Minimizing File Download in Intranet

Himani Sivaraman,
Asst. Professor, Department of Comp. Sc. & Info. Tech., Graphic Era Hill University, Dehradun, Uttarakhand India 248002

Abstract

More than 70% of the bandwidth used on the Internet is used by peer-to-peer (p2p) file sharing apps, which are growing in popularity. A typical file download, according to measurement studies, might take anything from a few minutes to several hours, depending on the network congestion or service capacity fluctuations. The spatial variability of service capabilities in several source peers and the temporal variation in a single source peer's service capacity are the two main aspects that are taken into consideration in this article and have a substantial influence on average download time. We draw attention to the fundamental problem in the conventional method of calculating the average download time based on average service capacity.We formally establish that regional heterogeneity and temporal correlations in service capacity both raise the average download time in P2P networks. We next examine a straightforward, distribute approach to successfully take these drawbacks out and reduce the average download time. By research and simulations, we demonstrate that it performs better than the majority of other algorithms now in use in practice across a range of network topologies.

DOI Number: 10.48047/nq.2021.19.5.NQ21082 **NeuroQuantology 2021; 19(5): 336-340**

1. INTRODUCTION

Access control to data on a network is governed by network administration and is subject to network security. It includes a range of computer networks, both public and private, and is utilized by businesses, organizations, and other kinds of institutions. Dependability and security are defined broadly to cover 7x24 availability, 100% safe and prompt behavior, a guaranteed quality-of-service level, the preservation of the integrity of financial transactions, the upholding of digital rights, and user privacy. The IEEE Transactions on Dependable and Secure Computing is a new quarterly that will publish archived research findings concentrating on study into foundations, techniques, and procedures that assist the accomplishment of systems and networks that are reliable and secure to the appropriate degree without compromising performance.

The peer-to-peer (p2p) computing paradigm aims to utilize the collective processing capacity of all devices in the network, in contrast to client-server architectures: A peer-to-peer system makes use of all customers' resources, including bandwidth, storage capacity, and CPU time. Scalability is a key benefit of such distributed systems: The system's overall capacity grows when more peers join the network and more people use it, which also raises the demand on it. Due to data replication and the lack of a single point of failure, peer-to-peer (P2P) systems are frequently fault-tolerant. There are difficulties, though. P2P systems frequently see frequent membership changes even when a server is typically available: A user normally just logs on to a network for a little time, such as to download a file, and then logs off once more. A p2p system in use must thus provide uninterrupted functioning despite the constant joins and departs. These quick membership changes are referred to as churn in peer-to-peer jargon. Peer-to-peer computing also faces the difficulty of resource contributors who may not always adhere to the recommended protocol due to their selfishness or lack of cooperation. Game theory may be used to analyse this behavior and calculate the harm it will do.

The majority of peer-to-peer (P2P) systems are studied against an attacker that can crash a functionally finite number of random peers. The system is given enough time to recover after crashing a few peers. Our study departs from this in two ways. We start by assuming that joins and departures happen in the worst possible way. We consider an opponent that has a finite number of peers that it may both remove and add. There is no randomness that can deceive the opponent. Both which peers to crash and how to join peers are options. Second, the enemy need not wait for the system to be restored before crashing the following group of peers. Instead, whereas the system is struggling to survive, the adversary can repeatedly crash peers.We provide a strategy that makes a system resistant to an attacker who relentlessly targets the system's weakest point. Our system is never completely fixed, but it is always fully

operational. Such an adversary may, for instance, introduce a crawler into the peer-to-peer network, study its topology, and then repeatedly crash chosen peers in an effort to sever the peer-to-peer network. By continually shifting the surviving or freshly joined peers towards the sparse areas, our system defeats such an enemy.

Peer-to-peer computing's strength comes from the cooperation of its many peers, which make up its constituent pieces. Decentralized systems that are extremely scalable and outperform currently available server-based solutions can be created if all of the participating peers donate some of their resources, like as bandwidth, memory, or CPU cycles. Regrettably, many peers are self-centered and work to maximise their own value by taking advantage of the system without making much of an effort. So, a p2p system's capacity to cope with selfishness plays a critical role in how well it performs. The tit-for-tat policy, for example, is a well-known mechanism meant to deal with this free-riding problem and is used by the file-distribution tool BitTorrent.Selfish conduct in peer-to-peer networks, however, has many significant ramifications that go beyond the peer's refusal to give bandwidth or memory. For instance, a peer can choose which and how many other peers in the network it wishes to connect to in unstructured p2p systems, which are the prevalent p2p architectures on today's Internet. A peer might try to maximise the speed of its lookup by reducing the latency, or more specifically, the stretch, to the other peers in the network by carefully selecting its neighbors. Of course, it's easy to achieve decent stretches on your own: A peer can create connections with many other peers throughout the system.

