Increasing service capacity of peer-to-peer file sharing networks by using a decentralized reputation system

MSc Research Project
MSc in Cloud Computing

Ibrahim Ayodeji
Student ID: x20227329

School of Computing
National College of Ireland

Supervisor: Shivani Jaswal

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Ibrahim Ayodeji
Student ID:	x20227329
Programme:	MSc in Cloud Computing
Year:	2022
Module:	MSc Research Project
Supervisor:	Shivani Jaswal
Submission Due Date:	15/08/2022
Project Title:	Increasing service capacity of peer-to-peer file sharing networks by using a decentralized reputation system
Word Count:	5728
Page Count:	21

Signature:	
Date:	19th September 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Increasing service capacity of peer-to-peer file sharing networks by using a decentralized reputation system

Ibrahim Ayodeji
x20227329

Abstract

Using peer-to-peer networks has allowed users to share resources amongst themselves. But to maintain the required level of quality of service, it is essential to maximize service capacity. Several Solutions for enhancing service capacity in peer-to-peer file sharing networks are examined in this research paper. These papers attempt to deal with issues such as free-riding, which places a heavy burden on peer-to-peer systems, and their ability to provide services and file availability, where some files could become inaccessible due to their age or the fact that no seeds are hosting them. This paper offers a solution based on using decentralized reputation system to provide a viable trustless solution to solve this. In this paper a system is developed based on the Ethereum blockchain where votes are cast to by peers to identify the most credible nodes for downloads. The system is tested using a simulation engine which imitates the behaviour of real peers against a variety of parameters in order to prove the effectiveness of the solution.

1 Introduction

Client-server systems, which commonly use the file transfer protocol (FTP) or Hypertext Transfer Protocol, are the norm for online file sharing techniques. Users must connect to a server through a client and request a file typically through an HTTP or HTTPS connection. The volume of requests, the server's bandwidth, the distance from the user, and other variables all have a significant impact on transfer speed. The downside of a centralized system is that if a file is accidentally or purposely deleted from the server, the network will no longer have access to it. These problems are partially resolved by peer-to-peer (P2P) file sharing networks. Network members distribute files amongst themselves via peer-to-peer file sharing systems.

Peer-to-peer networks can be difficult to define, particularly when compared to conventional client-server networks. A peer-to-peer network, according to [1], is a distributed network where users share their own hardware resources. These resources, which are needed to provide services and content, may include storage, processing power, and more. Each node, referred to as a *peer*, serves as both a client and a server.

A peer that has the entire file and is actively distributing is known as a seed, whereas a peer that has the complete file and is still downloading is known as a leecher. Due to its versatility and low bandwidth requirements, this has gained a lot of popularity [2]. Peer to peer networks use up about between 35 and 90 percent of all the online traffic. Out of all the networks, BitTorrent has been the most prevalent [3]. In these networks files are always available as long as there is someone sharing.

Peer-to-peer networks that are categorized as structured have fixed connections amongst peers while maintaining information about the available resources. Structured peer-to-peer networks have additional restrictions on the types of nodes and where data should be placed, but they feature efficient queries thanks to Distributed Hash Tables, with each peer in charge of a portion of the overall key space. Fast insertion and querying are made possible by DHT, but table maintenance is challenging due to regularly changing peers [4]. Unstructured peer-to-peer networks, on the other hand, are more adaptable and less limited than structured networks. Their connections may be hierarchical in centralized or hybrid networks or flat in decentralized networks. Techniques like flooding, random walking, and time-to-leave are used to make queries. Centralized networks, like BitTorrent, rely on central entities to perform certain network functions that have single points of failure.

A network's main objective is to improve service quality for all its peers. The network's service capacity, the average upload capacity of nodes in the network, can be used to quantify this. One may calculate service capacity, which is the average download throughput per peer, by dividing the aggregated upload service capacity by the number of peers that are downloading [5].

Making sure every node in the network contributes as much as they should is a significant factor. Free-riders are nodes that fail to add their quota [6]. Free-riders cause the network's overall bandwidth to decrease. Reciprocative techniques, including Tit for Tat, have been used in networks like BitTorrent. Peers in Tit for Tat only exchange blocks with peers who send to them more frequently [6]. This system can be implemented locally and does not require a centralized or distributed process. This view is myopic because it just considers the current upload rates and ignores how past rate history might affect future upload rates, resulting in subpar network performance. Additionally, it prevents new peers from accessing files, hence networks frequently utilize an optimistic unchoking technique to ease network entry. Free-riders take advantage of this to enter the network without contributing any resources. Peers' contributions to the network and other long-term data about them must be considered. A distributed reputation system, or the ability for any peer to view and update this information, is also required.

Examining several papers in the files, a wide array of solutions have been proposed. While some authors have developed solutions based on novel seeding mechanisms [7, 8] others proposed the use of unique unchoking mechanisms [2, 7, 8, 9, 10, 11, 12, 13]. Systems based on incentive mechanisms were also created by a number of other writers [2]. The majority of papers have considered increasing service accessibility from the standpoint of the seeders, by lowering free riding, or increasing the number of seeders by employing credit incentives.

However, it doesn't appear that any study has attempted to use a blockchain-based decentralized reputation system to improve service availability on peer-to-peer file sharing networks. Satoshi Nakamoto unveiled the revolutionary blockchain technology in 2008. The acceptance of cryptocurrency and other decentralized applications is evidence of their potential. Fully decentralized solutions can be implemented by utilizing blockchain technology.

The goal of this study is to determine whether a distributed reputation and incentive payment system can increase the service capacity of peer-to-peer file sharing resources. Peers would be able to only share with peers with positive ratings while excluding free-riders from the network if a reputation-based system was used to track peer reliability scores in a blockchain distributed ledger. Therefore, it appears likely that this approach will be able to increase service capacity.

1.1 Research Question

Can we increase the service capacity of peer-to-peer file sharing networks by using a decentralized reputation system?

1.2 Content

The first section of this essay reviews previous research done by other scientists. Discuss the many approaches that have been employed to enhance scalability in this area. After that, an approach outlining the algorithms and technologies to be used is suggested. Finally, ethical issues are discussed and areas for further research are defined in the concluding remarks.

2 Related Work

This section summarizes research that has been done to reduce free-riding and reward altruistic behavior in peer-to-peer file sharing networks in an effort to increase service availability. Using direct retribution methods, typically by creating choking algorithms, was suggested by certain writers [7, 8, 14, 15, 16, 17]. These, however, are skewed and do not take into account the experiences of other peers. Since then, interest in indirect reputation systems has grown, according to [10, 12, 18, 19].

As a way to encourage better sharing, several writers have also discussed payment-based techniques [10, 13], and some have used game theoretical approaches to simulate these systems [2, 9, 20, 21]. We will discuss these strategies in the following section.

