Case Study about Basic Torrent using SICSIM

Project Report

Uwe Dauernheim[†]
uwe@dauernheim.net

Alex Averbuch[†]
alex.averbuch@gmail.com

[†] EP2400 - Network Algorithms

November 18, 2009

1 Introduction

As part of this project a simplified, centralized version of BitTorrent was developed and evaluated using *SICSIM*[1], a Java-based network simulator.

This report briefly outlines the protocol in Section 2, introduces important definitions in Section 3, describes the experiments performed in Section 4, reasons about the results in Section 5, and concludes with a brief discussion in Section 6.

2 Protocol

The protocol under analysis is loosely based on BitTorrent, although greatly simplified.

As in BitTorrent, the protocol employs a centralized Tracker, but in this case the Tracker maintains a global view of all peers' state.

Piece selection and seeder selection strategies are random and performed by the Tracker only.

Peers send requests for pieces to the Tracker. The Tracker then responds with instructions comprising of: a random piece (that peer does not already have) to download next, and the address of the seeder for that piece.

Peers may not download multiple pieces from the same seeder in parallel.

3 Definitions

Before presenting the experiments and results, it is important to define a number of concepts. The following definitions are provided as a reference for Section 4 and Section 5.

In all experiments the simulated network links had a latency of:

$$T_{\text{network}} = 6 \pm 3\text{ms} \tag{1}$$

Churn defines the amount of peers joining/leaving the network, and the frequency of these events:

$$\text{Churn} = 0\% \ldots 100\% \tag{2}$$

In this protocol, files are divided into pieces. The size of each piece is given by:

$$\text{Size}_{\text{piece}} = \frac{\text{Size}_{\text{file}}}{\text{Count}_{\text{pieces}}} \tag{3}$$

[1] http://www.sics.se/~amir/sicsim/

In this protocol a peer's upload bandwidth is statically and evenly shared across its upload slots. The time required to simply transfer one piece between two peers is dependent on several factors, including piece size and the upload bandwidth available to one slot. This ignores any overhead incurred by protocol messages and is given by:

$$T_{\text{transfer}} = \frac{\text{Size}_{\text{piece}}}{\left(\frac{\text{BW}_{\text{up}}}{\text{Slots}_{\text{up}}}\right)} \tag{4}$$

The time required to perform all protocol interactions that support a piece transfer can be significant. There are six messages, including: sending request to/from tracker, performing handshake with seeder, initiating download from seeder, and receiving complete notification from seeder. This ignores any processing time between messages and also ignores the time needed to actually transfer the piece. It is given by:

$$T_{\text{protocol}} = 6 \times T_{\text{network}} \tag{5}$$

The time required to download a piece, excluding any processing time but including all protocol messages and the actual piece transfer, is given by:

$$T_{\text{download}} = T_{\text{protocol}} + T_{\text{transfer}} \tag{6}$$

Slot utilization defines how many of its available upload slots a peer is using. It is given by:

$$\text{Util}_{\text{slots}} = 0\% \ldots 100\% \tag{7}$$

It is unlikely that a peer is always utilizing all of it's available upload bandwidth. The amount of upload bandwidth a peer is utilizing at any given time is given by:

$$\text{BW}_{\text{peer}} = \text{BW}_{\text{up}} \times \text{Util}_{\text{slots}} \tag{8}$$

The benefit of peer-to-peer systems is realized by harnessing the resources of all peers. In this protocol, as more peers seed file pieces, the available global system bandwidth ($\text{BW}_{\text{system}}$) increases. However, it is difficult to accurately describe how $\text{BW}_{\text{system}}$ changes with time without simplifying the problem.

Assume that: pieces are uniformly distributed across all peers as soon as possible, all pieces are equally available in the network, and peers equally share the load of seeding as soon as possible. Given these assumptions this basic equation can be used to model the change in $\text{BW}_{\text{system}}$ with time:

$$\begin{aligned} T &= i \times T_{\text{download}} \\ \text{BW}_{\text{system}}^{T_0} &= \text{BW}_{\text{up}} \\ \text{BW}_{\text{system}}^{T} &= (\text{Slots}_{\text{up}} + 1) \times \text{BW}_{\text{system}}^{T-1} \end{aligned} \tag{9}$$

4 Experiments

Experiments were carried out to determine how various protocol parameters affect performance. This section details these experiments.