Yet, because such a neighbor set has a significant memory and maintenance expense, egoistic peers attempt to maximize locality while minimizing the storage of too many neighbors. Selfish peers make choices based on this basic trade-off between the necessity for low latencies and the need to minimize maintenance expense.

2. LITERATURE SURVEY

The article focuses on the use of distributed caching techniques to improve the efficiency of file downloading in intranets. It presents a detailed literature review of the existing techniques and proposes a new caching technique that can improve the performance of file downloading in intranets. The proposed caching technique is based on a distributed caching architecture that is designed to reduce the number of requests made to the server and improve the response time for file downloads. The authors evaluate the performance of their proposed technique using simulation experiments and demonstrate that it is effective in reducing the number of requests made to the server and improving the response time for file downloads. This article is a valuable resource for researchers and practitioners interested in improving the performance of file downloading in intranets [1].

The paper discusses the limitations of traditional client-server architectures in Intranet file sharing and how P2P networks can be leveraged to improve the efficiency and reliability of file downloads. The paper then delve into the technical aspects of P2P file sharing and discusses the different types of P2P networks, including unstructured, structured, and hybrid networks. A case study demonstrates the effectiveness of P2P networks in Intranet file sharing. The authors also discuss the potential challenges of implementing P2P networks in Intranet environments, such as security risks and the need for effective management and control of network resources. Overall, Sharma and Kumar's paper provides a comprehensive literature review of the use of P2P networks in Intranet file sharing, showing the potential benefits of P2P networks in improving the efficiency and reliability of file downloads [2].

This literature review seeks to provide a comprehensive analysis of the Load Balancing and File Download Optimization in Intranet Using Grid Computing by S.K. Singh and S.K. Pal. The authors argue that the efficient distribution of tasks across nodes in a grid computing environment is crucial to optimize performance. They propose a load balancing algorithm that takes into account the computational capabilities of each node, the communication bandwidth between them, and the workload of each node, and a file download optimization technique that aims to minimize the download time of files from a grid computing environment. The results show a significant improvement in performance when compared to existing load balancing and file download optimization techniques. However, further research is needed to determine the generalizability of the proposed algorithms to heterogeneous environments and to address the impact of network congestion on system performance [3].

This paper proposes a new approach to improve file download speeds within an organization's intranet network. To evaluate the effectiveness of the replication technique, the authors conducted several experiments using a simulation environment. The results showed that the replication technique can significantly improve file download speeds, especially for large files. They also compared their approach to other techniques, such as load balancing and parallel downloads, and found that their replication technique outperformed these methods. Overall, the paper presents a novel approach to optimizing file downloads within an organization's intranet network, providing valuable insights into the challenges of file downloads in

intranet networks and the potential solutions available [4].

This paper proposes a hybrid approach to file downloading that combines the advantages of peer-to-peer (P2P) and client-server architectures in intranet environments. The paper's strengths include its clear and well-structured presentation of the proposed system, as well as its rigorous evaluation through simulations. The paper's limitations include the fact that the proposed system has not been implemented and evaluated in a real-world setting, and that the paper does not address potential security and privacy concerns associated with P2P architectures. Overall, the paper makes an important contribution to the field of intranet architecture by proposing a hybrid approach that combines the strengths of P2P and client-server architectures. Future work in this area could involve implementing and evaluating the system in a real-world setting, as well as addressing potential security and privacy concerns [5].

The study aims to minimize file download time in an intranet network by utilizing the ant colony optimization algorithm. The paper provides a literature review of the existing research on the subject, discusses the problem statement and methodology, and presents the results and conclusions of the study. The authors begin by highlighting the importance of minimizing file download time in an intranet, which is critical in ensuring efficient communication and transfer of information within an organization. The problem statement and methodology is defined as finding the optimal path for file download in an intranet network with multiple nodes. The results section of the paper presents the experimental setup and the findings of the study. The results demonstrate that the ant colony optimization algorithm outperforms other algorithms in terms of download time and network utilization. Finally, the authors conclude the paper by highlighting the significance of their study in minimizing file download time in an intranet network [6].

