On Fairness, Optimal Download Performance and Proportional Replication in Peer-to-Peer Networks

Saurabh Tewari and Leonard Kleinrock

Computer Science Department, University of California at Los Angeles,
4403 Boelter Hall, Los Angeles, CA 90095, U.S.A.
{stewari, lk}@cs.ucla.edu

Abstract. Each peer in a peer-to-peer network, by definition, is both a consumer and a provider of the service. As a consumer, a peer wants to obtain its objects of interest as quickly as possible. However, as a service provider, the peer wants to serve no more than an equitable portion of the total workload. Our first observation in this paper is that if one satisfies the latter criterion of fairness in workload distribution, then one also minimizes the average download time when the delay at the server is convex in the utilization factor of the server. We had previously observed that controlled flooding search in unstructured networks is optimized when the number of replicas of a file is proportional to the request rate for that file. Here we show that such a replica distribution also ensures fairness in the workload distribution and, at the same time, minimizes the average download time seen by a download request.

1 Introduction

Peer-to-peer networks offer a promise of systems that automatically scale in capacity as the number of users increases and yet are extremely robust, automatically adapting to failures of nodes/links as well as to changes in usage patterns, all at virtually no cost. These loosely organized networks of autonomous entities (user nodes or "peers"), which make their resources available to other peers, represent a new computing paradigm where the service consumers are, now, the service providers as well. These systems offer the potential of a tremendous improvement over the traditional client-server architectures.

For a user of a peer-to-peer content distribution system, the measure of the system performance is the time it takes to fulfill request for a particular file which consists of two components: the time it takes to find who has that content, and the time to actually download the content. While a significant amount of attention has been paid to the search aspect in past, as the sizes of the files exchanged become larger and larger, the download time has become the dominant factor in the perceived performance of the system by a user. This shift towards larger file sizes also implies that resources consumed in supportm of file searches is no longer the dominant component of the peer-to-peer workload at a peer but, rather, servicing the download requests becomes the majority of the workload. Thus, as a service consumer, a peer wants to see the minimum possible download time, and as a service provider, the peer expects a fair distribution of the workload. In this paper, we address these objectives of fairness in workload distribution and minimization of the download time. After briefly discussing the related work, we present our model for the peer-to-peer system in Section 3. Optimization of the download time is discussed in Section 4 where we establish that our two objectives are perfectly congruent as uniform node utilization also minimizes the download time. Section 5 discusses uniform node utilization where we show that if the number of replicas of a file is proportional to the request rate for that file, each node operates at an equal utilization factor regardless of which files it is sharing, and hence, such a proportional file replication also minimizes the download time. Section 6 contains our conclusions.

2 Related Work

As the size of content exchanged over peer-to-peer networks has increased from few megabytes to hundreds of megabytes, a few researchers have begun to look the issue of download performance. [10] and [14] model the download service capacity in the BitTorrent file-sharing system [2] in relation to the evolution of the file request rate (specifically, they derive the download service capacity of the system as a function of the age of the file in the system) while [12] studies the download time in these systems via simulations. However, since all of them seek to model the performance of a particular system, they do not attempt to optimize the download performance. [4] presents a fairly detailed model for peer-to-peer file sharing systems that incorporates the effect of both the search-processing load as well as the download service loads at nodes and the download service capacities for each file for peer-to-peer file systems using different search mechanisms. However, like [10] and [14], its objective is also modeling of existing designs and they do not examine alternatives that may improve performance. Our peer-to-peer system model is much simpler which gives us the advantage that we can find opportunities for performance optimization with a relatively simple analysis.

