DS-Bit: Delay-Sensitive BitTorrent-Based Data Synchronization Protocol

Genwei Zhu\textsuperscript{1}, Yongli Wang\textsuperscript{1}\textsuperscript{*}

School of computer science and engineering, Nanjing University of Science And Technology, Nanjing, 210000, China
yongliwang@njust.edu.cn

Abstract. The main purpose of BitTorrent is to download files for individual users, and there are problems such as slow startup speed, insensitive to time delay, and slow synchronization of large-scale network data. This paper analyzes the bottleneck of BitTorrent's technical framework, and proposes delay-sensitive BitTorrent-based data synchronization protocol (DS-Bit). With the Pre-Connect strategy, we significantly reduce the impact of the delay between nodes in the system on the data synchronization time by sending messages in advance, and the delay-considered TFT rule (DCTFT) that considers the delay optimizes the choice of nodes to send data. Because the transmission rate of DS-Bit is too fast, if it is used for file downloads for individual users, the node that completed the first will exit the system early, which will reduce the total download rate of the system. Therefore, DS-Bit is mainly used for data synchronization between distributed servers.

1. Introduction
Today, when massive data transmissions are becoming more and more frequent, the network bandwidth or caching mechanism of servers that have been continuously added cannot fundamentally solve the needs of users, so the P2P (Peer-to-Peer) technology was born\cite{1}. In fact, P2P technology can be used not only in the file download of individual users, but also in data distribution between the nodes of blockchain, CDN file preheating, etc. At present, the most popular P2P protocol is BitTorrent. Although the BitTorrent protocol has been very mature, it still has some shortcomings such as slow startup speed, insensitive to delay, and slow synchronization of large-scale network data. In this paper, we analyse the shortcomings of BitTorrent and introduce the delay-sensitive BitTorrent-based data synchronization protocol (DS-Bit). We also show that DS-Bit is more fair than traditional BitTorrent.

2. Related Work
In 2003, B. Cohen first proposed the BitTorrent \cite{2} system in the P2P Economics Workshop. Since then, the BitTorrent system has not only been paid attention to in practical applications, but also became the focus of academia. \cite{3} proposed a simulation-based BitTorrent research and improved the TFT fairness strategy, but the research on the best number of concurrent uploads does not finish. \cite{4} implements a BitTorrent simulation program based on PeerSim platform and opened the source code, which is convenient for scholars to study BitTorrent better. \cite{5} studied the application of BitTorrent in enterprise networks, which reduced the bandwidth consumption and the download time in enterprise server clusters. What’s more, it provides stronger robustness for distribution in WAN connections when some failure occurs.
Since the traffic in the P2P network will occupy the bandwidth to a large extent, there are also many studies focus on the P2P traffic in the network. For example, [6] proposed clustering and traffic dispersion graph-based method (CTDG), which the accuracy of P2P traffic identification in high-speed backbone networks can reach 95 percent.

All the studies introduced above focused on using BitTorrent as a download tool for individual users. The characteristic of individual user downloading is that each node tries its best to ensure that the itself completed download first and minimize the occupation of its own bandwidth, which means that the node exits the BitTorrent system as soon as the downloading is completed, and then no longer contribute its data. However, when the server synchronizes data, the node that completed downloading will not exit the system, and still contribute its own data and bandwidth to the entire system. We analysis the traditional BitTorrent protocol and found some disadvantages as following:

- **Start Slowly** Because of the exist of the period of computing the “unchoke” nodes, Traditional BitTorrent starts slowly. Although the time for system starting seems to be insignificant compared with the total time of all the peers finish downloading when the file to be synchronized is large. When the file is small, the time for system starting can not be ignored.

- **Delay Insensitivity** In the real network environment, the Delay between nodes can not be ignored. In traditional BitTorrent system, nodes need to wait for the messages from their neighbors. And nodes can do nothing while waiting. This do not even affect the waiting node itself, but also affects the efficiency of data transmission of the entire system. However many studies in P2P did not consider the Delay as an important factor affecting data transmission, which means that when the network is very complex, those delay-insensitive protocols may not achieve the best results.