4.1 Slot Count

Slots_{up} and $\text{Slots}_{\text{down}}$ dictate how many pieces a peer can upload/download simultaneously.

These parameters were varied to gauge the effect they had on piece dissemination.

Experiment settings can be seen in Table 1.

Peers did not begin to download pieces until all peers had joined the network. This was done in an attempt to eliminate all variables from the experiment, other than the slot count.

Data collection began as soon as the last peer joined the network.

Parameter	**Value**
Churn	0%
Network Size	100
Peer Inter-arrival Time	10
BW_{up}	32
BW_{down}	Unlimited
$\text{Count}_{\text{pieces}}$	10
$\text{Size}_{\text{piece}}$	64
Slots_{up}	1-12
$\text{Slots}_{\text{down}}$	1-12

Tab. 1: Experiment Settings - Slot Count

4.2 Network Size

The number of peers in the system was varied. Experiment settings can be seen in Table 2.

Parameter	**Value**
Churn	0%
Network Size	1-1000
Peer Inter-arrival Time	10
BW_{up}	128
BW_{down}	Unlimited
$\text{Count}_{\text{pieces}}$	10
$\text{Size}_{\text{piece}}$	64
Slots_{up}	4
$\text{Slots}_{\text{down}}$	4

Tab. 2: Experiment Settings - Network Size

Peers did not begin to download pieces until all peers had joined the network. This was done in an attempt to eliminate all variables from the experiment, other than the number of peers. If this method had not been used, time taken to complete the experiment would have depended almost entirely on the time required for every peer to join the network.

Data collection began as soon as the last peer joined the network.

4.3 Upload Bandwidth

Clearly, peers' upload bandwidth has an effect on how fast pieces can be exchanged in the network. To explore effects related the upload bandwidth, this parameter was varied and experiments were performed.

Experiment settings can be seen in Table 3.

Parameter	**Value**
Churn	0%
Network Size	100
Peer Inter-arrival Time	10
BW_{up}	32-1024
BW_{down}	Unlimited
$\text{Count}_{\text{pieces}}$	10
$\text{Size}_{\text{piece}}$	64
Slots_{up}	4
$\text{Slots}_{\text{down}}$	4

Tab. 3: Experiment Settings - Upload Bandwidth

Peers did not begin to download pieces until all peers had joined the network. This was done in an attempt to eliminate all variables from the experiment, other than the upload bandwidth.

Data collection began as soon as the last peer joined the network.

4.4 Piece Count

Tests were performed to research the effects of varying $\text{Count}_{\text{pieces}}$. Note that $\text{Size}_{\text{file}}$ is fixed in these experiments, so $\text{Size}_{\text{piece}}$ is given by Equation 3.

These tests were performed to identify what affects $\text{Count}_{\text{pieces}}$ has on the amount of piece transfers that can be performed in parallel, and the progress of the protocol in general.

Experiment settings can be seen in Table 4.

Peers did not begin to download pieces until all peers had joined the network. This

Parameter	**Value**
Churn	0%
Network Size	100
Peer Inter-arrival Time	10
BW_{up}	64
BW_{down}	Unlimited
$\text{Count}_{\text{pieces}}$	600, 1200, 2400
$\text{Size}_{\text{piece}}$	1-50
Slots_{up}	4
$\text{Slots}_{\text{down}}$	4

Tab. 4: Experiment Settings - Piece Count

was done in an attempt to eliminate all variables from the experiment, other than the upload bandwidth.

Data collection began as soon as the last peer joined the network.

4.5 Churn

The amount of churn present in the system was varied. The experiments were performed using reliable seeder: Initial seeder never leaves or fails. Experiment settings can be seen in Table 5.

Parameter	**Value**
Churn	0%-70%
Network Size	100
Peer Inter-arrival Time	10
BW_{up}	64
BW_{down}	Unlimited
$\text{Count}_{\text{pieces}}$	10
$\text{Size}_{\text{piece}}$	64
Slots_{up}	4
$\text{Slots}_{\text{down}}$	4

Tab. 5: Experiment Settings - Churn

Peers began to download pieces as soon as they joined the network and data collection began as soon as the first peer joined the network. This was done in an attempt to simulate a real-world scenario.

4.6 Leaving Seeders

Tests were performed to research the effects of varying the amount of time that a peer stays in the network after downloading all pieces. This attempts to simulate reality, where downloaders often only connect to the network for as long as they have something to gain. This experiment is intended to highlight the protocol's reliance on the "good will" of peers.