The difficulties impacting service capacity that this paper seeks to address are discussed in this section. Following an overview of the mechanisms used in the solutions, it delves into the many approaches researchers have taken to address these problems. Finally, broad judgments about the review are derived.

2.1 Free-Riders

Free-riders are peers who try to consume shared resources in peer-to-peer networks without giving anything in return by exploiting weaknesses in the protocols. As was previously mentioned, optimistic unchoking allows free riders to take advantage of the BitTorrent network. By examining the incentives in the BitTorrent choke mechanism, [22] demonstrated that freeriding is possible in BitTorrent since it is ineffective in adequately rewarding and punishing free-riders. According to [23], a free rider can get one-fifth the download rate of normal peers by optimistically unchoking. [24] draws the conclusion that methods that penalize free-riders offer a significant boost in service capabilities in contexts with poor generosity. Therefore, a crucial strategy for achieving the objective of this research is to discover a practical way to prevent or limit free-riding. A reward system can promote more suitable seeding behavior in participant nodes, which would otherwise behave selfishly.

A reward mechanism can promote more suitable seeding behavior in addition to penalizing free-riders by incentivizing participant nodes who might otherwise act selfishly.

There are two sorts of incentive systems. These are known as payment-based mechanisms and retribution mechanisms.

2.2 Retaliation Mechanisms

The resources that peers can get from the network in retribution mechanisms are determined by their prior behavior. It is possible for this process to be direct or indirect. Direct interaction between peers determines what one peer may obtain from another peer. Which peers are allowed to download from another are chosen using unchoking techniques. A popular direct unchoking strategy is called "tit for tat," where peers upload to peers from whom they have downloaded more [25]. Indirect methods provide a peer access to resources by relying on their global reputation.

2.2.1 Direct Retaliation

Several authors have tried to devise more optimistic unchoking algorithms [7, 8, 14, 15, 17]. By using the unchoke histories of its neighbors, [14] proposed an algorithm to reduce free-riding. In order to let a peer choose the most appropriate peer to unchoke, it calculates a gain factor that takes into account how much download bandwidth it is likely to gain by doing so. Freeriders are less likely to be unchoked because their gain factor would be minimal.

[7] discovered that free-riders frequently consume necessary network bandwidth by directly accessing seeds as well as by utilizing optimistic leecher unchoking. To counteract this, they created a modified uploading algorithm that took into account the progress of the leecher's downloads. Leechers who are either just starting their downloads or nearing the end are given priority during this seeding. Free-riders rely more on seeds than leechers do, so cutting off their access to seeds for the majority of the downloading process significantly lowers their efficiency and frees up more bandwidth for reliable seeders. Additionally, this has the effect of reducing the ramp-up time for real leechers.

Multiple seeding strategies have also been implemented to complement the unchoking strategies. [17] proposed a team mechanism that sets a limit on the optimistic unchokes as the level of collaboration with peers who upload data at a similar rate rises. In groups of peers, this protocol dynamically groups peers with comparable upload bandwidth. Members of the team primarily fulfill their data download needs within their own group, and they only engage in optimistic unchokes when absolutely necessary. This team-based approach consequently improves peer performance through explicit cooperation. Since only current upload rates are taken into account when choosing which peers to unchoke, direct retribution techniques are myopic and have been demonstrated to be less effective [10]. Additionally, because performance histories are not shared between pairs of users who contribute, users who switch peers won't immediately face consequences. This reduces the potential of the system to increase service capacity. As a result, several writers have thought about reputation systems that use historical utilities to provide actionable foresights.

2.2.2 Indirect Retaliation

Indirect reputation mechanisms have emerged as a result of the flaws in direct retaliation. In indirect reputation systems, local score gathering and global score distribution are two main roles.

- Local Score Aggregation: Peers gather data on service quality from other peers during the local aggregation stage and convert it to scores.
- Global Score Dissemination: These outcomes are then kept in a global storage system that any peer may simply access throughout the stage of global score distribution.

To accomplish this, other writers have developed systems that follow multiple methods. PowerTrust is a reputation system that [12] created for peer-to-peer networks. By gathering input from peers' interactions with one another and utilizing the power-law distribution of peer feedbacks, the system creates a global reputation score for each peer. On top of the peers, a virtual network called the Trust Overlay Network is constructed to enable feedback exchange. The PowerTrust system then uses this data to generate aggregate scores.

[19] created two centralized systems for connecting a user to a reputation score that is dynamically updated based on the network resources they contribute. To quickly retrieve reputation, the system employed local storage. The first method, referred to as debit-credit reputation computation (DCRC), awards peers with credits for content contribution and debits for content download. The second method, known as credit-only reputation computation (CORC), does not provide any debits for supplying content but instead credits peer reputation ratings.

These systems have a lot of potential, but they lack the immutability and decentralization that blockchains may offer when it comes to reputation storage.

2.2.3 Blockchain Reputation Systems

The total decentralization of reputation systems in peer-to-peer networks is made possible by the use of blockchain technology. Several applications have already used blockchain-based reputation systems [13, 26, 27].

The trust mechanisms for Fog devices have also been addressed using blockchain reputation systems. A blockchain-based decentralized reputation system for Fog-IOT architectures was created by [13]. A call to a voting function on the management ethereum smart contract is made each time an IOT device interacts with a Fog node. To maintain the system's integrity, the trustworthiness of the rater is also monitored. A blockchain auction and reputation system-based energy trading plan for electric vehicles was proposed by [26]. For electric cars, it employs a reverse bidding system to select a supplier based on price and reputation data kept on the blockchain.

[10] designed a method to restrict the system's sharing of illegal files. A node requests a file from peers, receives their replies after a verification process, and then uses the findings to create a reputation index for those peers. By employing this method, the network's unsuccessful transaction rate was cut by a factor of 20. Having a longer-term distributed reputation approach was a significant problem that was not resolved, and this research intends to establish one.

Large amounts of data may now be stored in a decentralized, trustless environment thanks to the development of distributed ledger systems based on blockchain technology.

2.3 Game Theory Modelling

The issue of resource sharing has been examined in a number of studies using a game theoretic perspective. [2, 9, 20, 21]. Mathematical models are used in game theory as

Papers	Type	Limitations
[7, 8, 14, 15, 17]	Direct Retaliation	Myopic such that only current upload rates are considered
[10, 12, 18, 19]	Reputation Based	Does not offer the decentralization of reputation storage and immutability that blockchains provide
[10, 11, 13, 26, 27]	Blockchain Reputation Based	
[2, 9, 20, 21]	Payment Based	

Table 1: Summary of Literature Review and Research Niche

a theoretical framework to depict the interaction of rational actors. In a conventional centralized system, it is thought that system users collaborate to accomplish a shared objective. On the other hand, in peer-to-peer networks, peers compete for the same resources. This strategic behavior may be represented as a non-cooperative game in which participants compete to make the most money possible. These tactics for peer-to-peer networks involve choosing which peers to unblock, where to download from, what files to seed and for how long. Numerous writers [20, 21] have utilized Nash equilibrium to illustrate this non-coorporative game.