In terms of similarity in peer-to-peer system models and interest in file replication distribution as a mean to improve system performance, [9] and [13] are closely related to this paper. However, these papers address the search performance in unstructured peer-to-peer networks so their models include additional search-related constructs whereas our results are independent of any underlying search mechanism. We use the file replication distribution shown to be optimal for controlled flooding search in [13] as our starting point as we seek a replication distribution to minimize the download time. While [9] finds the proportional file replication to be sub-optimal for their preferred random walk search mechanism, in rejecting this solution it did note that such a distribution implies that each replica of a file services equal download requests per unit time. However, due to their focus on search performance, [9] did not take the next step of finding that this distribution implies uniform node utilization. Further, since it does not address the download time, the optimality of such a distribution for download time is overlooked. Further, our results are applicable to structured peer-to-peer systems as well. File replication is addressed in the context of structured peer-to-peer systems by [5] and [8] among others; while [5] addresses lookup performance, [8] is similar to our work in that it addresses the download performance. However it does not seek to explicitly optimize the download time.

Many researchers overlook file replication distribution as an option for improving system performance since the number of replicas of a file in currently deployed systems is not a control parameter but a byproduct of user requests and the downloading process [15]. However, as [9] and [13] discuss, near-optimal replication distributions can be automatically achieved using distributed algorithms that preserve the underlying user request and the download process characteristics.

3 Model

Our peer-to-peer system model consists of *nodes* and *files*. The term *files* represents any generic content while a node corresponds to a user or peer (these terms are used interchangeably in this paper). Each file has a certain request rate associated with it, reflecting user interest in that file. A file can have more than one replica in the system. Nodes have finite local storage space to store file replicas. If a user "requests" a file and if the file is not present in its local storage, the file is downloaded from one or more copies of that file in the network. While our model is independent of the specifics of the search mechanism used by the nodes to determine which peers they can download the file from, we do assume that the download requests for a file are uniformly distributed over all the replicas of that file[1]. Finally, we also assume that a node will always satisfy a request for a file present in its local storage[2]. We make the following two additional assumptions for ease of discussion:

Assumption 1. Each node is homogenous in service capacity.
Assumption 2. Each file is of equal size.

We note here that measurement studies of existing peer-to-peer file sharing systems show considerable variation in file sizes and the service capacity (link bandwidth) of peers [11]. However, our focus is on design alternatives in general peer-to-peer systems and alternative peer-to-peer applications may not share these characteristics of file-sharing systems. In a "true" peer-to-peer system, the peers contribute equal resources to the system so it is reasonable to assume that the service capacities of peers are equal. One can incorporate the cases of unequal file sizes and unequal server capacities in the analysis in this paper to obtain analogous results for the current peer-to-peer file sharing systems if desired. In fact, our analysis and the

[1] This assumption should hold if the search mechanism directs the nodes requesting the file to the nodes that have the file in a uniform manner and/or if the requesting nodes download the file from all nodes that have the file in parallel.

[2] This does not imply that all nodes share files but only that if a node makes a file available, it will satisfy a download request for the file (whereas if a node has a particular file but does not make it available, it will never get a search request for it). In this sense, the number of replicas of a file in the system should be considered as the number of *available* replicas of the file in the system.

derived results directly applies to the special case of non-homogenous peer service capacities observed in current file sharing systems where a significant fraction of peers do not contribute any resources, a phenomenon generally referred to as *free-riding* [11]. Presence of such users just decreases the overall download service capacity available but the fairness of workload distribution among the sharing nodes and optimality of the download performance is maintained; only the load on the sharing nodes increases[3]. We use the following notation for the system parameters:

M = number of nodes in the system
N = number of unique files in the system
K = per-node storage size in number of files
μ =download service capacity of a node (files/unit time)
λ_i = request rate for file i per node[4] (requests/unit time)
$\lambda = \sum_{i=1}^{N} \lambda_i$
n_i = number of replicas of file i in the system

As discussed earlier, the metrics of interest are the variation in the service load at each node and the average download time for a file request.

The service load Λ_j at node j that has files $R_j = \{r_1, r_2, r_3, \ldots, r_K\}$ where $r_i \in \{1, 2, 3, \ldots, N\}$ $\forall i$, in its storage is given by:

$$\Lambda_j = \sum_{i \in R_j} \frac{M\lambda_i}{n_i} . \tag{1}$$

since, given our assumption that the $M\lambda_i$ requests (per unit time) for file i are uniformly distributed over the n_i replicas of file i, each replica of file i serves $M\lambda_i/n_i$ requests per unit time.