- **Inflexible** Nodes in traditional BitTorrent system use tit-for-fat (TFT) to choose “unchoke” neighbors. But TFT only consider the download rate from other nodes. Since that during the connection, nodes do not transfer data all the time but spend a lot of time waiting for message and data cause the transmission delay, just considering the download rate is one-sided. When two nodes are far apart, the delay between them is very large. In the same time, the node with the fast transmission rate may not contribute more data than the node with the slow transmission rate but the greater delay. So only using the traditional TFT rule-making method can not achieve the purpose of TFT. As the network scale gradually increases, the delay between nodes gradually deepens the impact of the transmission rate, and the traditional TFT cannot adapt to the situation.

BitTorrent has great advantages for multi-node data distribution, so it is necessary to improve the BitTorrent protocol for data synchronization between servers.

### 3. Definition and Model

In this part, we introduce some terms and the model of DS-Bit.

#### 3.1. Definition of Terms

DS-Bit is a complex protocol. In order to describe DS-Bit better, we have clearly defined the following terms.

- **Def. 1. Peer** is the device entity that distributes data in a P2P network.

- **Def. 2. Piece** is the smallest unit of transmission. Each file that needs to be distributed is divided into pieces of several sizes (usually 32-256kb).

- **Def. 3. Torrent** is the “.torrent” file which records all Piece information of the file to be distributed and the address of the Tracker.

- **Def. 4. Tracker** is the server responsible for recording the status of each node in the P2P network.

- **Def. 5. Neighbour Set** is the set of neighbours. Each node maintains a set of neighbours and only requests data from its neighbours.

- **Def. 6. Unchoke and Choke**. Unblocking means that node A agrees to send data to node B. If it does not agree, it means blocking.
Def. 7. *Tit-for-Fat* (TFT) refer to the traditional Bit-Torrent node selection rules for “unchoke” nodes. A node will choose the node that provides the fastest download rate to upload data, that is, the node that contributes the most gets the most.

Def. 8. *Local Rarest First* (LRF) is the rule that a node selects the requested Piece. The node first traverses the piece information of its entire neighbour list, and finds the rarest piece to request to its neighbours.

3.2. Model of LS-Bit Implementation

As the Fig. 1 shows, DS-Bit mainly consists of three parts: the message manager, the data transmitter and the neighbor manager. The message manager is responsible for receiving messages sent by other nodes, classifying them, and sending messages to other nodes. The message sent by nodes drives the operation of the entire system. Of course, to reduce the loss of performance due to the delay is the goal of this paper. The neighbor manager is to acquire new neighbors and monitor the status of existing neighbors. The data transmitter is responsible for upload and download of pieces.

![Fig. 1 DS-Bit system: main modules](image)

4. DS-Bit Protocol

To resolve the problems of traditional BitTorrent shown in Section 2, We propose DS-Bit: Delay-Sensitive BitTorrent-Based Data Synchronization Protocol to achieve the following goals: 1) Reduce the time wasted due to waiting for messages; 2) Maximize the use of bandwidth for the purpose of data synchronization; 3) Fair transmission (nodes with high bandwidth can complete the download faster than nodes with low bandwidth). Fig. 2 shows the overall process of LS-Bit protocol data transmission. The specific improvements will be introduced in detail in the following content of this section.

4.1. Pre-Connect Strategy

As shown in Fig. 2, in traditional BitTorrent, when received an “unchoke” from node $B$, node $A$ first calculate the rarest pieces and arrange them into a list. Then $A$ request pieces to $B$ in order. Only after downloading a piece can the next request message be sent. This strategy ensures that each request is of the most needed piece currently, but each download of a piece must wait for one RTT. While waiting for the message, the bandwidth is wasted.