With the use of game theory models, we can deal with rational agents looking to maximize their profit from the system and demonstrate the effectiveness of our suggested solution. In accordance with the pricing strategies employed, there ought to be a link between service capacity and the average profit realized.

2.4 Payment-Based Mechanisms

In payment based incentive mechanisms, peers get paid to upload content while other peers pay to download content. A virtual currency is used to facilitate these transactions. Several authors have developed different strategies.

In peer-to-peer networks, payment incentives have been employed for a variety of purposes. Payment incentives were employed by [28] to increase the availability of peers in the network. They noticed that because peers are anonymous, utilizing incentive systems based on reputation was insufficient. According to this theory, the message's originator must pay each peer along the path it travels. To make a payment, the initiator buys a certificate that the system bank blindly verifies. This ensures that the certificate cannot be traced to the initiator; instead, only the terms are known, and the money may be utilized if they are available.

An incentive system based on tokens was presented by [21] for peer-to-peer apps that transmit video. They employed layered video coding, where an upgraded layer was only made available upon request and a base layer was made available to all peers. Peers receive token payments for accepting requests for streams of greater quality. By employing various techniques, these writers have demonstrated the effectiveness of payment-based methods.

Papers	Method	Type	Game theory modelling
[7]	Peers at the beginning and end of downloads get priority	Direct Retaliation	No
[8]	Time based seeding strategy	Direct Retaliation	No
[12]	Power law distribution of peer feedbacks	Reputation Based	No
[13]	Voting Function on Smart contract	Blockchain Based Reputation	No
[29]	Request-adaptive incentive	Payment Based	Yes
[21]	Layered file encoding system for different request tiers	Payment Based	Yes
Proposed Solution	decentralized blockchain based reputation system with reverse auction payment mechanism	Hybrid Reputation and Payment Based	Yes

Table 2: Summary of Important Papers

2.5 Research Niche

There have been several articles investigating various options for increasing service capacity. A synopsis of these works is shown in Figures 1 and 2. Table on displays the articles that were examined and the suggested method of implementation. The essential articles utilized to determine the effectiveness of the suggested approach are shown in Figure 2.

The majority of papers adopted one of three strategies: direct reputation systems that store user information globally, indirect retaliation mechanisms that only take into account local interactions with specific peers when determining who to unchoke, or payment-based systems that use tokens to reward seeders. As a result, we may take the finest qualities from existing systems and create our own innovative system. The approach that is being suggested makes use of the direct reputation system, which is set up on the blockchain using smart contracts.

3 Methodology

3.1 Blockchain Technology

The strategy will monitor peer reputations via a blockchain network. A blockchain is a distributed ledger system that was first created by Satoshi Nakamoto. [30]. It is composed of a number of blocks that are chronologically related to one another. A cryptographic chain is created because each block consists of a set of transactions, some block-specific information, and the hash of the preceding block. Only specialized nodes known as miners may add new blocks since it is often a computationally expensive and probabilistic operation that demands a lot of investment. Because altering the contents in any block invalidates all earlier blocks, this chain structure is what enables a blockchain resistant to alteration. If a block is public, any agent may query the ledger, making it possible for any peer to verify it. The method was developed initially for the bitcoin cryptocurrency but has subsequently been used in a broad range of solutions.

On the blockchain, smart contracts are applications that run automatically when specific criteria are satisfied. They enable trustless agreements, ensuring that the result is known to all parties [31].

3.2 Reputation System

3.2.1 Proposed Solution

The primary goal of the research is to create a framework for monitoring peer reputation on a network, which will reduce free-riding and increase network capacity with no additional expense. A "transaction" is started in the proposed system whenever a peer requests a piece of data from another peer. It's possible for this transaction to succeed or fail. If the result is positive, a normalized completion time to file size ratio is used to determine the service's quality score. A negative score is assigned if the request does not obtain the proper response. In order to update the seeder's reputation points, the peer then makes a call to the smart reputation management smart contract. The local reputation databases of all peers that subscribe to the reputation update event are updated.

3.2.2 Architecture

The proposed reputation system consists of two main modules:

- Client Module: This runs on the client-side to facilitate connection to the network. The duties of this module are:
  - Collection and Analysis: When a peer requests a chunk from another peer and the request is fulfilled, the peer not only records the size of the chunk but also the latency of the request. Requests that go unmet are also tracked. The proportion of a chunk's size to its latency is normalized. The score is 0 if the request is not handled. A number of chunks are downloaded and an average score is produced before being delivered in order to reduce the amount of requests to the blockchain.
  - Submission: Calls to the reputation system smart contract are made using the score produced by the analysis and the peer identity that handled the request.
  - Reputation Queries: The reputation of a user's neighbors is collected from the reputation system smart contract and used to help it select who to unchoke or download from in order to make the best option possible.
- Smart Contract: The smart contract system deployed on the blockchain has two functions:
  - Storage: If the servicing peer ID has already been stored, the storage function verifies its save peer identifications mapping when it is invoked. In the event that it is not, it is added with a value that is an array of reputation scores. Voting is restricted to peers with good reputation ratings.
  - Dissemination: The contract can be used to query the reputation score of a user by using their ID.

The main algorithm for this module is presented in Algorithm 1 & Algorithm 2. In algorithm 1, the user collects a score from a voter and adds it to the seeders score.

Algorithm 1 Client Collection and Submission Algorithm

```
1: procedure COLLECT AND SUBMIT SCORE($servicerId, chunkId$)    ▷ The module takes the selected peer and requests for the required chunk using the chunkId
2:     $startTime \leftarrow getTime()$
3:     $chunk \leftarrow requestForChunk(servicerId, chunkId)$
4:     $latency \leftarrow getTime() - startTime$
5:     if $!chunk$ then
6:         return Negative score depending on the size of the chunk
7:     else
8:         return $normalize(getchunksize(chunk)/latency)$
```

In algorithm 2, the smart contract adds the score submitted from the client using algorithm 1 to the seederId collected and adds it to the peers mapping.

Algorithm 2 Reputation Smart Contract

```
1: procedure UPDATE REPUTATION($voterId, servicerId, score$)
2:     $voterReputation \leftarrow peersMapping[voterId]$
3:     if $!voterReputation$ or $avg(voterReputation) \leq threshold$ then
4:         return failed
5:     else
6:         if $!peersMapping[servicerId]$ then
7:             $peersMapping[servicerId] \leftarrow [rating]$
8:         else
9:             $push(peersMapping[servicerId], rating)$
```

3.3 Technologies Used

3.3.1 Ganache

To provide results without the unpredictability of using a live blockchain, the suggested implementation would be built and executed on a local environment. Consequently, it has been decided to employ Ganache. Users may build private blockchains in their local environments using Ganache. It allows users to develop and test their apps in a secure and controlled environment without having to interface with the real Ethereum network. Both a client UI and a CLI may be used with it.

