JustTorrent: Value Based - Fairer and Faster Protocols for P2P File Sharing

Ozan Okumuşoğlu*, Muhammed Furkan Bayraktar*, Alptekin Küpçü*‡

* Department of Computer Engineering, College of Engineering, Koç University, İstanbul, Turkey

(ookumusoglu@ku.edu.tr, mubayraktar@ku.edu.tr, akupcu@ku.edu.tr)

‡ Corresponding Author; Tel: +90 212 338 1363, Fax: +90 212 338 1548, akupcu@ku.edu.tr

Received: 30.01.2017 Accepted: 14.03.2017

Abstract- BitTorrent is a highly effective peer-to-peer file sharing protocol. It allows clients to share large files over the network by dividing a file into pieces and dividing pieces into blocks. Studies so far showed that BitTorrent's tit-for-tat strategy does not provide fairness. Thus, there were some proposed modifications and extensions. Unfortunately, the previous attempts at providing fairness fail when pieces are not of equal value. In this paper, we consider value-based fairness in BitTorrent, for the first time, as we show that different pieces in a single torrent may be of different value. We suggest two separate modifications to increase fairness and decrease average download time in the BitTorrent protocol. The results of our modifications show us that via fair mechanisms, one may provide security against adversaries who only request valuable pieces in the system. Moreover, one may achieve around 30% better average download times for peers in the system, while remaining fair.

Keywords BitTorrent, fairness, fair exchange, peer-to-peer.

1. Introduction

BitTorrent is a popular peer-to-peer file sharing protocol. It enables users to share a file by dividing it into pieces. Indeed, torrents may contain multiple files, and the whole torrent is split into many pieces. The advantage of the system is that there is no single server; hence no single point of failure exists. Clients connect to each other to download pieces while they are also uploading the pieces that they have. The drawback of such a system is that nodes may choose not to upload and BitTorrent's tit-for-tat mechanism is not enough to avoid this free-riding [1-3].

There were attempts to prevent this free-riding behavior. Some were based on game theory [4], some used reputation [5-9], and some others employed cryptographic protocols [10, 11]. The buy and barter cryptographic protocols [10, 11], when used over BitTorrent, are guaranteed to prevent free-riding, but only in the case where all pieces are of equal value.

We realize that each piece in a torrent does not necessarily have the same importance in real life. Consider, for example, a popular song in an album. While downloading the torrent that includes the whole album, the pieces that correspond to the popular song are more important. As another example, consider a Bitcoin blockchain [12]. The most important block there is actually the last block, since mining would continue on top of that. Hence, if BitTorrent were used to distribute the Bitcoin blockchain among the peers, adversarial peers would target the latest blocks, rather than keeping a copy of the whole blockchain. Indeed, some Bitcoin clients already use a related idea of checkpointing (see *https://en.bitcoin.it/wiki/Checkpoint_Lockin*). Thus, we modify existing buy and barter protocols [10, 11] to work with pieces of different values. We show that using the value-based barter protocol, the system can be made fair even against an adversary who requests only the valuable pieces in the system. Using the value-based buy protocol, we show that **the average download time is decreased around 30%** compared to the original BitTorrent protocol.

We performed simulation tests comparing regular BitTorrent, existing (count-based) buy and barter protocols, where exchanging one piece with any other is considered fair, and our value-based buy and barter protocols. Our tests prove that using the value-based exchange protocols perform better than both the regular BitTorrent, and the original buy and barter protocols.

In our modifications to BitTorrent, we aim to achieve fairness, even against an adversary who requests only the valuable pieces in the system. Another goal is speed, meaning decreasing the average download time for the peers. For achieving a fairer BitTorrent, we use the value-based barter protocol. For achieving a faster BitTorrent, we use the value-based buy protocol. Our methods are based on controlling the upload/download ratio for each node. The previous works calculate this ratio mostly based on the number of pieces uploaded or downloaded, whereas we calculate it based on the total value of the pieces uploaded or downloaded. In our simulations, using the barter protocol corresponds to not requesting new pieces until our ratio goes above 1, and using the buy protocol corresponds to not leaving the system until the ratio is 1.

Our **contributions** in this paper can be outlined as follows:

- ➢ We consider the different values of different pieces in BitTorrent, for the first time.
- ➢ We modify existing cryptographic fair exchange protocols to work properly in terms of exchanging equal value, instead of equal number of blocks.
- ➢ Under this value-based setting, we create a **fairer BitTorrent** and show that it provides fairness even against an adversary who requests only the valuable pieces in the system, and achieves a fairer distribution of the value among the peers in the system.
- ➢ We also achieve a **faster BitTorrent**, again using value-based fair exchange ideas, and our simulations show around 30% better average download times, compared to the original BitTorrent.

The paper is structured as follows: Section 2 provides background information about the current BitTorrent protocol, buy, and barter methods. In Section 3, we discuss the relationship between previous work and our research. In Section 4, we first motivate the value-based setting and then explain the proposed modifications done on the system for our fairer and faster solutions. Section 5 provides information about our simulation environment and discussions about our results. We conclude our work in Section 6 by also providing future directions.

2. Background

BitTorrent is designed for efficient peer-to-peer file sharing [13] (and *http://bittorrent.com*). Files are divided into equal-sized *pieces* (256 KB) and pieces are divided into blocks. In order for the system to work, all the pieces of a file must be provided in to the system by at least one node. A node can upload a piece that it previously downloaded while downloading other pieces.

