. The minimal cost to get the entire file is then $\max_{i=1..r}(S_b / Bw_{i,j_i})$. As stated previously, this expression holds if there is no bottleneck due to the parallel downloads.

If $\sum_{i=1}^{k} Bw_{i,j_i} < Bw_0$, where Bw_0 denotes the access bandwidth limitation, then the minimal cost is given by the previous formula. On the other hand, if $\sum_{i=1}^{k} Bw_{i,j_i} \geq Bw_0$, due to the TCP fairness, the bandwidth Bw_0 is equally split into the k connections, i.e. a bandwidth of Bw_0 / k is supposed to be available for each connection. But if there is a bandwidth $Bw_{i,j_i} < Bw_0 / k$, the corresponding connection is not modified by the limitation resulting from this bandwidth limitation. The previously given cost value only depends on the minimum value of the Bw_{i,i_i}, then we can state that $C_{(b)} = \frac{S_b}{\min_{i=1..k}(Bw_{i,j_i})} \; if \; \min_{i=1..k}(Bw_{i,j_i}) < Bw_0 / k$ and $\frac{S_b}{Bw_0 / k} \; else$.

The case (c) FEC-block Dissemination can be analysed in the same way than case (b). The k original blocks constituting the file become now n blocks due to FEC redundancy. Getting the file consists now in downloading the k blocks of minimum cost over the n blocks. If the n blocks are replicated r' times over the network, we define $Bw_{i,j_i} = \max_{j=1..r'}(Bw_{i,j})$ for i=1,..n and the function $f : \{1,...,n\} \longrightarrow \{1,...,n\}$ such that $Bw_{f(1),j_{f(1)}} > Bw_{f(2),j_{f(2)}} > ... > Bw_{f(n),j_{f(n)}}$. Using similar considerations than in the previous case, the cost is expressed as follows $C_{(c)} = \frac{S_b}{Bw_{k,j_{f(k)}}}$ if $(Bw_{f(k),j_{f(k)}} < Bw_0 / k)$ and $C_{(c)} = \frac{S_b}{Bw_0 / k}$ else.

Simulation Study

The simple model given in the previous section has been implemented for simulation. The main idea is to consider 3 arrays of size r for entire file dissemination and of size k.r and n.r' for respectively no-FEC block dissemination and FEC-block dissemination. These arrays contain the available bandwidth towards the peers containing the blocks or the files. To be fair between the three studied approaches in terms of data storage load over the P2P network, we assume the total number of blocks disseminated into the network to be constant, then $r' = k.r / n$. The arrays are randomly filled following a bandwidth distribution deduced from a campaign of real measurements on the Gnutella network. This campaign was realized between the 1st and the 29th of January 2003 on ENSICA local area network, connected to a Gigabit Metropolitan network, this network being connected to the French research backbone RENATER network with a 155 Mbits/s access bandwidth. In this campaign, 1000 files were downloaded from 1000 different peers in sequential then in parallel (10 simultaneous connections) accesses. The measurements results, given in Figure 1, strongly confirm the assumption that "no bottleneck will appear in the backbone due to the traffic generated by the parallel downloads" (the curve representing the bandwidth towards a peer in a parallel access is the same than for the sequential access).

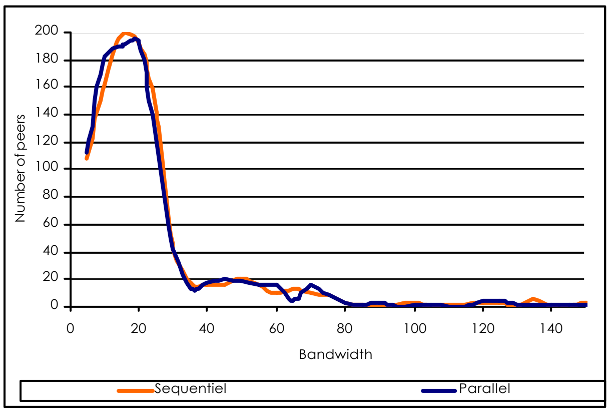

Fig. 1. Topology of the Gnutella network in terms of bandwidth (Kb/s).

The arrays were independently filled and analysed 1.000.000 times. The results of the simulations are presented in the next Section. Note that in all simulations except the last one, the access bandwidth at transport level is supposed to be 500Kb/s.

Results

In the following Figures, the results are presented in terms of gain percentages of the FEC and/or No FEC dissemination approaches compared to the entire file dissemination in a parallel downloading context.