3.3.2 Solidity

A high level object-oriented programming language called Solidity is used to create smart contracts for the Ethereum network. Its syntax is comparable to those of languages that also use curly braces, such as Java and JavaScript. It is statically typed and supports both user-defined and primitive types.

3.3.3 Solc

By converting Solidity code into bytecode that can be installed on an Ethereum Virtual Machine, Solc is a compiler. Python and JavaScript are only two examples of the languages in which it may be utilized.

4 Implementation

The implementation is written in three main applications. The architecture would consist of three parts. The frontend application, the signalling-backend service and the blockchain module.

4.1 Peer-to-peer Network

The architecture consists of nodes that communicate with each other directly. Each node represents an individual connection to the network. The connections with other nodes is facilitated using the web Real Time Communication technology that allows real time voice, text and video data transfer directly, without the need of an intermediate server, between nodes without having to worry about compatability. WebRTC consists of several interrelated APIs and protocols which work together to achieve this. In order to create a connection between two nodes, one of the connecting nodes creates an "offer". This text document contains information that could be used to facilitate connections. The document can be transmitted by any means but in our application transfer is facilitated through a signalling server. Nodes are connected to the signalling server through websockets, a communication protocol that enables full-duplex connection over a single TCP connection. When the other nodes receive the offer, it uses this offer to generate an answer in a similar format to the offer and transmits this back to the original node through the signalling server. With each node possesing each others information, a connection can then be established between them. Files can then be sent between nodes as data streams.

In order to facilitate transfer of metadata with every chunk transfer through binary buffer a protocol was developed to attach metadata to the head of every binary message passed between nodes and an algorithm to effectively extract the metadata.

4.2 Client application

The front end application is a web application developed with the React framework, an open-source front-end JavaScript library for building user interfaces based on UI components. When the application is loaded, it uses the browsers implementation of webRTC and websockets. When the website is launched, an account and a private key is generated for that user by the blockchain module. A websocket connection is then established between the user and the signalling server. Whenever a user hosts a new file, the files metadata is transmitted to the signalling server. The signalling server then passes this metadata to the remaining nodes. When a file is selected to be downloaded, the client application makes a call to the blockchain module with the list of users currently hosting that file to acquire their current reputation scores. A webRTC connection is then established with the hosting peer via the signalling server. All further interactions between these nodes are carried out using this same connection. The file request is then sent to the hosting peer using this channel and the file is exchanged.

The file is broken down into 16 Kilobyte chunks and transmitted by the hosting channel. For each successful chunk transmitted a success and failure vote is cast respectively for each successful or failed chunk transfer.

4.3 Signalling-Backend server

The signalling server is developed using the NodeJS. NodeJS is a cross-platform backend JavaScript runtime environment running on googles V8 engine. It is used for non-blocking, event-driven servers with real-time pushed based architecture in mind. It uses nodeJS implementation of web sockets to enable connection with the peers. When a new peer is connected to the network it is added to the registry and transmitted to other nodes. The same is done when a new file is hosted by a node. When a user request to make a connection to a node, it sends a peer connection offer to the signalling server as well as a peer ID. The server then uses this ID to route the offer then does the save when it receives the answer.

4.4 Blockchain Server

The backend server is also developed using NodeJS. When launched, its spins up a Ganache blockchain instance with the appropriate configuration. The smart contract is deployed on this local blockchain. This server controls all communication with the blockchain. It registers users on the blockchain by giving them the right to vote and registers these votes.

The smart contract consists of Two functions:

- GiveRightToVote: This function takes a userID and adds it to a map of registered users while initializing that users reputation score to 4.
- Vote: This takes in a score and seeder address to add the appropriate score to seeders reputation. The current score of each user can be generated by calling the peers property of the smart contract with the approriate ID.