There are also *trackers* in the system, which keep track of all the peers in the system. When a new node wants to join the system, it first connects to the tracker and gets a subset of peers who have (parts of) the file that he wants to download. Then, the node tries to establish a connection with each node in this subset. If the connection is successful, the subset is called as the *neighbors* of the node. Eventually a group of neighbors is called as the *peerset* of a peer. If a peer has the whole file and it is still in the system, then it is called a *seeder*. If a peer is still downloading the file, it is called a *leecher*. Note that seeder and leecher are defined per torrent (e.g., a peer can be a seeder of one torrent, while being a leecher of the other).

A *rarest first* policy is applied during the requests of pieces. At the beginning of the download, the peers start to download random pieces, but after a while they start requesting the rarest pieces among their peerset. When a peer wants to upload pieces to other peers, it sends an *unchoke* message to them. While it is downloading pieces from other peers, it sends a *choke* message. The choked neighbors will be responsible for sending the pieces and unchoked ones will download pieces from the nodes that they are unchoked by.

Although there are no limits for uploading and downloading of the file, the nodes may choose not to upload and this is called free-riding [1-3]. Since the file is not distributed by such free-riders, this has some drawbacks. BitTorrent tries to block this by a rate-based tit-for-tat policy. Every ten seconds, peers check the upload and download rates of their choked and unchoked neighbors in their peerset and detect the unchoked peer who provides the lowest download rate and the choked peer who provides the highest download rate. If unchoked peer's download rate is less that the choked one, then a choke message will be sent to the unchoked peer and an unchoke message is sent to the choked peer. A fair distribution of the file is tried to be achieved with this system. Yet, previous works show that this is *not* sufficient [1-3, 10, 11].

A cryptographic fair exchange protocol may be applied on top of BitTorrent to provably prevent free-riding [10, 11] if all the pieces of the file are of equal value. Unfortunately, we argue that this is not always the case, and hence we would require better fairness solutions to solve the issue. Since we build upon the buy and barter methods of [10, 11], we briefly explain how they can be used to exchange the pieces in BitTorrent:

Barter Method: The method is introduced as the fair exchange of items by [10]. Fair exchange commonly involves two sides, Alice and Bob. They both want something from each other, and the barter protocol ensures that either each of them gets what (s)he wants, or neither of them does. The protocol in [10] is very efficient in repeated scenarios such as BitTorrent block exchanges, and at a high level, involves first exchanging encrypted blocks, and then exchanging decryption keys. The exchanges can be one block by one block, or batches of blocks, as long as both sides send the same number of blocks.

Buy Method: Belenkiy et al. [11] gave a protocol for buying digital content in exchange for electronic payments. Suppose Alice wants to buy a file from Bob, and at the end either Alice gets the file giving an electronic coin to Bob, or both get nothing. They achieve this via use of virtual currencies named *e-coins* (endorsed e-cash) [14], and verifiable escrows [15].

3. Related Work

3.1. Fairness Definitions

Yue et al. [16] and Fan et al. [17] explain and analyze the relation between performance and fairness in BitTorrent protocol. Levin et al. [4] also provide proofs for the lack fairness in the current BitTorrent protocol using a game-theoretical approach. They show that there are no incentives to upload using their maximum bandwidth. They also showed that system is vulnerable to Sybil attacks [18].

Sherman et al. [19] focused on problems of unfairness in BitTorrent protocol. Their system is close to our model and they achieve an increase in upload/download rates by using a deficit mechanism for choking and unchoking. Our paper also focuses on upload/download rates, but our tests were made on a larger number of nodes to represent real life conditions well. Furthermore, they do not consider fairness as we do.

Rahman et al. [20] tries to provide efficiency by focusing on the bandwidths of the nodes. On the other hand, the ones who have lower bandwidths but willing to upload more does not have a chance to get better download rates in their system since they will always get lower proportions. Our system does not make changes with respect to bandwidth, but it gives an opportunity for faster download rates to the ones who are willing to upload in to the system. This also provides an opportunity to us to distribute the file over all nodes without any discrimination.

3.2. Fair Exchange

Asokan et al. [21] first introduced the concept of optimistic fair exchange. They introduced two optimistic fair exchange protocols for exchanging electronic goods like signatures or data. In the optimistic setting, the trusted third party, which is required for fair exchange [22], is involved only if there is a problem.

[10, 11] proposed the use of electronic coins to bring fairness to BitTorrent from a different aspect. Their papers explain the *buy* and *barter* protocols to exchange pieces and e-coins, without losing privacy. Their system provides fairness by making P2P accountable, also relying on a trusted third party.

Thommes and Coates [23] also provide modifications to the BitTorrent protocol by using conditional optimistic unchoke model in order to increase the fairness. Their definition of fairness also represents the upload/download ratio. With their improvement in the protocol, they showed that fair exchange can actually increase the performance of the current BitTorrent protocol.

The papers above try to find a solution for fairness problem in BitTorrent protocol, but their modifications did not consider the value-based setting that is more realistic, and proper measurements were not taken in [10, 11]. In this paper, we offer a mechanism that can be used together with the current BitTorrent protocol. Our system prevents nodes from exploiting the system and every node has the same opportunity to use the advantages of the system without considering their bandwidths.