The amount of time a peer remains in the network after becoming a seeder (T_{seeding}) was varied. After this predefined time, the seeding peers left the network.

Experiments were performed using an unreliable initial seeder: Initial seeder may leave or fail, to simulate a more realistic scenario. Experiment settings can be seen in Table 6.

Parameter	**Value**
Churn	0%
Network Size	100
Peer Inter-arrival Time	10
BW_{up}	64
BW_{down}	Unlimited
$\text{Count}_{\text{pieces}}$	10
$\text{Size}_{\text{piece}}$	64
Slots_{up}	4
$\text{Slots}_{\text{down}}$	4
T_{seeding}	0-500

Tab. 6: Experiment Settings - Leaving Seeders

Peers began to download pieces as soon as they joined the network. Data collection began as soon as the first peer joined the network.

5 Results & Analysis

After carrying out the experiments, the results were analyzed.

This section details the results and attempts to reason about each observation.

5.1 Slot Count - Static Download

In general, it is most beneficial if a piece can be disseminated among the peers as soon as possible. This idea is formalized by Equation 9.

When pieces are disseminated more quickly BW_{system} increases more quickly because more peers can contribute as seeders of those pieces. The individual upload bandwidth of each seeding peer, BW_{peer}, accumulates to produce BW_{system}.

The results presented by Figure 1 were observed when $Slots_{up}$ was varied between 1-12 and $Slots_{down}$ remained static at 12.

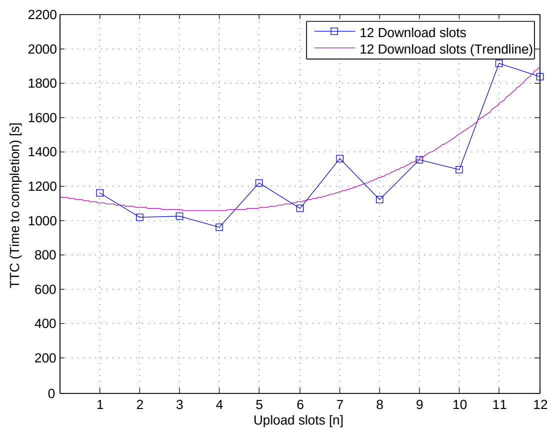

Fig. 1: Time To Completion vs Upload Slots

From these results the following phenomena can be observed:

Low time to completion when $Slots_{up}$ is minimal. When $Slots_{up}$ is minimal $T_{transfer}$ is also minimal, and in general this is desired. However, depending on BW_{up}, $T_{transfer}$ may take such a short time that reducing it further provides negligible benefit with regard to disseminating the pieces. This is because $T_{download}$ will be increasingly dominated by $T_{protocol}$. This is a point of diminishing returns, and results in slower dissemination of pieces as the peer's upload bandwidth would be more efficiently used by adding more, slower, slots.

Lowest time to completion when $Slots_{up}$ is still reasonably low. Optimal performance was observed when $Slots_{up}$ resulted in $T_{transfer}$ that was as low as possible without entering the point of diminishing returns. We reason that this occurred due to the combination of the following:

1. Fast piece dissemination due to low $T_{transfer}$
2. Ability to perform more piece transfers in parallel

Highest time to completion as $Slots_{up}$ increases further. As $Slots_{up}$ increases, $T_{transfer}$ also increases due to the peer's upload bandwidth being shared between all upload slots. As a consequence, other peers must wait longer before they can download a piece and seed it, and BW_{system} increases at a slower rate. It also becomes increasingly difficult for a peer to utilize all of its upload bandwidth as it has more upload slots to keep busy. See Equation 8.

5.2 Slot Count - Static Upload

The results presented by Figure 2 were observed when $Slots_{down}$ was varied between 1-12, but $Slots_{up}$ remained static at 4.