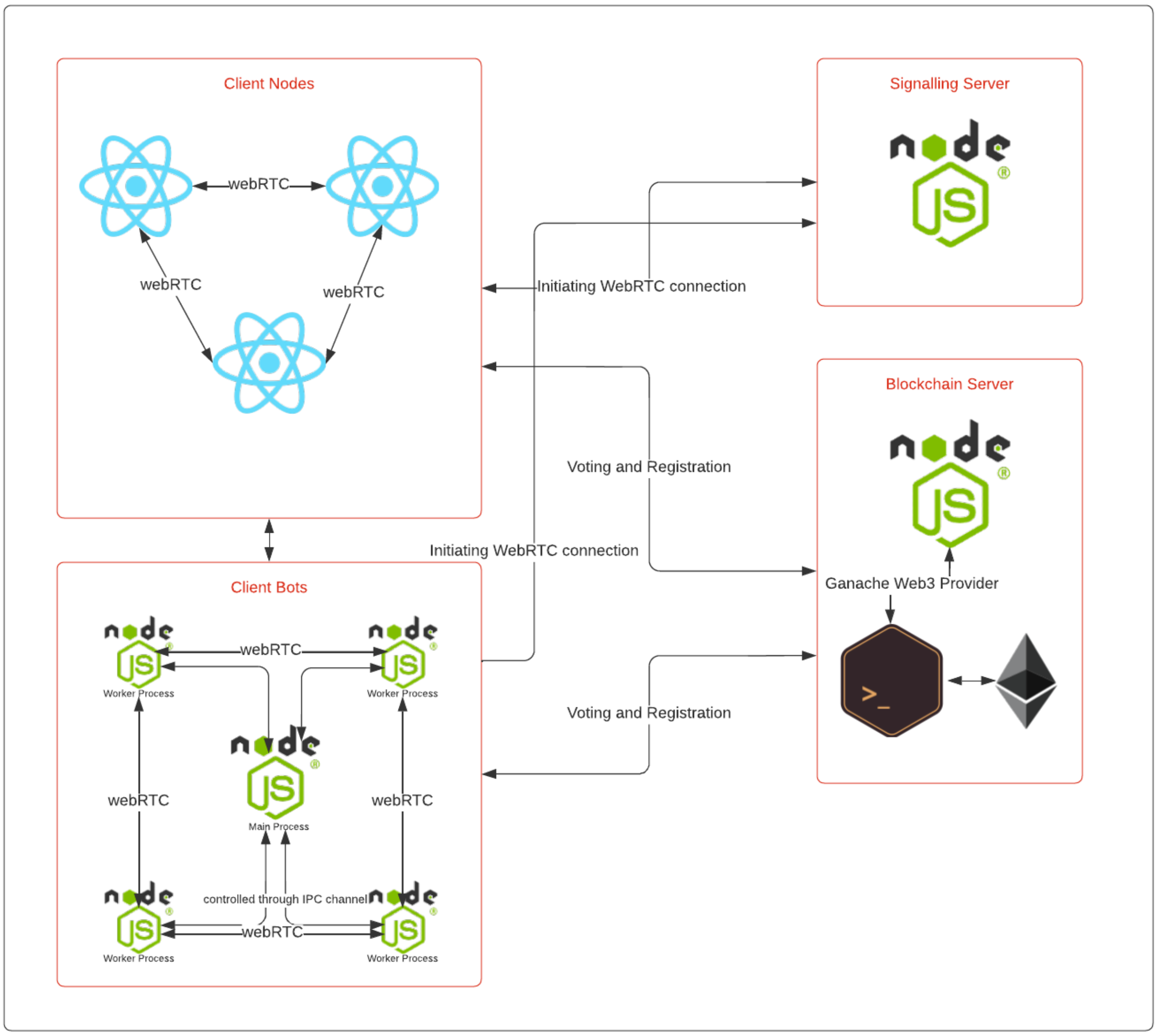


Figure 1: Reputation System Architecture.

4.5 Configuration

In order to fully evaluate the system proposed, results must be gathered from a simulation with flexible perimeters. Therefore, a simulation engine has been developed. This engine has been written in NodeJS and is meant to simulate the activities of active nodes from hosting files, downloading and voting. Its uses a cluster architecture to spin up a set of child processes from the master process and return its results back to the user. The simulations run in cycles with each cycle node in the cycle running with the same account but hosting and downloading different files. The parameters are discussed below:

4.6 Number of Nodes and Failure Rate

Each node is worker process that takes a number of files to download and host. In the initialization phase, each node is registered on the network. An account is generated for the user and the smart contract registers it through the registration function call. Only registered users are allowed to vote and be voted for. During each cycle it downloads the required files and votes based on the success or failure of chunk downloads. Each node also takes in a failure rate to determine how likely it is to fail to send a chunk successfully.

Table 3: Test for effects of Failure Rate

No of Nodes	No of Cycles	No of Files per Node	Size of files	Reputation Score Range	Failure Rate Range	Uses Reputation
10.00	5.00	20.00	medium	8.00	0.20	true
10.00	5.00	20.00	medium	8.00	0.20	false
10.00	5.00	20.00	medium	8.00	0.40	true
10.00	5.00	20.00	medium	8.00	0.40	false
10.00	5.00	20.00	medium	8.00	0.60	true
10.00	5.00	20.00	medium	8.00	0.60	false

4.6.1 Number of Cycles

In each cycle a random selection of downloadable files are selected then for each node a new worker process is created. Each cycle consists of 3 stages.

- Initialization: In this stage, several child processes, depending on the number of nodes specified, are generated using nodeJS clusters. The nodes receive the ID's they shall be identified by in that cycle. The files each node shall host are also registered in this stage.
- Connection: In the stage, the nodes read the files they are required to download and generate a list of nodes they shall have to connect to in that cycle. They then initialize a webRTC connection with each of those peers.
- Start: In this stage the actual transfer of files is initiated and metrics collection begins. As each node finishes download of all the required files, it send a IPC signal to the parent process. When all nodes have sent this signal the cycle ends and all child processes are destroyed.

Table 4: Test for effects of Number of Cycles

No of Nodes	No of Cycles	No of Files per Node	Size of files	Reputation Score Range	Failure Rate Range	Uses Reputation
10.00	1.00	20.00	medium	8.00	0.40	true
10.00	1.00	20.00	medium	8.00	0.40	false
10.00	3.00	20.00	medium	8.00	0.40	true
10.00	3.00	20.00	medium	8.00	0.40	false
10.00	5.00	20.00	medium	8.00	0.40	true
10.00	5.00	20.00	medium	8.00	0.40	false
10.00	10.00	20.00	medium	8.00	0.40	true
10.00	10.00	20.00	medium	8.00	0.40	false

4.6.2 Number of Files per Node and File Sizes

The workload of files for each node to send and receive consists of text files of variable sizes. These files are generated using recipes that express what the file should contain. The simulation runs on three file types:

- Small: These are 10Kb files.
- Medium: These are 524Kb files.
- Large: These are 1024Kb files.

Table 5: Test for effects of Number of files

No of Nodes	No of Cycles	No of Files per Node	Size of files	Reputation Score Range	Failure Rate Range	Uses Reputation
10.00	5.00	10.00	medium	8.00	0.40	true
10.00	5.00	10.00	medium	8.00	0.40	false
10.00	5.00	14.00	medium	8.00	0.40	true
10.00	5.00	14.00	medium	8.00	0.40	false
10.00	5.00	20.00	medium	8.00	0.40	true
10.00	5.00	20.00	medium	8.00	0.40	false

Table 6: Test for effects of Size of files

No of Nodes	No of Cycles	No of Files per Node	Size of files	Reputation Score Range	Failure Rate Range	Uses Reputation
10.00	5.00	20.00	small	8.00	0.40	true
10.00	5.00	20.00	small	8.00	0.40	false
10.00	5.00	20.00	medium	8.00	0.40	true
10.00	5.00	20.00	medium	8.00	0.40	false
10.00	5.00	20.00	large	8.00	0.40	true
10.00	5.00	20.00	large	8.00	0.40	false

4.7 Voting Scores

With each successful or failed transfer, votes are cast to affect the reputation of the seeder. The difference in the negative or positive votes can be varied to evaluate its effect on the system as nodes with lower reputation are less likely to be selected as seeds.

Table 7: Test for effects of Voting Scores Range

No of Nodes	No of Cycles	No of Files per Node	Size of files	Reputation Score Range	Failure Rate Range	Uses Reputation
10.00	5.00	20.00	medium	4.00	0.40	true
10.00	5.00	20.00	medium	4.00	0.40	false
10.00	5.00	20.00	medium	8.00	0.40	true
10.00	5.00	20.00	medium	8.00	0.40	false
10.00	5.00	20.00	medium	12.00	0.40	true
10.00	5.00	20.00	medium	12.00	0.40	false

5 Evaluation

In this study, in order to determine the failure rate and average bandwidth per node, several experiments were carried out using the simulation engine with a variety of parameters. The tables below shows the configuration of the experiments carried out.

5.1 Results

The objective of this paper is to determine whether using a reputation would improve service capacity. To determine this, certain results have to be gathered from the simulation experiments. The most crucial is the average bandwidth of the nodes over the course of the experiment. The next is the failure rate. These are the rates at which the chunk requests fail in a cycle.

5.1.1 Number of Files

As the number of files being transfered changes, it is observed that the effectiveness of the system increases. Even at the lower end of the number of files, it is still observed that the performance in both bandwidth and lower failure rate is significant.