4. Proposed Modifications over BitTorrent

We created two different modified protocols, which use fairness in their core. In this paper, fairness definition is to make upload/download ratio close to 1. Both of our protocols force the fairness of the system by making upload/download rate closer to 1. Each protocol comes with its own advantage and they are relatively simple to implement. One of them is called the *Fairer Torrent Protocol* and it uses the *value-based barter* method to provide fairness against a value-seeking adversary. The other uses the *value-based buy* method to provide better average download times for the system, and is called the *Faster Torrent Protocol*. We tested our protocols for the fairness they have provided, as well. At the end, we observe that they have certain advantages over each other, hence the choice between those protocols comes to a trade-o between security and better average download time. Another advantage of our protocols is that they can work together with the current BitTorrent protocol; so in a real life implementation, users will not need to search for a certain protocol. Below, we first explain the valuation of the pieces and the adversarial behavior we prevent. Then buy and barter methods in our scheme will be explained. Finally, our protocol changes will be introduced.

4.1. Valuation of the Pieces

As we briefly discussed, while BitTorrent considers all the pieces having the same value, in real life some torrent pieces are more important than others, such as the popular song and Bitcoin examples in the introduction section. Yet another example can be given from the finance sector. Companies publish their consolidated financial reports yearly with lots of details over a hundred pages, but auditors mostly focus on the income statement page of this report. In such cases, the value of that page cannot be considered to be the same as the others. Our system allows users to put different values to such pieces, and introduces a fairer environment in the sense of the piece values. As far as we know, this value-based setting has not been considered in the BitTorrent protocol. In our analyses, we achieved better results with value-based protocols.

In such a value-based real world, it is easy to imagine an adversary who wishes to download only the valuable pieces (e.g., the popular song in an album or the balance sheet page of the reports), hence not fully contributing to the system. Preventing such an attack means distributing the valuable pieces fairly among different peers. Thus, we want that when such an adversary, who targets only the valuable pieces, finish downloading those, many other peers also hold those valuable pieces, by just adhering to our protocol. This means, such an adversarial behavior is not useful, and all peers should stick to the protocol definition. It further implies a fairer distribution of pieces among the peers.

The adversary only requests the valuable pieces as it is explained above. Other than that, the adversary behaves completely like a normal node and there is no way for a node to know that it is actually connecting to an adversary. The purpose of the adversary is to download only the valuable pieces, and upload as few as possible. Our Fairer Torrent Protocol prevents such adversaries from downloading only valuable pieces; instead, forces them to upload the valuable pieces.

In terms of deciding the piece values in a torrent, we may allow the file owner to set the values of pieces before uploading the file in to system. The purpose of the valuation is to increase the distribution of file in to the system and obtain better average download times. If the file owner wants to distribute his files faster, then value setting must be done rationally. If the file owner chooses to assign the same value to each piece, then pieces will be indifferent from each other and downloading or uploading any piece will not make any difference. If the owner chooses to give lower values to actual valuable pieces, then the value-seeking adversary can easily decrease the average actual value of the system. If the owner assigns higher values to all pieces, the pieces will be indifferent again. The main point in the valuation is giving values to pieces with respect to each other, since values only make sense within one torrent; not across torrents. In our tests, we considered a rational file owner who assigns values to pieces in proportion to their actual values.

Another way of setting values can be thought by using certain formats for certain files. For instance, electronic healthcare records (EHR) consist of many different parts. According to [24], the EHR of a patient travels along with the patient throughout his/her life. It contains laboratory results, radiology images, personal statistics, and more. The record travels from one clinic to another via Internet or internal network. The point is that all the information contained in the report is not necessary for the doctor in the first place. Therefore, our system can be used in such cases to give values to certain information contained in the report to make it more efficient and fast. For example, the data can be valued according to date, in order to make the newest ones more valuable than the old ones. Such formats can be extended more, but it is always for the advantage of file owner to assign rational values to pieces in order to distribute the file better in to the system and protect the system against adversaries who only request the valuable pieces.

Furthermore, a feedback mechanism can be set up as a future work for looking up the rare pieces in the system. As explained by Levin et al. [15], the fairness of the BitTorrent is suspicious. The rarity of pieces is also an important clue for the values of the pieces. For example, peers can send feedback to trackers in order to show the pieces which are requested more than others. By looking up that feedback, the tracker can collect this data and send new value distributions to the clients requesting that file. By this rarity-based scenario, values for the pieces will always reflect the real values with respect to their rarity in the system and the whole file will be distributed more to the entire system. This provides dynamism to the system, since actual value changes in a file can be iterated by the tracker.

Another valuation strategy can be the one where the file is important when it is sequential. An example for such a system is audio/video streaming. During a real-time conference video sharing, the newcomers mostly start from the current position and continue. The ones who start from the beginning will be very few. As a result, the next (upcoming) frames will always be demanded more, and we should assign higher values to those pieces than the earlier ones. We can set values for pieces dynamically in order to be used in our protocol. We can also make a more dynamic system where the values are set with respect to each leecher. If a user starts from the previous parts of the video, the values for that leecher will be lower than the values for another leecher who starts from the current time.

Similarly, consider a video streaming website. In general, most videos are not played until the very end. One main reason is that most movie files contain uninteresting credits at the last frames. Thus, most users stop their download at the point. In such a setting, the last frames can be assigned lower values than the first ones. This assignment can be done globally, such that all users are aware. This global assignment can be employed in the popular songs in a music album setting above as well.