To reduce the wasted time, we propose a strategy called *Pre-Connect*. When a piece upload is about to be completed, the sender will send a message in advance to notify the receiver that it is ready
to receive a new piece. This message is called Pre-Con-Info which contains a timer that is responsible for the countdown of sender’s uploading completion. When the countdown completed, receiver immediately send the request message to the sender. At this point, when the piece from the sender has not completely been reached the receiver the receiver has already sent a request to the sender. The Pre-Connect strategy reduce the “jitter” caused by the message transmission delay, so that the entire system can make better use of bandwidth.

Node A (Receiver)       Node B (Sender)

Establish TCP Connection
Handshake
“Interest” Message
DS-Bit: Get Transmission Delay
BitTorrent: TFT

Select Rarest Piece
Request Piece
Piece

DS-Bit: Send Request In Advance
DS-Bit: “Pre-Con-Info” message
Request Piece
Piece

DS-Bit: “Pre-Con-Info” message

Fig. 2 The overall process of DS-Bit

4.2. Delay-Considered TFT (DCTFT)

The TFT strategy used by traditional BitTorrent protocol only considers the download rate, without considering the delay. In a large-scale network, the communication between nodes may need to pass through several switches. At this time, the time wasted by the message transmission cannot be ignored. In order to solve the problem, we propose an algorithm for senders to select the nodes to upload called DCTFT: delay-considered TFT. The main idea of the algorithm is to calculate and sort the theoretical time of that a piece transmit from the sender to the receiver, and then select a certain number of nodes with the least theoretical time to send the “unchoke” message. The specific process is shown in Algorithm 1.

```
Algorithm 1: DCTFT
1. Input: neighborList[], number
2. Output: uploadSet[]
3. Begin
4. uploadSet[] ← empty
5. for neighbor in neighborList[] do
6. time ← latency + (Piece.size / downloadSpeed)
7. end for
8. for i ← 0 to number do
9. pick the i-th least time of neighbor ∈ neighborSet
```

...
10. end for
11. return uploadSet[]
12. END

Through the DCTFT algorithm, the sender will give preference to the node with lower transmission delay when the download rate of several nodes is the same. This algorithm can be easily understood from a simple network. As shown in Fig. 3 (a), the delay between A and B is 20ms while the download rate is 2Mbps; the delay between A and C is 80ms while the download rate is 4Mbps; the delay between B and C is 25ms while the download rate is 4Mbps. Assuming that A, B, and C can only transmit data to one node at the same time, A has a unique Piece. Two nodes B and C request the Piece from nodes other than themselves at the same time. The traditional TFT rule only considers the transmission rate, so the transmission order is shown in Fig. 3(b): A→C, C→B, the total time required is 162.5ms. The delay is also considered by DCTFT, and the transmission order of DS-Bit is shown in Fig 3(c): A→B, B→C, and the total time required is 136.25ms. Compared with the traditional TFT, DCTFT improved the performance by 26.25ms. The situation analyzed above is just a hypothesis and we will verify the measures by experiments in Section 5.

```
Piece : 128KB
A Delay : 20ms
  Rate : 2Mbps
  B

C
```

(a) a Hypothetical Network Topology

```
A
  total time : 162.50ms
  TFT order
  C
```

(b) The Transmission Order of TFT

```
A
  total time : 136.25ms
  DCTFT order
  C
```

(c) The Transmission Order of DCTFT

Fig.3 The transmission order in BitTorrent and DS-Bit

5. Performance Evaluation

In order to test the performance of DS-Bit, we built a simulator based on an Event-Driven engine of PeerSim to simulate a file distributed from the data source to all the peer nodes in a P2P network and record the corresponding data. Our experiment ran on a 3.6GHz, 16GB RAM machine.

5.1. Simulator Details and Metrics

In order to improve the scalability of the simulation and control the unrelated disturbance factors of the actual network, we simplified the network model as following:

1. Assume that the backbone network has unlimited capacity (regardless of the shared bottleneck link in the core network). Research [7] shows that the bottleneck links almost all occur on the links near the end system.