Table 8: Results of No of Files per Node

Uses Reputation	No of Files per Node	Failure Rate	Bandwidth
true	10.00	0.46	245,269.46
false	10.00	0.55	204,090.47
true	14.00	0.40	350,730.97
false	14.00	0.55	232,058.04
true	20.00	0.37	414,178.61
false	20.00	0.55	289,206.50

5.1.2 Number of Cycles

The number of cycles has been fluctuated through the experiment. As expected there is a steady increase in performance as time passes by either in the form of more cycles or more files to be sent as the systems learns which peers are more viable. As the number of cycle increases the upgrade in performance dimishes as the network settles to its optimal performance.

Table 9: Results of No of Cycles

Uses Reputation	No of Cycles	Failure Rate	Bandwidth
true	1.00	0.48	373,318.14
false	1.00	0.57	292,898.89
true	3.00	0.41	387,457.41
false	3.00	0.55	296,678.41
true	5.00	0.37	414,178.61
false	5.00	0.55	289,206.50
true	10.00	0.37	421,789.09
false	10.00	0.54	292,456.10

5.1.3 Failure Rate Range

The higher the difference in the availability of the different shows increasing gain performance in the use of the reputation system compared to without. In real world systems with wide variability in availability this proves the efficacy of the reputation system.

Table 10: Results of Failure Rate Range

Uses Reputation	Failure Rate Range	Failure Rate	Bandwidth
true	0.20	0.37	412,034.32
false	0.20	0.42	384,347.19
true	0.40	0.37	414,178.61
false	0.40	0.55	289,206.50
true	0.60	0.35	2,508,375.00
false	0.60	0.61	26,987.67

5.1.4 Size of file

With smaller file sizes there is the overhead of having to make collections and voting which is obvious in the bandwidth. But with larger files the effect of this overhead diminishes.

Table 11: Results of Size of file

Uses Reputation	Size of files	Failure Rate	Bandwidth
true	10.00	0.49	9,547.34
false	10.00	0.54	9,231.15
true	524.00	0.37	414,178.61
false	524.00	0.55	289,206.50
true	1,024.00	0.37	404,565.67
false	1,024.00	0.55	278,789.09

5.1.5 Voting Score Range

The difference in voting ranges also shows significant changes. As this has no effect on the no reputation system its performance remains the same. But with the reputation system, there is a clear increase in performance between a range of 4 and 8. The difference in failure rate and bandwidth does not change by much between 8 and 12 thus showing the optimal voting range lies between those points.

Table 12: Results of Voting Score Range

Uses Reputation	Voting Score Range	Failure Rate	Bandwidth
true	4.00	0.41	354,077.27
false	4.00	0.55	298,442.01
true	8.00	0.37	414,178.61
false	8.00	0.55	289,206.50
true	12.00	0.37	418,052.58
false	12.00	0.55	291,305.21

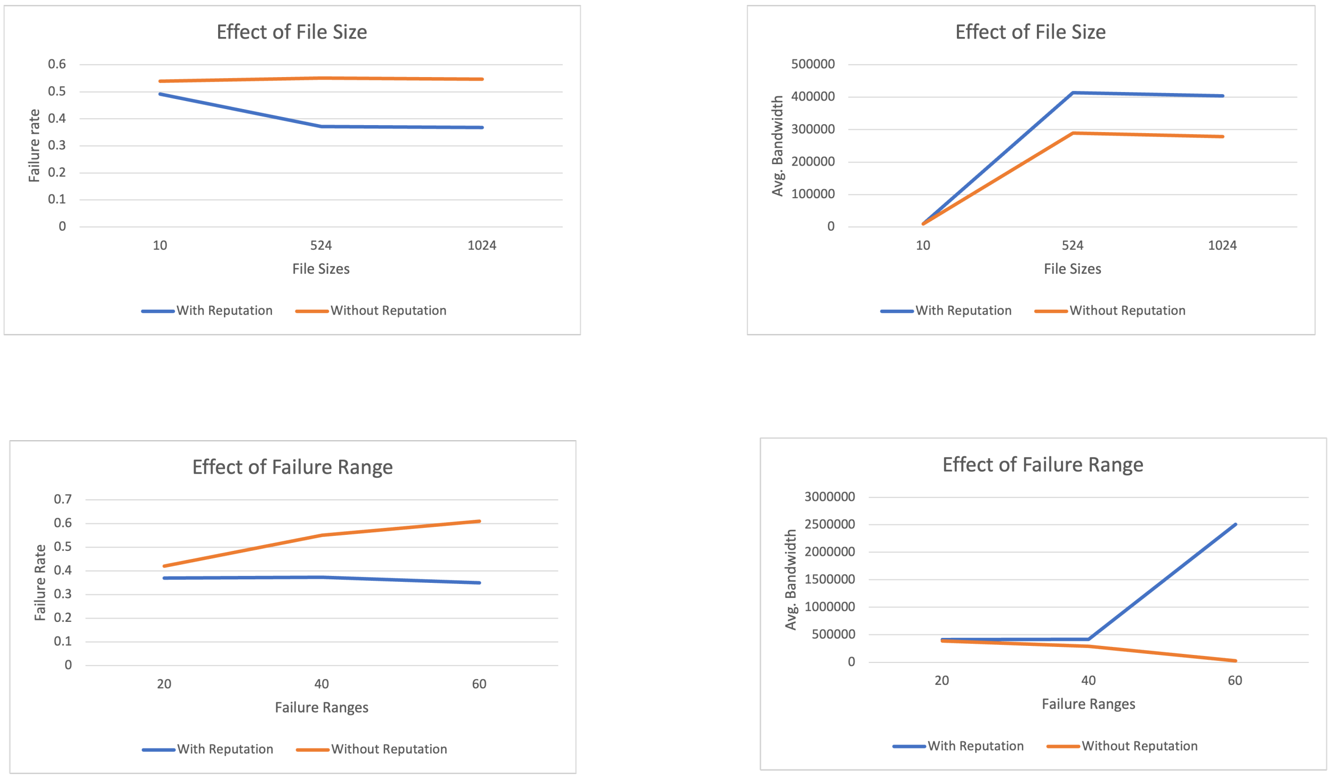
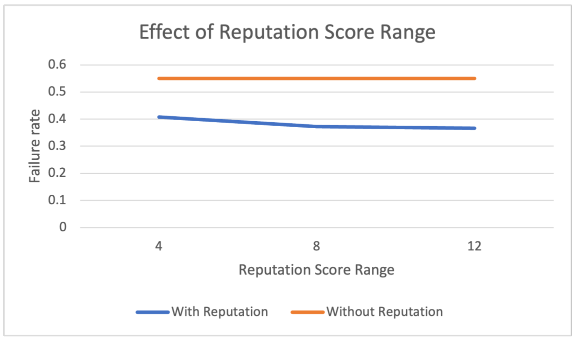
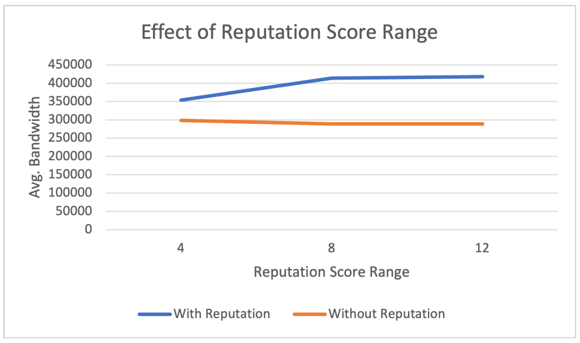
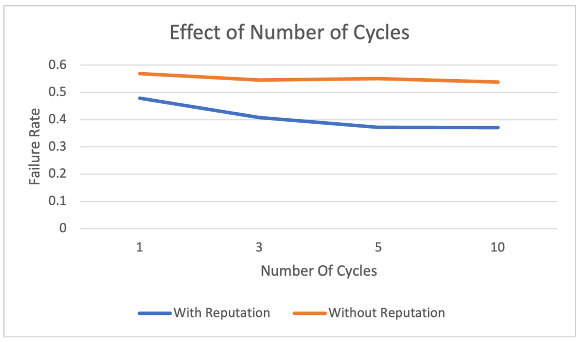