From these results the following phenomena can be observed:

As given by Equation 9, it is most beneficial if a piece can be disseminated among the peers as soon as possible. Because there is no limit on BW_{down} in this model, there is no penalty for increasing the number of download slots, $Slots_{down}$. Increasing $Slots_{down}$ is always beneficial, as

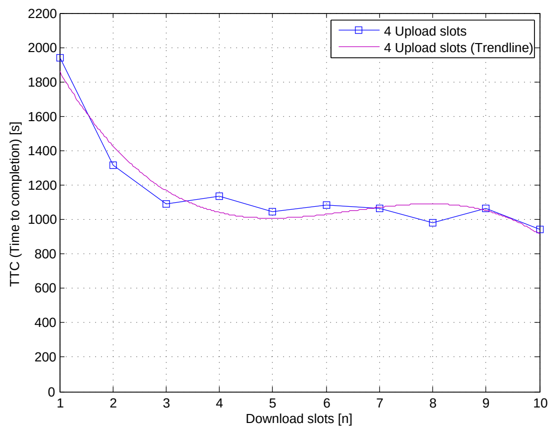

Fig. 2: Time To Completion vs Download Slots

long as $\text{Slots}_{\text{down}} < \text{Count}_{\text{pieces}}$. After this point, adding more download slots has no effect on performance. This is confirmed by the results.

5.3 Network Size

The results presented by Figure 3 were observed when the number of peers in the network was varied between 1-1000.

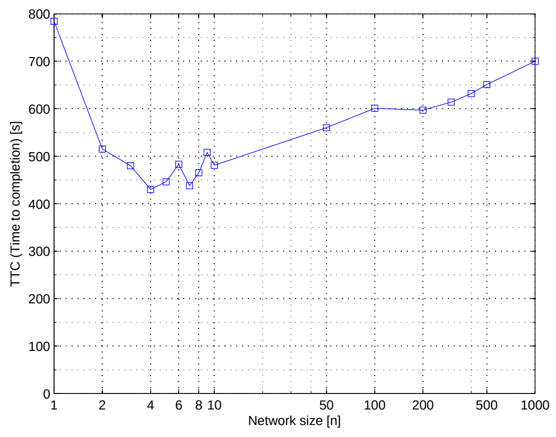

Fig. 3: Time To Completion vs Network Size

From these results the following phenomena can be observed:

With a minimal number of peers the time to completion is highest. This is because peers can only download one piece from the same seeder at any one time, so only a limited amount of work can be done in parallel in this case.

As more peers are added to the network time to completion reduces. Adding more peers allows some work to be performed in parallel.

As number of peers steadily increases time to completion slowly increases. After the initial benefit of performing work in parallel, adding more peers results in more data that must be downloaded by all peers. The amount of data to be downloaded increases, but so does $\text{BW}_{\text{system}}$. The increase in $\text{BW}_{\text{system}}$ partially counteracts the increase in data to be downloaded, which is why the time to completion only increases slowly.

5.4 Upload Bandwidth

The results presented by Figure 4 and Figure 5 were observed when varying BW_{up} between 32-1024.