Figure 2(a) evaluates the impact of the number of blocks k constituting the file for the case of No FEC and FEC disseminations. In the No FEC approach, the number of replicas is r=10. In the FEC approach, the redundancy factor n/k is equal to 5 and the number of replicas is r'=2. The main conclusions are that the FEC approach is always better than the No FEC approach and that the maximum gain (up to 60%) is obtained very soon. 10 blocks seems to be a good compromise between the performance gain and data storage load. Note that the negative gain for the No FEC approach is explained by non sufficient number of blocks compared to the entire file replication in the parallel approach.

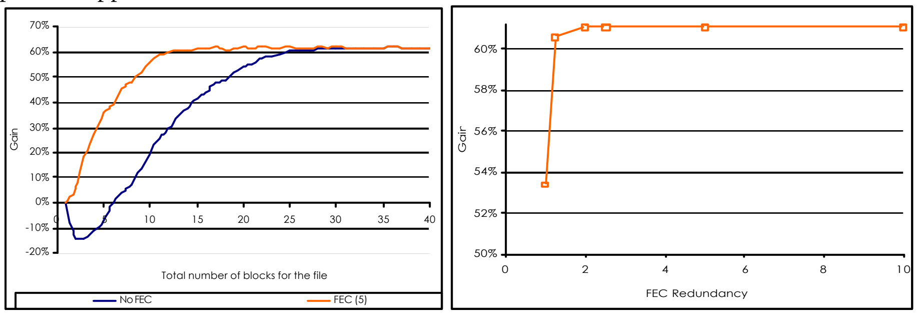

Fig. 2. (a) Impact on the number of blocks. (b) Impact on FEC redundancy factor

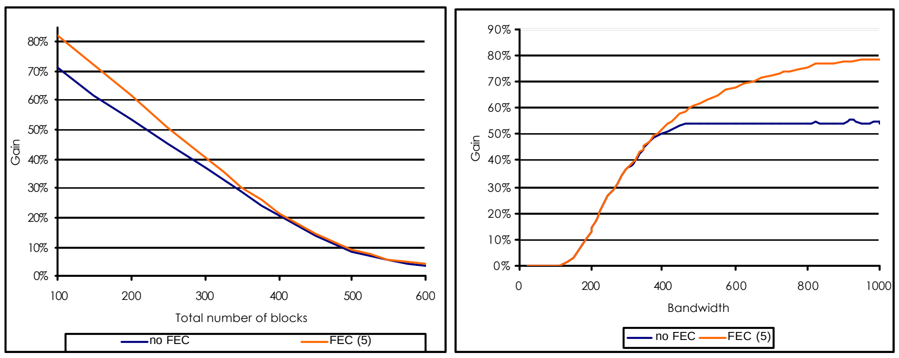

Fig. 3. (a) Impact of total number of blocks (b) Impact of access throughput (Kb/s)

The impact of the FEC redundancy factor (i.e. n/k) is shown on Figure 2(b). In this simulation, the number of file blocks is k=20 and the number of replicas is r=10 for the reference case (entire file dissemination). In order to be fair, the number of replicas r' is varying to satisfy the equality $r' = k.r/n$. The results show that a low redundancy factor n/k (2 or 3) is sufficient to obtain very important gain (up to 61% here).

Figure 3(a) presents the results of simulations which make the number of replicas r and r' vary. The value of k is fixed to 20 and the redundancy factor is equal to 5 (then, n=100). These results clearly show that PeerFecT is more powerful with a limited total number of blocks (up to 400). This observation can be explained by the fact that a large number of file or blocks replicas make them very accessible and then the FEC become less interesting.

The last simulation studies the impact of access bandwidth variation on performance. The file is split into k=20 blocks, each one being replicated r=10 times for the No FEC approach. The FEC approach uses a redundancy factor of n/k=5 and so a number of replicas r'=2. Figure 3.(b) presents the obtained results. This curve determines the domain of interest of PeerFecT. The left part shows that up to 350 Kb/s, there is no significant gain between No FEC and FEC approaches. This result illustrates that no performance gain is obtained when the bottleneck is situated on the access network. The right part of the curve illustrates the gain compared to the No FEC technique.

Implementation Issues

This section considers the implementation of PeerFecT over a typical P2P architecture by analysing the main modifications of the classical P2P services due to PeerFecT.