Figure 2: Effects of Node Failure Range and File Size.







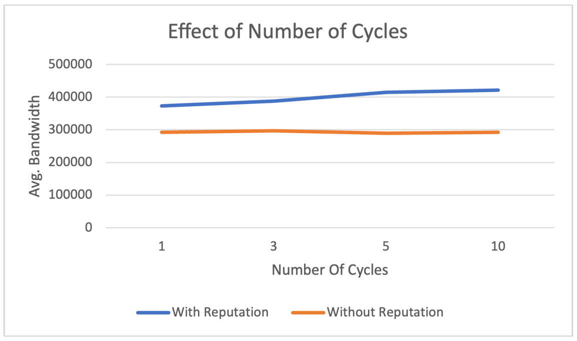
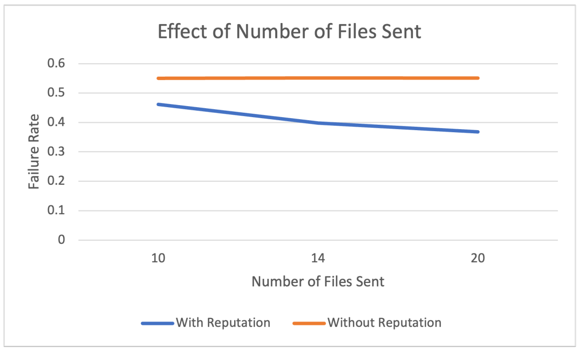


Figure 3: Effects of Reputation Votes Range And Number of Cycles.



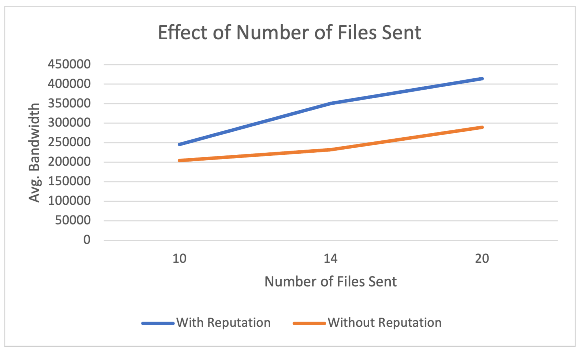


Figure 4: Effects of Number of Files Sent Per Node.

5.2 Summary

From the results there is a clear difference in the performance between using the reputation system and without. In the simulation system the table shows that there is a steady decrease depending on the size of the range failure rates applied to nodes. With a smaller range, there the difference in the average bandwidth between nodes is less pronounce. But with an increase in the range, this difference is far more drastic. A higher difference in the failure rate means the difference in reputation has a higher effect as more requests are directed toward nodes with a higher success rate requiring fewer transactions. Overall in all cases there is increased bandwidth when compared to not using the reputation system. But this effect requires some time either in number of cycles or number of files transferred to show its full potential.

6 Conclusion and future work

This study investigates potential methods to increase the service capacity in peer-to-peer file sharing networks. By using blockchain technology, it is possible to make the data genuinely decentralized, trustworthy, and unchangeable. The results of the studies used to support this solution's feasibility indicate that it consistently outperforms a system that ignores user reputation.

Further research should be carried out on also integrating a payment incentive in order to improve file availabilty. When there is the possibility of monitoring compensation, seeders are more likely to optimize their systems to reduce failure as much as is possible.

Managing peer reputation update fraud is a growing problem. Having peers who have attained a specific level of reputation only be permitted to vote is one potential approach. One more is to restrict votes to those that are comparable to the most recent votes of the seeds. Utilizing an identifying token provided by peers and seeders, maybe in the manner suggested by [32], is a third option.

The amount of transactions that can be completed per unit of time owing to the nature of the proof of work concept is a significant limitation, even if the majority of decentralized blockchain applications are deployed on the Ethereum blockchain. The average transaction rate on the Ethereum network is 15 per second. To overcome this restriction, other potential blockchains like as Solana might be investigated.

Link to project presentation: https://youtu.be/ssqsy9yau8I Link to demo video: https://youtu.be/O9lwc5Xw6GA

References

[1] R. Schollmeier, "A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications," in *Proceedings First International Conference on Peer-to-Peer Computing*, pp. 101–102, 2001. CORE2018 Rank: C.

[2] X. Kang and Y. Wu, "Incentive mechanism design for heterogeneous peer-to-peer networks: A stackelberg game approach," *IEEE Transactions on Mobile Computing*, vol. 14, no. 5, pp. 1018–1030, 2015. JCR Impact Factor 2021: 5.577.

[3] C. Aperjis, M. J. Freedman, and R. Johari, "Peer-assisted content distribution with prices," in *Proceedings of the 2008 ACM CoNEXT Conference*, CoNEXT '08, (New York, NY, USA), Association for Computing Machinery, 2008. CORE2021 Rank: A.

[4] M. Yang and Y. Yang, "An efficient hybrid peer-to-peer system for distributed data sharing," *IEEE Transactions on Computers*, vol. 59, no. 9, pp. 1158–1171, 2010. JCR Impact Factor 2021: 2.663.

[5] X. Yang and G. de Veciana, "Performance of peer-to-peer networks: Service capacity and role of resource sharing policies," *Perform. Eval.*, vol. 63, p. 175–194, mar 2006. JCR Impact Factor 2020: 1.987.