2. Assume that the TCP connections between nodes are stable. The TCP establishment between nodes only takes 1.5 RTTs, which is too small compared to the time required for file transmission. Once a connection established, it can be maintained for a long time.

3. Ignore malicious traffic in real networks and unknown traffic from other sources.

4. Ignore the bandwidth occupied by the message propagation between nodes.
(5) After a node complete downloading (becoming a Seed node), it will not exit the system. In file download scenario, the node will exit the system after the download is complete. For individual users, they only want to ensure that their data can be downloaded as soon as possible. If it is a long-running server, this problem will not occur.

To simulate the heterogeneous network in reality better, we set up four types of nodes with different download rates: 640Kbps, 1024Kbps, 2048Kbps, 4096Kbps (the number of each node accounts for 4/1 of the number of summary points), the upload bandwidth of each node is the same as its download bandwidth. The delay between nodes is between 10ms and 400ms system time.

Unless otherwise specified, we use the following settings in our experiments. First, we use a file size of 150MB with a block size 128KB. The number of neighbour set is 40 in both the DS-Bit and BitTorrent system and the maximum number of concurrent uploads for each node is 5. Second, there is only one tracker and one seed in both system when the simulation began. Third, each group of experiments were run randomly 10 times, and we recorded the system status every 100ms system time during the simulation, and the average value of the experimental results was recorded.

The metrics used in our experiments is shown as following:
- Total time of the file synchronization, which means the time from the system initialization to the file distribution completed.
- Upload bandwidth utilization, which means the percentage of bandwidth occupied to the total bandwidth.
- CDF of download completed peers, which means the number of downloading completed peer over time.

5.2. Reliability Analysis of Experiments

System error and random error. In our experiments, the data was recorded every 100ms system time, so the system error is 100ms. However, in our experiments the final results are all above 300s, and the system error was within the acceptable range.

The influence of variable factors. In our experiments, we set the same random seeds of each group of experiments. For example: In the DS-Bit and BitTorrent systems, a 15MB file is distributed to 400 peers. In this way, all basic config of the two experiments are the same, which can eliminate the influence of variable factors on the experimental results to the greatest extent. We ran 10 times for each group of experiment and recorded the average of the 10 results as the final result.

5.3. Experimental Results

The experimental results are presented in several aspects: total time of file synchronization, the bandwidth utilization and CDF of download completed peer.

5.3.1. Total Time of the File Synchronization

We simulated the distribution of files with size of {15MB, 30MB, 50MB, 150MB, 200MB, 300MB, 400MB} from a source to all the peers of a network with size of {15, 30, 50, 100, 150, 200, 300, 400, 500, 600, 700, 800, 900, 1000} in both BitTorrent and DS-Bit system. Fig. 4 and Fig. 5 respectively shows the total time of the files of different size synchronized to different networks in BitTorrent and DS-Bit. It can be seen that the performance of DS-Bit is much better than traditional BitTorrent. When the file size reaches 400MB and the network size reaches 1000, the time of BitTorrent is over 1.5 times the time of DS-Bit. The feature that network size has less impact on the total time than file size is reflect both in BitTorrent and DS-Bit, but it is more obvious in DS-Bit System. When the network size increase to a certain scale, the total time is hardly affected by it.
5.3.2. Bandwidth Utilization and CDF
To study the details of data distribution in DS-Bit and BitTorrent systems, we selected the experimental results with the file size of 150MB and the network size of 600 and plot the line graphs of CDF of download completed peers and bandwidth utilization over time.

As shown in Fig. 6, the bandwidth utilization in DS-Bit system reaches the maximum much faster than in traditional BitTorrent. The peak value which is over 80 percent of DS-Bit is bigger than that of traditional BitTorrent which is about 50 percent. It shows that the data of DS-Bit is scattered faster. The more nodes have data, the more nodes can provide their own uplink bandwidth.