The download time for a file depends on three main factors: the size of the file, the traffic on the underlying network layer, and how fast the node(s) serving the content (henceforth, referred to as the server) can upload the file. In practice, however, the upload link bandwidth of the node serving the content is the predominant constraint in downloads. Besides, including the effect of the traffic on the underlying network layer is difficult as downloads are not the only thing carried on the underlying network layer. Therefore, we approximate the download time for equal-sized files by the delay experienced by a download request at the server. For a server to upload the desired content, the upload link as well as the server processing cycles to feed the link should be available. Thus, the delay experienced at the server depends on the overall workload on the server of which the download request rate serviced is only a part. We

[3] This can be seen with the modifications to our analysis of Sections 4 and 5 as follows. If a fraction, $1-\gamma$, of the nodes do not service any downloads, the total storage capacity of the system goes down to γKM and, with proportional file replication, each of the sharing nodes now services λ/γ download requests per unit time, on average. Since each of the sharing nodes operates at uniform utilization level, the average download time is optimal.

[4] λ_i is the request rate for file i in the system averaged over all M nodes i.e. $\lambda_i = \frac{\sum_{j=1}^{M} \lambda_{ij}}{M}$.

do not explicitly model the effect of interruptions by other processes running on the server but rather include them in the randomness of the delay distribution function. Finally, whenever we talk of download time in this paper, it is with the understanding that for larger size files, we seek the download time performance for fixed-size file blocks. Hence, in our model the delay seen by a request at a content server only depends on the download request rate at the server and the download service capacity of the server. To simplify the analysis, we further assume that the delay distribution function at each server is identical. Thus, the average download time for a request serviced by a node j running at utilization factor Λ_j/μ is $T(\Lambda_j/\mu)$. Therefore, the average download time τ for a file request is given by:

$$\tau = \sum_{j=1}^{M} \frac{\Lambda_j}{M\lambda} T(\frac{\Lambda_j}{\mu}) . \tag{2}$$

where Λ_j is as described by Eq. (1), $M\lambda$ is the total download request rate in the system and $T(\rho)$ is the delay distribution at a server running at utilization factor ρ.

4 Download Time

The desirability of load-balancing multiple homogenous servers providing a service is well known[5] (although some exceptions have been noted in literature [3]). Formally, the optimality of operating each node in the system at the same utilization factor requires that the service time at the server be convex in the utilization factor at the server (which is true for majority of the queueing systems [7]). In the context of our system, we can see this as follows.

We can rewrite Eq. (2) as follows:

$$\tau = \frac{\mu}{M\lambda} \sum_{j=1}^{M} \rho_j T(\rho_j) .$$

where $\rho_j = \Lambda_j/\mu$. Since $T(\rho)$ is convex, $\Phi(\rho) = \rho T(\rho)$ is also convex. Using convexity of $\Phi(\rho)$[6] [6]:

$$\Phi\left(\frac{\rho_1 + \rho_2 + \rho_3 + \cdots + \rho_M}{M}\right) \le \frac{\Phi(\rho_1) + \Phi(\rho_2) + \Phi(\rho_3) + \cdots + \Phi(\rho_M)}{M}$$

For ease of the remaining discussion, we state this result as applied to our peer-to-peer file sharing system as the following theorem.

Theorem 1. If the download time $T(\rho)$ at a node is convex in the utilization factor ρ at the node, the average download time in the system is minimized with uniform node

[5] For recent evidence, one only need look at the large number of commercial as well as open-source load-balancers available for data center applications where multiple web servers are used to support popular web sites.

[6] Also referred to as the extension of *Jensen's Inequality* to convex combinations of more than two points [1].

utilization (all nodes in the system operating at an equal utilization factor) when the delay distribution function $T(\rho)$ is identical for all nodes and each node has identical service capacity.