The main classical P2P service used by PeerFecT is the *Searching* service. Indeed, the increasing of the global number of different blocks in the network may introduce a potential bandwidth overhead due to the signalisation messages generated by the searching phase. Nevertheless, depending on the underlying searching algorithm, an adapted naming of the various blocks can easily cope with this problem. For example, Gridella P2P system [1] provides a searching service that finds, with a low overhead, a set of files whose filename matches the same prefix. Then, using such a service in PeerFecT, all encoded blocks names are build from a prefix associated to the researched filename. The suffix can be an identifier associated to the block.

Another potential problem concerns the throughput estimation between peers. This estimation is crucial in PeerFecT because it is used to choose the location the blocks are to be downloaded from. Most of P2P systems provide mechanisms for such throughput estimation. Even these estimations are not accurate, recent research results indicate how to obtain an accurate estimation of the throughput [5][3]. Moreover, simple algorithms already implemented in classical P2P architectures allow classifying peers according to their relative throughput, even if it is not really accurate.

The implementation of the main functional blocks of the core of PeerFecT system can be described as follows. The main problem of the encoding/decoding phases is that they could be expensive in time for large files. However, the encoding/emission (resp. reception/decoding) can be organized in order to pipeline these operations. Since the encoding/decoding throughputs are in most cases greater than the transmission one [6], the latency due to this phase can be neglected.

The other important point of the PeerFecT scheme is the dissemination of the encoded blocks. A detailed description of the different possibilities is beyond the scope of this short paper, but two approaches are possible. First one is a user-driven scheme only using the downloading of blocks to ensure the block dissemination. This solution takes advantage of being costless in terms of downloading overhead (i.e., the downloading is achieved by users) and ensures the property of disseminating blocks depending on file popularity. The second approach is a pro-active scheme that spreads a given number of blocks onto the network before any user-download. This scheme enhances the performance for a given content as soon at it is published. The blocks distribution can be managed in such a way to control the total number of blocks disseminated over the P2P network.

Concluding Remarks

In this paper, we have presented a P2P file sharing system that uses an erasure code to disseminate information over the peers. Performance gains were estimated by comparing our system to classical approaches of various P2P systems.

Several lessons emerge from the obtained results. First, the use of FEC allows increasing the average availability of data and decreasing the access time. Secondly, although the obtained results depend on the distribution law of the bandwidth over the peer network, there is a set of values (number of blocks, FEC redundancy factor, and number of replicas) that optimize the peering architecture. Finally, the use of such FEC-based system enhances robustness and data availability, while keeping constant the storage capacity of every peer. All the simulations are based on a real measurement campaign providing an accurate estimation of the bandwidth distribution law of the peer. As a result, an accurate and realistic estimation of the PeerFecT performances are provided. In the sights of the obtained results, future work mainly concerns real implementation of PeerFecT and development of an adapted erasure code.

References

[1] K. Aberer, M. Punceva, M. Hauswirth, R. Schmidt, "Improving Data Access in P2P Systems", IEEE Internet Computing 6, 1, pp. 58-67, Jan./Feb. 2002.

[2] E. Cohen, S. Shenker "Replication Strategies in Unstructured Peer-to-peer Networks", ACM SIGCOMM 2002, August 2002.

[3] T.S. Eugene Ng, Y. Chu, S. Rao, K. Sripanidkulchai, H. Zhang, "Measurement-Based Optimization Techniques for Bandwidth-Demanding Peer-to-Peer Systems", INFOCOM 2003.

[4] J. Lacan, L. Lancérica, L. Dairaine, "When FEC Speed Up Data Access in P2P Networks", in Proc. of IDMS-PROMS, LNCS, Springer, 2002

[5] K. Lai, M Baker, "Nettimer : a tool for measuring bottleneck link bandwidth", Proc. USENIX Symp. on Internet Technologies and Systems, 2001.

[6] L. Rizzo, "Effective Erasure Codes for Reliable Computer Communication Protocols", In *Computer Communication Review*, April 1997.

[7] P. Rodriguez, A. Kirpal, E. W. Biersack, "Parallel-Access for Mirror Sites in the Internet", In Proceedings of IEEE/Infocom'2000, Tel-Aviv, Israel, 2000.

[8] S. Saroiu, P. Gummadi, S. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems", T.R. UW-CSE-01-06-02, Seattle, July 2001

[9] H.Weatherspoon, J. D. Kubiatowicz, "Erasure Coding vs. Replication: A Quantitative Comparison", inProc. of IPTPS '02, March 2002