For evaluation, we choose to model the importance of the files in two ways. In one model, all pieces are counted as one, but only some of those are chosen as the important ones. In such a system, we calculate the upload/download ratio by looking at the piece count, and the adversary only requests the pieces which are chosen as valuable. The other way is giving values to pieces between 1 and 10, and the adversary is interested in those with high value only. In our tests, for a certain file, the values of pieces are set randomly. In this case, our upload/download ratio is based on the total values of the pieces uploaded and downloaded.

4.2. Value based Buy and Barter Methods

The buy and barter methods are based on fair exchange between peers. In our paper, we modified the existing protocols [10, 11] to make them usable in our value-based setting. In our value-based protocols, when blocks are exchanged, they do not have to be one block by one block, or even equal number of blocks. What we enforce is that peers exchange equal values. For example, in order to protect the upload/download ratio, when a node downloads a piece with value 7, it must upload pieces where their values add up to 7. This can be done, for example, by uploading one piece with value 4, one piece with value 2, and one piece with value 1. In the buy setting, a peer may choose to upload just one piece with value of 10, which will increase the upload/download ratio greatly using less bandwidth, and thus enable that peer to download more in the next exchange.

4.3. Fairer Torrent Protocol

This protocol is deigned to make the system fairer and secure against the adversary explained above. In order to fulfil the requirements in the system, we used the barter method between nodes [11]. According to the barter method, each node is required to have an upload/download ratio over 1 before requesting a new piece. Hence before requesting a new piece, we first check if the node is allowed to download a new piece. If it is allowed, the algorithm continues as it is in the current BitTorrent protocol. If the ratio is below 1, then an unchoke message is sent to all nodes in the peerset in order to upload more pieces in to the system and increase the ratio.

The check for the next piece in this protocol is very simple. We set the ratio as uploaded/downloaded pieces and then check if it is over 1 or not. The important point in this algorithm is that the determination of the pieces uploaded and downloaded. As described in the previous subsection, when values of the pieces are simply 1, we will count each piece as 1 and calculate the ratio. So, in such a scenario, when a node downloads one piece, it must also upload one piece in order to protect its ratio. This corresponds to the scenario in the previous work [2]. When the pieces are given values (e.g., between 1 and 10), then the upload count will be the total value of the uploaded pieces. It will show the total uploaded values in to the system by the current node, and similarly we count the total value for the downloaded pieces. In such a system, the total value of the network will be protected, and sharing valuable pieces will be more important for each peer. Remember that our ratio is computed using different values per piece, whereas previous works value all pieces equally, and hence compute the ratio only by the count (number of pieces).

Algorithm 1 is used for requesting new blocks for further processing. If the algorithm returns false, then the node is not able to download further pieces and sends unchoke messages to its peerset. The unchoke messages will allow the node to upload some of its pieces to others and this will increase its upload/download ratio. If some pieces were requested while the ratio is above 1, but now the ratio is below 1, then those pieces are always accepted from the neighbors in order not to waste the already-used network resources.

Algorithm 2 allows us to keep the adversary under control because no node can request all the valuable pieces without uploading anything. Instead, such adversaries who try to download only the valuable pieces will be forced to become beneficial to the system since they replicate valuable pieces in the system by uploading them more. Moreover, the nodes become always willing to upload to the others. In Section 5, we show that the value-based solution prevents the adversary, as suggested.

There is a problem regarding bootstrapping new users in this scenario. This is done by seeders in our system, and other alternatives (including the solution in the next section without valuations) were proposed in the literature [11].

Algorithm 1: Upload/Download ratio check for a node.

```
function isAllowedToDownload
    ...
    ratio ← piecesUp / piecesDown
    ...
    if ratio >= 1
        return true
    else
        return false
    end if
end function
```

Algorithm 2: Requesting new blocks, if the node is allowed to download.

```
function requestNextBlocks
    ...
    if not isAllowedToDownload
        foreach Node p in neighborNodes
            send UNCHOKE message to p
        end foreach
        return
    end if
    ...
end function
```

4.4. *Faster Torrent Protocol*

Although we provided security with our Fairer Torrent Protocol, we observed in our simulations that the average download times are increasing for the nodes in the system. The reason for this is that the nodes are always searching for someone to upload in order to continue their download. To solve this problem, we introduce the Faster Torrent Protocol, which uses the value-based buy method instead of barter to introduce fairness and speed into the system.

The advantage of the buy method is that during the download process, the protocol works as it is in the original BitTorrent protocol. The only thing that changed is that nodes cannot leave the system until their upload/download ratio is greater than or equal to 1. This is because if they leave earlier, they will not have enough e-coins (some virtual currency to ensure the ratio: it does not need to have monetary value) to come back to the system for downloading the next file. Nodes can use the file when they have all the pieces, so they do not need to wait to be able to use the file, but they need to continue uploading until they reach the upload/download ratio of 1.

As it is done in our Fairer Torrent Protocol, the upload/download ratio will be based on the piece count in one scenario, and will be based on the values in the other. When the ratio is based on the piece count, nodes can upload any piece since the ratio will be affected at the same rate. On the other hand, when the ratio is based on the values of the pieces, the nodes can reach a higher upload/download ratio by uploading the valuable pieces more. This system prevents the valuable pieces becoming a bottleneck for a file.

In this protocol, each node needs to have the upload/download ratio greater than or equal to 1 when they leave the system and nodes cannot exploit the system since they must contribute as much as they download. If a node tries to leave the system before reaching that ratio, it will not have enough credits to download the other files in the system.