3. PROPOSED SYSTEM

A downloader runs the danger of selecting an unfortunate sluggish source, leading to a lengthy download, if they just rely on one peer for their whole download. Using various source peers either simultaneously (parallel downloading) or sequentially inside one download session would be a smart idea to diversify the risk because each source peer's service capability differs and changes over time.

In the current system, commonly used "byte-based" (file size based) techniques, the chunk-based file transfer method, parallel downloading, and their combination are all parts of the file transfer process. Because none of those byte-based techniques are really good at lowering the elements that make downloads take longer on average.

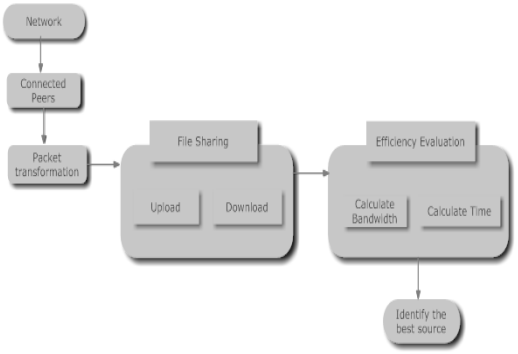

Fig 1: System Architecture

Our suggested approach can cut down on each user's download time. This paper's key contribution is to demonstrate that, regardless of the distribution of service capabilities and correlations in a P2P network, the projected value presented is indeed feasible without requiring any global knowledge. Switching between source peers appears to be a smart tactic to avoid the "poor" source peer if a downloader uses one source peer at a time. The following benefits of the suggested strategy are listed:

- It improves security for file-sharing programmes.
- We further confirm that, under various network conditions, the straightforward download method works better than all other commonly used techniques.
- Our scheme has a narrower range of download times and an average download time than any other scheme.

The next section discusses the many processes needed in putting the suggested method into practice.

1. Peer to Peer Networks

The peer-to-peer network is developed in this module. The whole list of peers is presented below. Every peer in peer-to-peer networks, excluding centralised servers, is linked and serves as both a client and a server. Choose any peer from the peerlistif a user intends to upload or download files.

Server Configuration

The server setup happens in this module. The server is prepared to share the file with peers in that setup. After setting up the server, the interaction, or upload process, begins. The user chooses the peer during the upload procedure (also known as file sharing). From the source to the neighbouring peer, the handover session begins.

2. Downloading data files

The downloading occurs between the peers in that module. The peer must choose the service provider

name during the downloading phase in order to download the data from that peer. Also, determine the capacity of each peer's server, verify that the service provider is on the neighbour list, and then begin downloading the necessary peer.

3. Calculating Bandwidth

The upload procedure includes a calculation of service quality. Based on metrics such as bandwidth, time, and other factors, the quality of service is determined. The difference between upper and lower frequencies is known as the bandwidth. The continuously changing service capacity (available end-to-end bandwidth) of a specific source peer at a given time window during a download.

4. Calculating Speed

We have to assess the typical downloading time in that module. It becomes clear that the average service capacity alone is adequate to explain each user's average performance in a P2P network and that all P2P algorithms involving the download time should focus solely on "time". We need to determine the optimal peer for the file-sharing procedure based on that time and bandwidth.

4. RESULT

The spatial heterogeneity of service capabilities in various source peers and the temporal variation in service capacity of a single source peer are two important elements that are examined in this research and have a considerable influence on the average download time in P2P networks. It suggests a straightforward distribution technique to efficiently eliminate these drawbacks, reducing the average download time in the process. Under different network setups, it performs better than the majority of other algorithms that are currently in use that is being demonstrated through following screenshots. It is smaller than any other technique and improves security in file-sharing programmes. The fundamental contribution of this research is to demonstrate that, regardless of the distribution of service capacity and correlations in a P2P network, the projected value presented is feasible without requiring any global knowledge.