While the desire for uniform node utilization in multiple server systems is driven solely by performance optimization, in peer-to-peer systems it has significance much beyond the download performance optimization. As discussed earlier, in peer-to-peer systems, each peer is an independent entity and fairness in the upload contribution asked of each peer is an important objective.

5 Uniform Node Utilization

The optimal replica distribution for controlled flooding search in unstructured peer-to-peer systems has been found [13] to be one where the number of replicas of each file is proportional to the request rate for that file, i.e. $n_i \propto \lambda_i \ \forall i$ where n_i and λ_i are as defined in Section 3. In this section, we find that the above proportional replication also achieves uniform node utilization and, hence, optimal download time.

Theorem 2. If the download requests for a file are uniformly distributed over all the replicas of the file in the system, each node operates at the same utilization factor, independent of the files it has in its storage, when the number of replicas of file i, n_i, is proportional to the average request rate for file i, λ_i, for all files, i.e.

$$n_i \propto \lambda_i \ i \,. \tag{3}$$

when each file is of equal size.

Proof:
Given M nodes in the system, each with the capacity to store K files, the total amount of storage available in the system is MK. Since $\lambda = \sum_{i=1}^{N} \lambda_i$ is the total request rate in the system averaged over the M nodes, $n_i \propto \lambda_i \ \forall i$ implies that:

$$n_i = \frac{\lambda_i}{\lambda} KM \quad i \,. \tag{4}$$

As noted in Section 3, each replica of file i serves $M\lambda_i/n_i$ requests per unit time if the download requests for files are uniformly distributed over all the replicas of the files. Therefore, for the replica distribution defined by Eq. (4), each replica of file i serves λ/K requests per unit time. Since each node would clearly keep as many files as possible locally, as long as $K < N$, i.e. the per-storage size K is not sufficient to store all the possible N files in the system, each node has K files in its storage. Therefore, the download request rate served by each node is:

$$\Lambda_j = \sum_{i \in R_j} \frac{M\lambda_i}{n_i} = \sum_{i \in R_j} \frac{\lambda}{K} = \lambda \,.$$

Thus, each node serves λ download requests per unit time, the same as the download request rate injected by a node on average. Hence, when the number of replicas of each file is proportional to the request rate for that file, each node, on average, serves an equal number of download requests per unit time. Since each node is assumed to have an equal download service capacity, each node has the same node utilization factor.

Q.E.D.

From Theorems 1 and 2, it directly follows that:

Corollary 1. If the delay distribution function $T(\rho)$ at a node operating at utilization factor ρ is convex and identical at each node, and the download requests for a file are uniformly distributed over all the replicas of the file in the system, the average download time in the system is minimized when the number of replicas of file i, n_i, is proportional to the average request rate for file i, λ_i, for all files, i.e.

$$n_i \propto \lambda_I \quad i\,.$$

under Assumptions 1 and 2.

Notice that while the uniform node utilization condition in Theorem 1 is both necessary and sufficient for download time optimization, the proportional file replication of Theorem 2 is only a sufficient condition as we did not show that such a distribution is the only solution for uniform node utilization. However, such a solution is attractive as it is independent of which files a node keeps in its local storage. Thus, even if the files present at a node keep changing (depending, for example, on the requests made more recently or more frequently) but as long as the number of replicas in the overall system stays proportional to the request rate for the file, the average download time remains optimal. Thus, for example, in a peer-to-peer system where a consequence of finite per-node storage is the possibility of deletion of old files to make space for newly requested files, if the per-node storage management algorithm leads to proportional replication at equilibrium, one has an autonomic content delivery system that automatically adjusts the resource allocation to the optimal setting even as the file request patterns evolve. [13], in its presentation of optimal file replication for controlled flooding search in unstructured peer-to-peer networks, includes such an example system where LRU storage management leads to near-proportional file replication. [9] also discusses distributed algorithms in a similar peer-to-peer system model although they sought to achieve a square-root replication (i.e. $n_i \propto \sqrt{\lambda_i}$) distribution which is optimal for random walk search in unstructured peer-to-peer networks. We emphasize here that our result that the proportional file replication minimizes the download time while, at the same time, ensuring fairness in workload distribution is independent of any search mechanism. Thus our result is applicable to both structured as well as unstructured peer-to-peer systems as long as the underlying system directs the download requests for a file uniformly over all the replicas of that file. Further, this result is independent of any specific file popularity distribution and does not assume any specific request arrival process or service time distribution. Our only assumption is that the download request arrival process and the download service time distribution is such that the delay distribution at a server is convex in the server utilization factor; this is not unduly restrictive as a majority of queueing systems have convex delay distribution [7].