This protocol makes each node behave like a seeder in the system and the load on the original seeders are decreased since each node is looking for some other node to upload pieces. In a dynamic system, newly arrived nodes can easily find some node to download the pieces from; hence their average download time is decreased. Another advantage of this system is that the file is distributed across the network much faster since all nodes are willing to upload. This result causes a decrease in the average download times for the nodes. The more dynamic the system becomes, the more nodes will take advantage of the system. Our experiments in Section 5 confirm that our value-based solution enables around 30% faster downloads.

5. Experimental Analysis and Results

5.1. Setup

For our tests we used the BitTorrent simulator implemented on PeerSim (*http:// peersim.sourceforge.net*). Since we did not make any changes on the bandwidth distribution, the simulator provided us a better environment to test our protocols. However, we had to fix the simulator to deal with a large number of nodes.

Furthermore, to better represent real torrent systems, we conducted the experiments with a dynamic environment, where some nodes leave and some new nodes join the system. To make the system dynamic, we improved the network dynamics module provided by the PeerSim as follows: For the Fairer Torrent Protocol, we let new nodes enter and exit the system, but we focused our results on the control group, which is the group of nodes from the beginning of the experiment. For the Faster Torrent Protocol, we used all the nodes that participated during the experiment for the analysis. To represent a more realistic setting where nodes join the system because they want to download the whole file, we made sure that the nodes do not exit the system before they complete downloading the file (except the adversary).

Even when a node completes downloading the files in the torrent, for the regular BitTorrent, they stay in the system with probability 10%, and for our Faster Torrent Protocol, they stay in the system until their upload/download ratio is 1 (since the buy protocol essentially enforces this behavior). With all these contributions to the PeerSim, we managed to complete our experiments.

We performed our tests separately for each protocol. Each protocol is compared against the original BitTorrent protocol, and tests are performed with similar configurations for each protocol (see below).

5.2. Faster Torrent Protocol Analyses

For the Fairer Torrent Protocol, our purpose is to test whether we can protect our system against the abovementioned adversary or not. The simulation configuration and the behavior of the adversary are the same for the original BitTorrent protocol and our protocol. We start our simulation with 200 nodes, and the system can grow up to 500 nodes. The initial 200 nodes will be our control group, and the system continues to run until the adversary gets all the valuable pieces it wants; at this point we stop the simulation. There is only one adversary in our system, which can connect to all nodes in the system. As described in Section 4.1, the adversary only requests valuable pieces, which corresponds to a random subset of 30% of all the pieces in the torrent. The file size is set as 150 MB.

We also tested our protocols with many other configurations. We observed that the initial number of nodes and the maximum number of nodes did not affect our results, just as the file size also did not reverse our findings. Therefore, we do not unnecessarily present their results below. When we increase the file size, we observed that the distribution of the valuable pieces into the system actually increases. Thus, in some sense, one may consider the analyses below as some worst-case.

Original BitTorrent: The adversary is highly successful in the original BitTorrent protocol. As it is seen on Figure 1, when the adversary collected all the valuable pieces (178 pieces) in the system, the closest node could only get around 60% of them. More than half of the nodes could not download even 30% of the valuable pieces.

Barter Method Based on Piece Count: In this protocol, the upload/download ratio is based on the number of pieces. Nodes can download as long as the number of their uploaded pieces are greater than or equal to the number of the pieces downloaded.

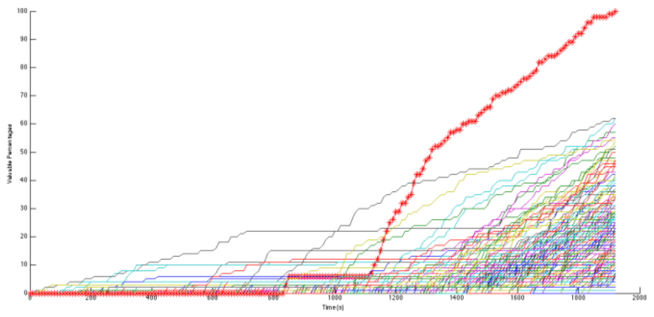

Fig. 1. Adversary in the original BitTorrent Protocol (marked *). X-axis shows the simulation time and Y-axis shows the percentage of the valuable pieces downloaded.

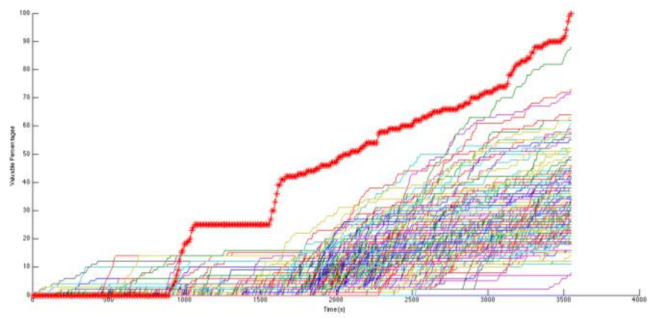

Fig. 2. Adversary in the Count-Based Fairer Torrent Protocol

(marked *). X-axis shows the simulation time and Y-axis shows the percentage of the valuable pieces downloaded.