[6] X. Chen and S. A. Jarvis, “Analyzing bittorrent’s seeding strategies,” in *2009 International Conference on Computational Science and Engineering*, vol. 2, pp. 140–149, 2009. CORE2021 Rank:: A.

[7] A. Chow, L. Golubchik, and V. Mishra, “Improving bittorrent: A simple approach,” *Proceedings of the 7th International Conference on Peer-to-peer Systems, IPTPS’08*, pp. 8–8, 02 2008. CORE2018 Rank: C.

[8] A. Legout, N. Liogkas, E. Kohler, and L. Zhang, “Clustering and sharing incentives in bittorrent systems,” *SIGMETRICS Perform. Eval. Rev.*, vol. 35, p. 301–312, jun 2007. JCR Impact Factor 2021: Unavailable.

[9] M. Feldman, K. Lai, I. Stoica, and J. Chuang, “Robust incentive techniques for peer-to-peer networks,” in *Proceedings of the 5th ACM Conference on Electronic Commerce*, EC ’04, (New York, NY, USA), p. 102–111, Association for Computing Machinery, 2004. CORE2021 Rank: A*.

[10] S. Marti and H. Garcia-Molina, “Limited reputation sharing in p2p systems,” in *Proceedings of the 5th ACM Conference on Electronic Commerce*, EC ’04, (New York, NY, USA), p. 91–101, Association for Computing Machinery, 2004. CORE2021 Rank: A*.

[11] R. Dennis and G. Owen, “Rep on the block: A next generation reputation system based on the blockchain,” in *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, pp. 131–138, 2015.

[12] R. Zhou and K. Hwang, “Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 4, pp. 460–473, 2007. JCR Impact Factor 2021: 2.687.

[13] M. Debe, K. Salah, M. H. Rehman, and D. Svetinovic, “Towards a blockchain-based decentralized reputation system for public fog nodes,” in *2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1–6, 2019. CORE2021 Rank: C.

[14] Z. Ma and D. Qiu, “A novel optimistic unchoking algorithm for bittorrent,” in *2009 6th IEEE Consumer Communications and Networking Conference*, pp. 1–4, 2009. CORE20201 Rank: B.

[15] V. Atlidakis, M. Roussopoulos, and A. Delis, “Enhancedbit: Unleashing the potential of the unchoking policy in the bittorrent protocol,” *Journal of Parallel and Distributed Computing*, vol. 74, no. 1, pp. 1959–1970, 2014. JCR Impact Factor 2021: 3.734.

[16] T. Locher, S. Schmid, and R. Wattenhofer, “Rescuing tit-for-tat with source coding,” in *Seventh IEEE International Conference on Peer-to-Peer Computing (P2P 2007)*, pp. 3–10, 2007. CORE2018 Rank: C.

[17] R. Izhak-Ratzin, N. Liogkas, and R. Majumdar, “Team incentives in bittorrent systems,” in *2009 Proceedings of 18th International Conference on Computer Communications and Networks*, pp. 1–8, 2009. CORE2021 Rank: B.

[18] R. Zhou, K. Hwang, and M. Cai, "Gossiptrust for fast reputation aggregation in peer-to-peer networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 9, pp. 1282–1295, 2008. JCR Impact Factor 2021: 6.977.

[19] M. Gupta, P. Judge, and M. Ammar, "A reputation system for peer-to-peer networks," in *Proceedings of the 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, NOSSDAV '03, (New York, NY, USA), p. 144–152, Association for Computing Machinery, 2003. CORE2021 Rank: Unranked.

[20] A. Goswami, R. Gupta, and G. S. Parashari, "Reputation-based resource allocation in p2p systems: A game theoretic perspective," *IEEE Communications Letters*, vol. 21, no. 6, pp. 1273–1276, 2017. JCR Impact Factor 2020: 3.436.

[21] Z. ImaniMehr and M. DehghanTakhtFooladi, "Token-based incentive mechanism for peer-to-peer video streaming networks," *J. Supercomput.*, vol. 75, p. 6612–6631, oct 2019. JCR Impact Factor 2020: 2.474.

[22] S. Jun and M. Ahamad, "Incentives in bittorrent induce free riding," in *Proceedings of the 2005 ACM SIGCOMM Workshop on Economics of Peer-to-Peer Systems*, P2PECON '05, (New York, NY, USA), p. 116–121, Association for Computing Machinery, 2005. CORE2018 Rank: C.

[23] D. Qiu and R. Srikant, "Modeling and performance analysis of bittorrent-like peer-to-peer networks," *SIGCOMM Comput. Commun. Rev.*, vol. 34, p. 367–378, aug 2004. JCR Impact Factor 2021: 2.712.

[24] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica, "Free-riding and whitewashing in peer-to-peer systems," in *Proceedings of the ACM SIGCOMM Workshop on Practice and Theory of Incentives in Networked Systems*, PINS '04, (New York, NY, USA), p. 228–236, Association for Computing Machinery, 2004.

[25] A. L. Jia, L. D'Acunto, M. Meulpolder, and J. A. Pouwelse, "Modeling and analysis of sharing ratio enforcement in private bittorrent communities," in *2011 IEEE International Conference on Communications (ICC)*, pp. 1–5, 2011. CORE2018 Rank: B.

[26] M. Debe, H. R. Hasan, K. Salah, I. Yaqoob, and R. Jayaraman, "Blockchain-based energy trading in electric vehicles using an auctioning and reputation scheme," *IEEE Access*, vol. 9, pp. 165542–165556, 2021. JCR Impact Factor 2020: 3.367.

[27] Z. Geng, Y. He, C. Wang, G. Xu, K. Xiao, and S. Yu, "A blockchain based privacy-preserving reputation scheme for cloud service," in *ICC 2021 - IEEE International Conference on Communications*, pp. 1–6, 2021. CORE2018 Rank: B.

[28] D. Figueiredo, J. Shapiro, and D. Towsley, "Incentives to promote availability in peer-to-peer anonymity systems," in *13TH IEEE International Conference on Network Protocols (ICNP'05)*, pp. 12 pp.–121, 2005. CORE2021 Rank: A.

[29] H. Jiang, S. Guo, D. Zeng, and H. Jin, "Improving content availability by request-adaptive incentive in private peer-to-peer communities," in *2012 IEEE Global Communications Conference (GLOBECOM)*, pp. 2708–2713, 2012. CORE2021 Rank: B.

[30] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[31] V. Buterin, "A next generation smart contract & decentralized application platform (2013) whitepaper," *Ethereum Foundation*, 2013.

[32] N. Fotiou, I. Pittaras, V. A. Siris, S. Voulgaris, and G. C. Polyzos, "Oauth 2.0 authorization using blockchain-based tokens," 2020.