6 Conclusions

In this paper, we address the issue of fairness in the workload distribution and optimization of average download time in peer-to-peer systems. Our first observation is that the download time is minimized when each peer is operating (in terms of download requests it services) at an equal utilization factor. Next, we show that when the number of replicas of each file is proportional to the request rate for that file, each peer operates at a uniform utilization level independent of which files it has in its storage. This result on proportional file replication implying equal workload distribution and optimal download time is independent of any search mechanism and we only assume that the download requests for a file are uniformly distributed over all the replicas of that file in the system. The optimality of the proportional file replication for the average download time also does not depend on any specific request arrival process or service time distribution, but only on the assumption that the delay distribution for a download request is convex in the utilization factor of the peer servicing that request.

References

1. Boyd, S., and Vandenberghe, L.: Convex Optimization, Cambridge University Press, London, 2004.
2. Cohen, B.: Incentives Build Robustness in BitTorrent. In Proceedings of the Workshop on Economics of Peer-to-Peer Systems, 2003.
3. Crovella, M., Harchol-Balter, M., Murta, C.: Task Assignment in a Distributed System: Improving Performance by Unbalancing Load. In Proceedings of ACM SIGMETRICS 1998.
4. Ge, Z., Figueiredo, D. R., Jaiswal, S., Kurose, J., Towsley, D.: Modeling peer-peer file sharing systems. In Proceedings of IEEE Infocom 2003.
5. Gopalakrishnan, V., Silaghi, B., Bhattacharjee, B., Keleher, P.: Adaptive replication in peer-to-peer systems. In Proceedings of the 24th ICSDCS, 2004.
6. Hardy, G. H., Littlewood, J. E., Pólya, G. Inequalities, Cambridge University Press, London, 1952.
7. Kleinrock, L.: Queueing Systems, Volume I: Theory. Wiley Interscience, New York, 1975.
8. Kubiatowicz, J., Bindel, D., Chen, Y., Eaton, P., Geels, D., Gummadi, R., Rhea, S., Weatherspoon, H., Weimer, W., Wells, C., Zhao, B.: OceanStore: An Architecture for Global-scale Persistent Storage. In Proceedings of the 9th ASPLOS, 2000.
9. Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S. Search and Replication in Unstructured Peer-to-Peer Networks. In Proceedings of the 16th ACM ICS 2002.
10. Qiu, D., Srikant, R.: Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks. In Proceedings of ACM SIGCOMM 2004. .
11. Saroiu, S., Gummadi, K. P., Gribble, S. D. A Measurement Study of Peer-to-Peer File Sharing Systems. In Proceedings of MMCN 2002.
12. Stutzbach, D., Zappala, D., Rejaie, R.: Swarming: Scalable Content Delivery for the Masses. University of Oregon Computer and Information Science Technical Report, UO-CIS-TR-2004-1, 2004.

13. Tewari, S., Kleinrock, L.: Analysis of Search and Replication in Unstructured Peer-to-Peer Networks, In Proceedings of ACM SIGMETRICS 2005.
14. Yang, X., de Veciana, G.: Service Capacity of Peer to Peer Networks. In Proceedings of ACM INFOCOM 2004.
15. Zou, L., Ammar, M.: A File-Centric Model for Peer-to-Peer File-Sharing Systems," In Proceedings of ICNP 2003.