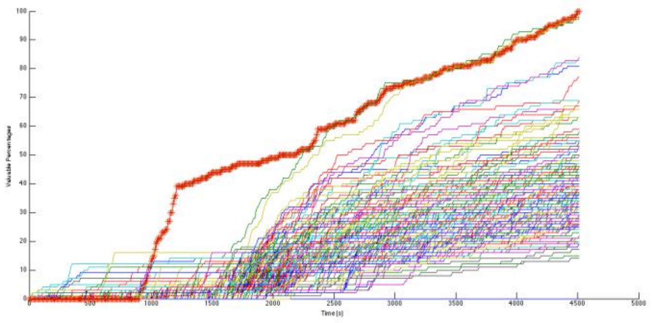

Fig. 3. Adversary in the Value-Based Fairer Torrent Protocol (marked *). X-axis shows the simulation time and Y-axis shows the percentage of the valuable pieces downloaded.

As it is seen on Figure 2, the adversary still got the valuable pieces before any other node, but the gap between the adversary and the other nodes highly decreased. Around 9 nodes got over 60% of the valuable pieces, 3 of whom got over 70% of the valuable pieces. In this scenario, the adversary cannot collect all valuable pieces without distributing them into the system, due to the barter protocol. When it collects all the pieces, its neighbors get those pieces too and valuable pieces are distributed into the system much more than the original BitTorrent protocol.

Barter Method Based on Values of Pieces: In this protocol, nodes are not allowed to download new pieces if the total value of the pieces downloaded is greater than or equal to the total value of the pieces uploaded. In order to make this test to similar to the previous ones, we assign uniformly random values between 1 and 10 to each piece, and design the adversary such that it only requests pieces which have a value greater than 7. In a uniform random distribution, this will be equal to 30% of the pieces.

In Figure 3, the adversary performs its attack on our Fairer Torrent Protocol, which uses the value-based barter method. In this system, the other nodes collected valuable pieces as much as the adversary did. As it is shown in the figure, two nodes collected all valuable pieces even before the adversary. When we compare Figure 3 with others, value-based bartering provides nodes an opportunity to collect the valuable pieces. Since the adversary is forced to

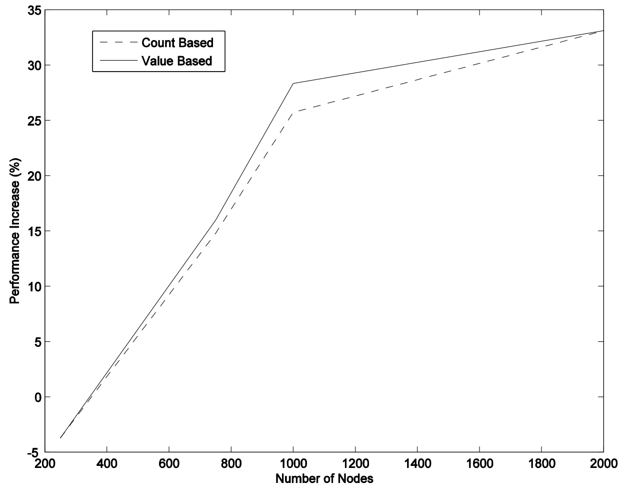

Fig. 4. Performance increase changes with respect to the number of nodes using the Faster Torrent protocol in comparison to the original BitTorrent protocol. X-axis shows the number of nodes and Y-axis shows the performance increases in percentage. Dashed line shows the count-based protocol and the solid line shows the value-based protocol.

upload the valued pieces to others, around 25 nodes achieved to get over 60% valuable pieces and 9 of them got over 70% of the valuable pieces. This shows that, if one employs value-based barter on the BitTorrent block exchange mechanism, it is possible to fairly distribute the value among the peers in the system. This explains the reason we called the protocol the Fairer Torrent Protocol.

5.3. Faster Torrent Protocol Analyses

In these analyses, our purpose is to show that the file is distributed more in to the system and we also expect an increase in the average download speed. As in the previous protocol, we compare the original BitTorrent with our protocol based on the buy method. The buy method is also implemented with ratio based on piece count and ratio based on values of the pieces. In order to test our system, we started with 100 nodes. We run the simulator for 10000 seconds and configured the network dynamics such that it adds 20 nodes with probability 0.5 every 100 seconds. The network dynamics is also responsible for removing the nodes from the system, but removal depends on the protocol that we use (explained below). At the end of 10000 seconds, we check the number of nodes who have entire file, the average download time, and the average time a node spends in the system (including the seeding time).

We also tested our Faster Torrent Protocol with many other configurations. We observed that as the number of nodes increases through the 2000 nodes, the average download time performance increases up to 33% compared to original BitTorrent protocol in terms of average download time (see Figure 4). Table 1 shows the performance increases with respect to maximum 1000 nodes allowed in the system. On the other hand, we also observed that the configuration on the file size does not affect the performance of our system.

Table 1. 1000 nodes enter the system. Initially only 1 seeder exists. Columns left to right show the tested protocols, the number of nodes who downloaded the entire file, the average download time, and the average time in the system, respectively.

Protocol	Number of Nodes	Avg. Download Time	Avg. Time
Original BitTorrent	756	3323 sec	3553 sec
Count-Based	869	2469 sec	5133 sec
Value-Based	909	2381 sec	5017 sec

Single Seeder: We first performed our test with the *flash crowd scenario*, using only one seeder. The nodes that enter the system have 0% of the file. As it is seen in Table 1, when the simulation terminates, BitTorrent has the lowest number of completed nodes. Average download speed is also the lowest in the original BitTorrent because it is hard to find nodes to download from, since there is only one seeder and many nodes leave after they finished their download. On the other hand, the buy method forces the nodes to upload until their ratio is greater than or equal to 1. The advantage of this is that the nodes have around 30% better average download times in the end. The drawback of this mechanism is that, nodes stay in the system more than they do in original BitTorrent. Yet, this drawback does not prevent nodes from using the file while they are in the system. Furthermore, in the buy method based on the piece count, nodes did not spend more network resources than they download, since they only upload until ratio is equal to one. In this way, fairness is achieved one to one for each node and this also causes newly arrived nodes to finish faster than the ones in the original BitTorrent protocol. In buy method with value-based ratio, although nodes may spend more network resources if they keep uploading lesser valued pieces, they can also spend less bandwidth by uploading more valuable pieces in to the system. This provides a better distribution of valuable files into the system. The reason that nodes spend less time in the system with buy method based on values is that they can reach the ratio of one in a faster manner by uploading valuable pieces more.