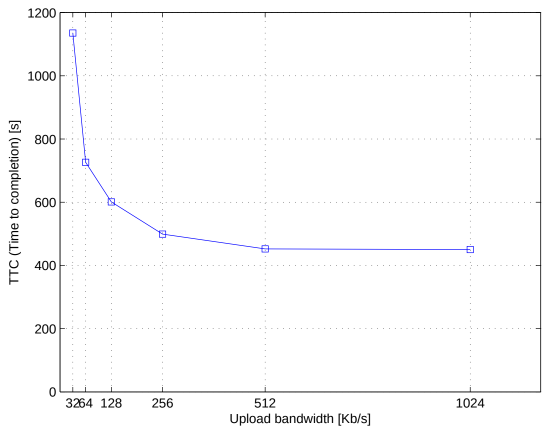

Fig. 4: Time To Completion vs Upload Bandwidth

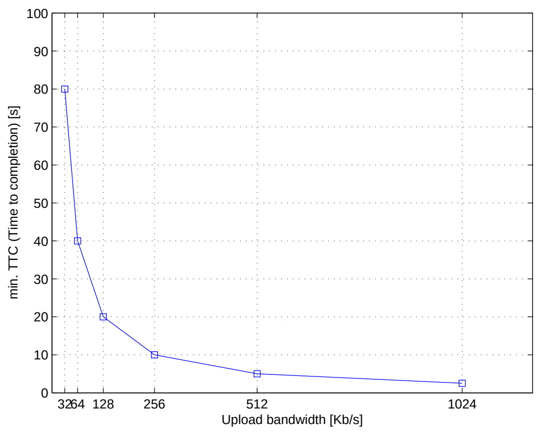

Fig. 5: T_{transfer} vs Upload Bandwidth

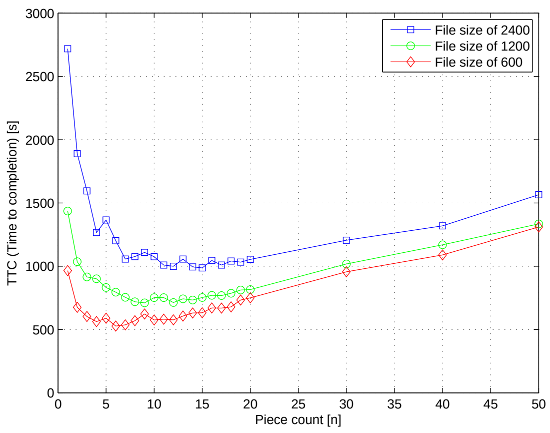

Fig. 6: Time To Completion vs Piece Count

From these results the following phenomena can be observed:

Time to completion improved most when $\mathbf{BW_{up}}$ was raised from 32 to 64. When BW_{up} is 32 T_{transfer} is 80 time units (refer to Equation 4). Obviously as BW_{up} is doubled, T_{transfer} will halve. Therefore, time to completion will improve as BW_{up} is increased.

As $\mathbf{BW_{up}}$ increases further, time to completion improves at a slower rate. The phenomenon of diminishing returns discussed earlier is applicable here also, so as BW_{up} continues to increase T_{transfer} accounts for less and less of the total bandwidth requirements, and T_{protocol} starts to become the limiting factor. At this point, increasing BW_{up} has a negligible effect on the time to completion.

5.5 Piece Count

The results presented by Figure 6 and Figure 7 were observed when varying $\text{Count}_{\text{pieces}}$ between 1-50.

From these results the following phenomena can be observed:

Time to completion is high when $\mathbf{Count_{pieces}}$ is low. When $\text{Count}_{\text{pieces}}$ is low the process of disseminating pieces can not be parallelized very well. T_{transfer} becomes higher so $\text{BW}_{\text{system}}$ increases at a slower rate. In this case, a peer can not begin seeding until it has a large percentage of the file. Moreover, when $\text{Count}_{\text{pieces}}$ is low $\text{Util}_{\text{slots}}$ is likely to also be low (refer to Equation 7).

As $\mathbf{Count_{pieces}}$ increases time to completion begins to decrease. When $\text{Count}_{\text{pieces}}$ increases $\text{Size}_{\text{pieces}}$ decreases. As a result T_{transfer} decreases too and peers can begin seeding a given piece earlier. In other words $\text{BW}_{\text{system}}$ increases at a faster rate. This leads to more work being performed in parallel because the file is broken down into more pieces. Consequently, peers can utilize their upload slots and upload bandwidth more efficiently.

Increasing $\mathbf{Count_{pieces}}$ too much has a negative effect. As $\text{Count}_{\text{pieces}}$ continues to increase $\text{Size}_{\text{piece}}$ becomes increasingly small, and T_{transfer} accounts for less and less of the total bandwidth requirements. Again, we see that T_{protocol} takes over as the