Fig. 7 also shows that the first node to complete the download in DS-Bit system use a shorter time than that in BitTorrent. And the node that has completed the download becomes a new seed that only upload pieces, do not request pieces, which means that these nodes leave more download opportunities to nodes whose download rate is low. Compared with DS-Bit, the nodes in BitTorrent tend to complete the download at the same time, indicating that the nodes with more uploads are not ahead of the nodes that upload less in terms of download completion time. DS-Bit is more fair, the more data that a node contributed to other nodes, the faster its download rate is.

6. Conclusion
In this paper, we designed and implemented DS-Bit (delay-sensitive BitTorrent-based data synchronization protocol) in PeerSim for data synchronization between servers. The main improvements of DS-Bit are the Pre-Connect strategy and DCTFT. The Pre-Connect strategy greatly reduces the impact caused by the delay between nodes on data transmission. DCTFT optimizes the choice of node uploading pieces. In order to verify the effectiveness of DS-Bit, we designed multiple
group experiments and conducts the reliability analysis of the experiments. As far as possible, the influence of variable factors on the experimental results were eliminated. The experimental results verified that the performance of DS-Bit better and the completion time of nodes is more fair.

DS-Bit has great advantages in data synchronization in heterogeneous networks with a large number of nodes and large delays. In the future, it can be used not only for data synchronization between nodes of ordinary distributed database systems, but also for CDN. However, DS-Bit still has some shortcomings, such as increasing the complexity of the protocol. In the future, we can continue to improve DS-Bit from other protocols such as Bee[8] and Crew[9].

Acknowledgments
This work has been awarded by the National Natural Science Foundation of China (61941113), the Fundamental Research Fund for the Central Universities (30918015103, 30918012204), Nanjing Science and Technology Development Plan Project (201805036), and “13th Five-Year” equipment field fund (61403120501), China Academy of Engineering Consulting Research Project(2019-ZD-1-02-02), National Social Science Foundation (18BTQ073), State Grid Technology Project (5211XT190033).

References
[1] Li Zupeng, et al. The Development and Prospect of P2P Network Technology. Telecommunications Science, 2003, 19.3: 1-55.
[2] COHEN, Bram. Incentives build robustness in BitTorrent. In: Workshop on Economics of Peer-to-Peer systems. 2003. p. 68-72.
[3] BHARAMBE, Ashwin R.; HERLEY, Cormac; PADMANABHAN, Venkata N. Analyzing and improving a bittorrent networks performance mechanisms. In: Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications. IEEE, 2006. p. 1-12.
[4] FRIOLI, Fabrizio; PEDROLLI, Michele. A BitTorrent module for Peersim. Internet http://peersim.sourceforge.net/code/bittorrent.tar.gz, 2008.
[5] COSTA, Marco Antonio D.'Alessandro; RUBINSTEIN, Marcelo Gonçalves. Performance analysis of a locality-aware BitTorrent protocol in enterprise networks. Peer-to-Peer Networking and Applications, 2019, 12.4: 751-762.
[6] Su Yangyang, et al. P2P traffic identification method based on clustering and traffic dispersion graph. Application Research of Computers, 2019, 11.[2018-08-10].http://www.arocmag.com/article/02-2019-11-053.html.
[7] AKELLA, Aditya; SESHAN, Srinivasan; SHAIKH, Anees. An empirical evaluation of wide-area internet bottlenecks. In: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement. 2003. p. 101-114.
[8] WU, Chi-Jen, et al. Bee: A Best Effort Peer-to-Peer Delivery Protocol for Internet Data Dissemination Application. Technical Report (TR-IIS-06-015-) IIS Academia Sinica, 2006.
[9] DESHPANDE, Mayur, et al. ReCREW: A reliable flash-dissemination system. IEEE Transactions on Computers, 2012, 62.7: 1432-1446.