More Seeders: Our tests with 10 seeders in Table 2 also indicate better results with respect to the number of completed nodes and the average download times. By comparing Table 1 and Table 2, we conclude that having more seeders increase the number of completed nodes by 43 in the original BitTorrent, while increasing the number of completed nodes by 84 and 44 respectively by using buy method based on piece count and buy method based on values of pieces. Note that our value-based buy method solution still performs around 30% better than the original BitTorrent.

Table 2. 1000 nodes enter the system. Initially 10 seeders exist. Columns left to right show the tested protocols, the number of nodes who downloaded the entire file, the average download time, and the average time in the system, respectively.

Protocol	Number of Nodes	Avg. Download Time	Avg. Time
Original BitTorrent	799	3433 sec	3683 sec
Count-Based	953	2466 sec	5657 sec
Value-Based	952	2416 sec	5279 sec

All these tests showed us that our Faster Torrent Protocol provides better distribution of files in the network, and this comes with the benefit of better average download times. As future work, we plan to test our system on a real-life client implementation and investigate putting values on pieces based on real measurements directly.

6. Conclusion

In this paper, we presented two modifications to the BitTorrent protocol. Our modifications are based on the fairness definition where the upload/download ratio has to be at least one. With our tests, we observed the effects of enforcing such ratio in the BitTorrent protocol. As a result of our experiments, our Fairer Torrent Protocol, implemented using the value-based barter protocol, provided security against adversaries who only requests valuable pieces in the system. There were many other peers who managed to perform equally well as the adversary, and hence this means the adversarial behavior is not beneficial and peers should stick to our protocol. Our Fairer Torrent Protocol performs a much **fairer distribution of the value among the peers** in the system.

Our Faster Torrent Protocol, implemented using the value-based buy protocol, provided a **performance increase in download rates around 30%**. In both of our modifications, we achieved **better results with value-based buy and barter protocols than their count-based versions**. We achieved this by considering different piece values in a single torrent for the first time, and modifying existing cryptographic fair exchange protocols to work properly under this value-based setting.

As future work, we plan to test our system on a real-life client implementation and also perform tests in order to observe the effects of our modifications on network consumption. We further plan to investigate putting values on pieces based on real measurements directly, though this was outside the scope of this paper.

Acknowledgements

The authors would like to acknowledge the support of TÜBİTAK, the Scientific and Technological Research Council of Turkey, under grant number 111E019, and the European Union COST Action IC1306. The authors also acknowledge Koç University for the "Best Senior Project" award.

References

[1] Jun S, Ahamad M. Incentives in bittorrent induce free riding. In 2005 ACM SIGCOMM Workshop on Economics of Peer-to-peer Systems; 22 August 2005; New York, NY, USA: ACM. pp. 116–121.

[2] Liogkas N, Nelson R, Kohler E, Zhang L. Exploiting bittorrent for fun (but not profit). In 2006 International Workshop on Peer-to-Peer Systems; 27-28 February 2006; Santa Barbara, CA, USA.

[3] Locher T, Moor P, Schmid S, Wattenhofer R. Free riding in bittorrent is cheap. In 2006 Fifth Workshop on Hot Topics in Networks (HotNets-V); 29 November 2006; Irvine, CA, USA. pp. 85-90.

[4] Levin D, LaCurts K, Spring N, Bhattacharjee B. Bittorrent is an auction: Analyzing and improving bittorrent's incentives. Comp Comm R 2008; 38: 243–254.

[5] Marti S, Garcia-Molina H. Taxonomy of trust: Categorizing p2p reputation systems. Comput Netw 2006; 50: 472–484.

[6] Zhu B, Jajodia S, Kankanhalli MS. Building trust in peer-to-peer systems: a review. Int J Sec Netw 2006, 1: 103–112.

[7] Dingledine R, Mathewson N, Syverson P. Reputation in p2p anonymity systems. In 2003 Workshop on Economics of Peer-to-Peer Systems; 5-6 June 2003; Berkeley, CA, USA.

[8] Friedman EJ, Resnick P. The social cost of cheap pseudonyms. J Econ Manage Strat 2000; 10: 173–199.

[9] Marti S, Garcia-Molina H. Identity crisis: Anonymity vs. reputation in p2p systems. In 2003 International Conference on Peer-to-Peer Computing; 1 September 2003; Washington, DC, USA: IEEE. pp. 134–141.

[10]Küpçü A, Lysyanskaya A. Usable optimistic fair exchange. Comput Netw 2012; 56: 50 – 63.

[11]Belenkiy M, Chase M, Erway CC, Jannotti J, Küpçü A, Lysyanskaya A, Rachlin E. Making p2p accountable without losing privacy. In 2007 ACM Workshop on Privacy in Electronic Society (WPES); 29 October 2007; Alexandria, VA, USA. New York, NY, USA: ACM. pp. 31–40.