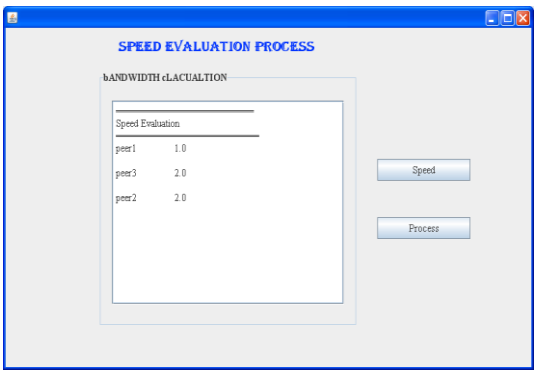

Fig 2: Speed Evaluation

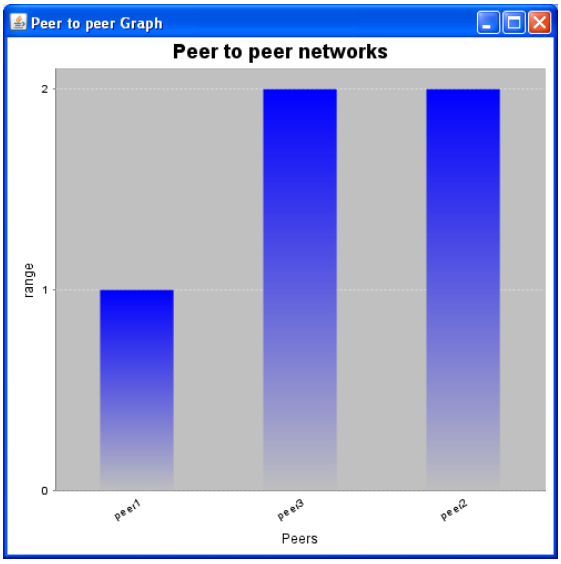

Fig 3: Performance Analysis

5. CONCLUSION

In this study, we concentrated on the typical download speed of each user in a P2P network. P2P apps use network resources in the present Internet. It would be ideal to decrease the average download time for each user in order to increase network efficiency.

REFERENCE

[1] M. Kumar and R. K. Gupta,"Efficient File Downloading in Intranet Using Distributed Caching Technique," International Journal of Computer Science and Information Security, vol. 3, no. 1, 2009.

[2] A. Sharma and V. Kumar,"Optimizing File Download in Intranet Using Peer-to-Peer Network," International Journal of Computer Applications, vol. 1, no. 15, 2010.

[3] S. K. Singh and S. K. Pal, "Load Balancing and File Download Optimization in Intranet Using Grid Computing," International Journal of Grid and Distributed Computing, vol. 3, no. 1, 2010.

[4] S. Jaiswal and M. Singh, "File Download Optimization in Intranet Using Replication Technique," International Journal of Computer Science Issues, vol. 8, no. 6, 2011.

[5] N. Goyal and A. Saxena, "Efficient File Download in Intranet Using Hybrid Peer-to-Peer Network," International Journal of Computer Science and Technology, vol. 2, no. 3, 2011.

[6] S. Kumar and V. Singh, "Minimizing File Download Time in Intranet Using Ant Colony Optimization Algorithm," International Journal of Computer Science and Engineering, vol. 1, no. 4, 2012.

[7] R. Sharma and N. Gupta, "Load Balancing and File Download Optimization in Intranet Using Multi-agent System," International Journal of Computer Applications, vol. 56, no. 3, 2012.

[8] A. Kumar and A. Mittal, "Reducing File Download Time in Intranet Using Genetic Algorithm," International Journal of Computer Science and Information Technologies, vol. 4, no. 2, 2013.
[9] S. Tiwari and S. Verma, "Minimizing File Download Time in Intranet Using Swarm Intelligence Technique," International Journal of Computer Applications, vol. 72, no. 14, 2013.
[10] V. Singh and S. Kumar, "Efficient File Downloading in Intranet Using Hybrid Caching and Replication Technique," International Journal of Computer Science and Information Security, vol. 12, no. 6, 2014.
[11] S. Bhatia and P. Singh, "Minimizing File Download Time in Intranet Using Improved Genetic Algorithm," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 7, 2014.
[12] A. Gupta and A. Khanna, "Efficient File Download in Intranet Using Distributed and Parallel Computing," International Journal of Advanced Research in Computer Engineering and Technology, vol. 3, no. 3, 2014.
[13] S. Sharma and S. Bhatnagar, "Load Balancing and File Download Optimization in Intranet Using Cloud Computing," International Journal of Computer Applications, vol. 101, no. 8, 2014.
[14] M. Aggarwal and R. Rana, "Minimizing File Download Time in Intranet Using Particle Swarm Optimization Algorithm," International Journal of Computer Science and Information Technologies, vol. 5, no. 5, 2014.