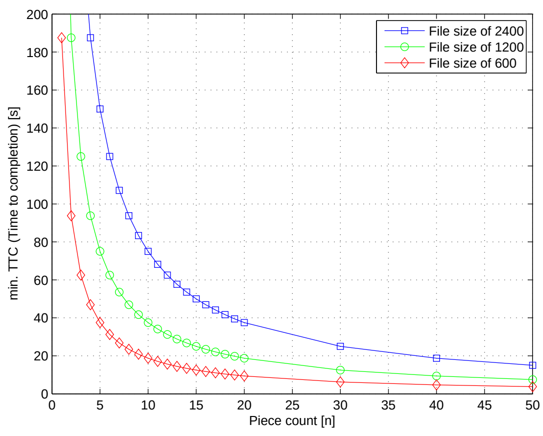

Fig. 7: T_{transfer} vs Piece Count

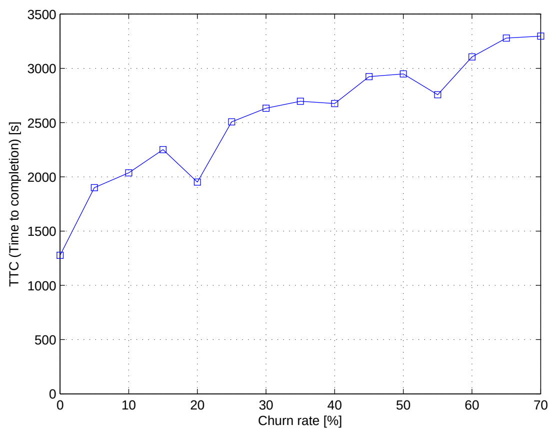

Fig. 8: Time To Completion vs Churn

major overhead. At this point breaking the file into more pieces results in a measurable amount of additional protocol overhead, but has negligible benefits on T_{transfer}. The point of diminishing returns is reached and time to completion increases once again.

5.6 Churn

An important fact that should be introduced here is that, to create churn the simulator must generate additional events that add/remove peers from the network. It takes a small amount of time to create each event and to increase the amount of churn the simulator must create more events. It is natural for an experiment with more churn to run for a longer duration, if for no other reason then simply because more events must be executed.

The results presented by Figure 8 were observed when varying the amount of churn between 0-70%.

From these results the following phenomena can be observed:

As expected, when there is no churn in the network time to completion is lowest. There are fewer events. Peers never fail or leave, so pieces are never lost and the amount of pieces in the system monotonically increases.

Time to completion increases quickly when churn is introduced. Amount of simulator events increases. The amount of pieces in the system is no longer guaranteed to monotonically increase. Furthermore, run time is dependent on which peers fail. If a failing peer is seeding any pieces, those pieces will be lost and $\text{BW}_{\text{system}}$ will also decrease.

Time to completion continues to increase as more churn is added. The amount of simulator events continues to increase. Due to the rise in number of failures, it is also increasingly likely that the number of pieces in the system will not monotonically increase, and the probability that failing peers are also seeding peers becomes higher.

An important observation of the results is that all live peers eventually become seeders. The reason for this is because the initial seeder is reliable (can never fail), all peers are guaranteed to eventually make progress.

5.7 Leaving Seeders

The results presented by Figure 9 were observed when varying the amount of time peers remained in the network after seeding, between 0-500ms.

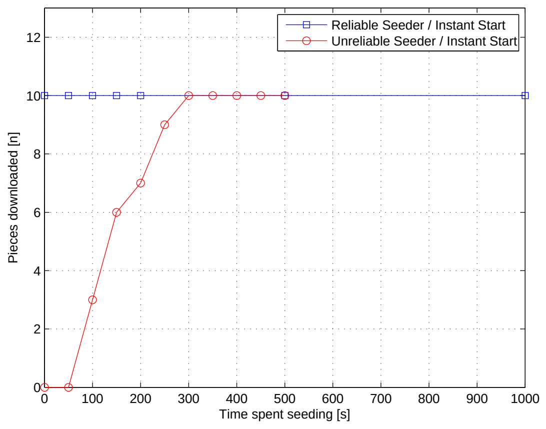

Fig. 9: Available Pieces vs Time Seeding

From these results the following phenomena can be observed:

When peers seed for a minimal amount of time, some peers never become seeders. Seeders leave before every piece can be disseminated to some other peer.

As time spent seeding increases, the remaining peers are able to download more pieces. Some peers still never download the entire file. Seeders leave before every piece can be disseminated to some other peer. However, the longer a seeder stays the more unique pieces will be disseminated. For all peers to eventually become seeders it is important that every piece is sufficiently replicated in the network.

Once time spent seeding surpasses a threshold (300 time units), all peers become seeders. All peers are able to eventually acquire all pieces and become seeders. Simply put, seeders must stay in the network long enough to ensure every piece is replicated sufficiently such that every piece disseminates throughout the network to enough peers to guaranty that when seeders leave, no pieces become unattainable.

6 Discussion

Despite Basic Torrent's relative simplicity, while carrying out this project various interesting phenomena were discovered. These were documented, reasoned about, and used to incrementally build theories and understanding of the protocol.

Most findings were significantly more complex than they initially appeared, and many phenomena proved to be multi-dimensional and difficult or impossible to explain with relation to only one variable.

The goal of this report was to introduce a simplified version of BitTorrent. The protocol was not intended for practical use, but rather for use in analyzing phenomena that arise in peer-to-peer systems like BitTorrent. To that end, creating and experimenting with Basic Torrent has been worthwhile.

■