[12]Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. 2008.

[13]Cohen B. Incentives build robustness in bittorrent. In 2003 Workshop on Economics of Peer-to-Peer Systems; 5-6 June 2003; Berkeley, CA, USA. pp. 68-72.

[14]Camenisch J, Lysyanskaya A, Meyerovich M. Endorsed e-cash. In 2007 IEEE Symposium on Security and Privacy; 20-23 May 2007; Oakland, CA, USA. Washington, DC, USA: IEEE. pp. 101–115.

[15]Camenisch J, Shoup V. Practical verifiable encryption and decryption of discrete logarithms. In 2003 Annual International Cryptology Conference; August 17-21, 2003; Santa Barbara, California, USA: Springer-Verlag. pp. 126–144.

[16]Yue Y, Lin C, Tan Z. Analyzing the performance and fairness of bittorrent-like networks using a general fluid model. Comput Commun 2006; 29: 3946–3956.

[17]Fan B, Lui JCS, Chiu DM. The design trade-offs of bittorrent-like file sharing protocols. IEEE/ACM Trans Netw 2009; 17: 365–376.

[18]Douceur JR. The sybil attack. In 2002 Revised Papers from the First International Workshop on Peer-to-Peer Systems; 1 January 2002; London, UK: Springer-Verlag. pp. 251–260.

[19]Sherman A, Nieh J, Stein C. Fairtorrent: Bringing fairness to peer-to-peer systems. In 2009 International Conference on Emerging Networking Experiments and Technologies, Rome, Italy. New York, NY, USA: ACM. pp. 133–144.

[20]Rahman R, Meulpolder M, Hales D, Pouwelse J, Sips H. Improving efficiency and fairness in p2p systems with effort-based incentives. In 2010 IEEE International Conference on Communications; 23-27 May 2010; Cape Town, South Africa. Washington, DC, USA: IEEE. pp. 1-5.

[21]Asokan N, Shoup V, Waidner M. Optimistic fair exchange of digital signatures. IEEE J Sel Area Comm 1998; 18: 591–606.

[22]Pagnia H, Gartner FC. On the impossibility of fair exchange without a trusted third party. Technical report TUD-BS-1999-02, Darmstadt University of Technology, 1999.

[23]Thommes RW, Coates MJ. Bittorrent fairness: analysis and improvements. In 2005 Workshop on Internet, Telecom and Signal Processing; December 2005; Noosa, Australia.

[24]Eichelberg M, Aden T, Riesmeier J, Dogac A, Laleci GB. A survey and analysis of electronic healthcare record standards. ACM Comput Surv 2005; 37: 277–315.

Appendix

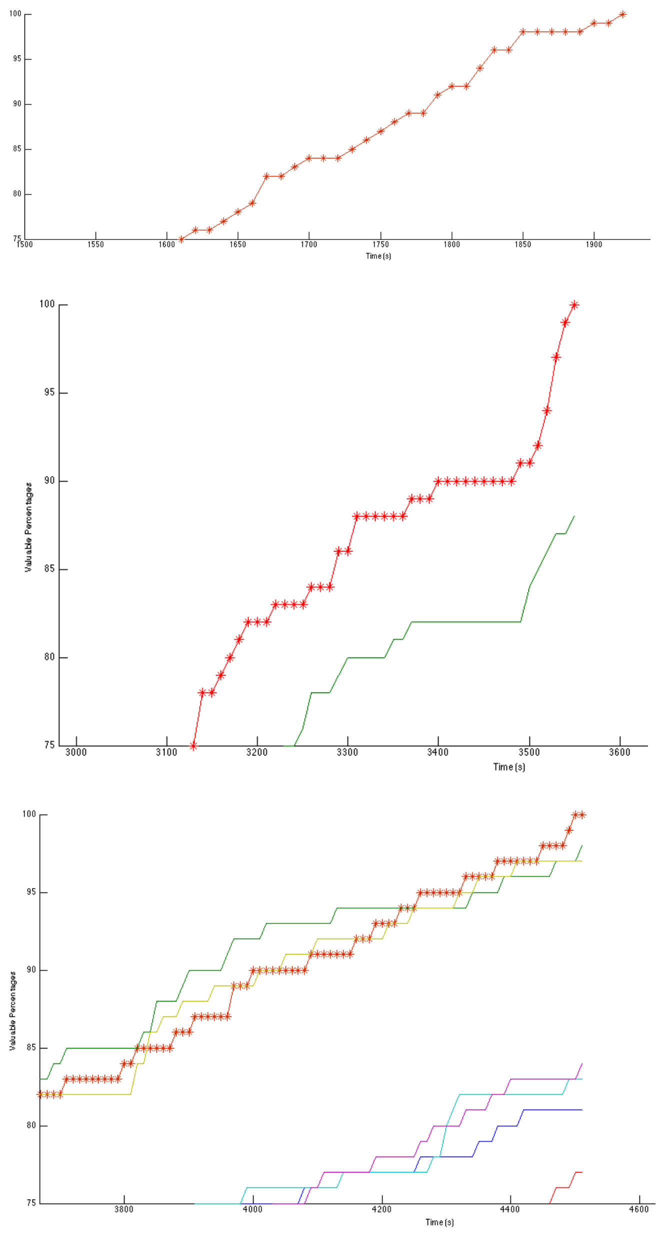

Fig. A. Zoomed-in versions of Figures 1, 2, and 3, respectively.