| **Fundamentals of Computer Networking** | **Project 3: Selfish BitTorrent Client** |
|---|---|
| **CS4700/CS5700 Fall 2009** | **7 November 2009** |

*This project is due at 11:59pm on December 1st, 2009.*

## 1 Description

You will modify the trading policy of a BitTorrent client in order to improve the performance. Unlike previous projects, the majority of your grade in this project will come from performance. Thus, we will give you working code, and you will be responsible for improving the performance. Part of your grade will come from how your code performs when run against the solutions of your the course staff, and part will come from how your code performs against your peers.

In this project, you will have to do background reading on BitTorrent. Since we are providing you with a working implementation, you are *expected* to research the BitTorrent trading scheme, improvements to it, and possible attacks. *Simply turning in the code that we provide (or a slightly modified implementation) will result in a very low grade.* You should write selfish code, the goal of which is to improve your download speed.

## 2 Requirements

We will provide you with the code you will need to make the project functional. Thus, all you must do is increase the performance of your client. In particular, our code must

- Outperform the client provided by the TAs, in terms of download speed.
- Outperform the client provided by the TAs, in terms of the upload to download ratio.
- Outperform the the solutions of your peers.

As before, you should not assume that other peers will follow the protocol. Since these are the solutions of your peers, they may be malicious, they may upload bad data, they may lie to you, and the certainly will be greedy and selfish. Your solution should handle these errors gracefully, recover, download the file correctly, and *not* crash.

## 3 Your program

For this project, you will submit a complete BitTorrent program. It is recommended that you use the starter code provided by the TAs; however, you may implement a BitTorrent client from scratch if you so wish (and are crazy). If you do so, you are welcome to use a language of your choice. However, all of the code must be your own; you are not allowed to use code from other BitTorrent implementations.

The command line syntax for your sending is given below. The program takes either a filename or a url pointing to a tracker. The syntax for launching your sending program is therefore:

```
Usage:  snark [--debug [level]] [--port <port>] [--show-peers] [--share] (<url>|<file>)
```

with the following details:

`--debug` (Optional) Shows some extra info and stacktraces. Argument `level` is how much debug details to show (defaults to SEVERE, other options INFO and ALL).

`--port` (Optional) The port to listen on for incoming connections (if not given defaults to first free port between 6881-6889).

`--show-peers` (Optional) If enabled, periodically prints peer information.

`--share` (Optional) Start torrent tracker with the provided file..

`file` (Required if `--share`) The file to share.

`url` (Required if no `--share`) URL pointing to .torrent metainfo file to download/share..

Luckily, the code we provide to you already does all of the argument parsing and supports these options. You should develop your BitTorrent program on the CCIS Linux machines, as these have the necessary compiler and library support. You are welcome to use your own Linux/OS X/Windows machines, but you are responsible for getting your code working, and your code *must* work when graded on the CCIS Linux machines. If you do not have a CCIS account, you should get one ASAP in order to complete the project.

## 4 Testing your code

In order for you to test your code, the code we provide can be used to set up a private torrent for you to test with. In order to start a torrent, you will execute

```
snark --share <file>
```

In the output, you will see the line

```
Torrent available on <url>
```

In order to connect clients to this torrent, you simply run

```
snark <url>
```

Since BitTorrent is a scalable protocol, you can run any number of clients.

## 5 Submitting your project

You should submit your project by emailing the instructor at `amislove@ccs.neu.edu`. You should submit a `.tar.gz` file as an attachment to the email, which contains your code. Please have the subject line for the email be `[CS4700] [Project 3 Submission]` followed by your last name and the last name of your other group member(s).

This `.tar.gz` file should unpack to have the following structure

```
project3-{lastname1}-{lastname2}/
                                    README
                                    file1.{c, java, pl, ...}
                                    ....
```

If using a compiled language (e.g., C, Java, C++), you should include a `Makefile` or `build.xml` that will compile your code. The `README` should have a brief description and explanation of the approach you use, a list of properties/features of your design that you think is good, as well as examples of how to run your code.

## 6 Undergraduate student version

The undergraduate version of this project can make the following simplifying assumptions: You may assume that other clients will *not* say they have pieces that they don't, and you may assume that other clients will *not* upload junk data to you (i.e., all file hashes will match).

Students enrolled in the graduate version of this project may not make either of these assumptions.

## 7 Advice

A few pointers that you may find useful while working on this project:

- Start by familiarizing yourself with the operation of the BitTorrent protocol. This will be covered in class, but you are responsible for learning this yourself as well. Then familiarize yourself with the code provided. You will need to modify this code, so you should take some time to familiarize yourself with it.
- Check the Blackboard forum for question and clarifications. You should post project-specific questions there first, before emailing the course staff.
- Finally, *get started early* and come to the TA lab hours - these are held from 5:00pm - 7:00pm on Wednesdays in the lab at 212 West Village H. You are welcome to come to the lab and work, and ask the TA and instructor any questions you may have.

## 8 BitTorrent resources

There is a large amount of BitTorrent literature out on the Web. A few relevant documents and papers:

- `http://wiki.theory.org/BitTorrentSpecification` Documentation of the low-level BitTorrent protocol.
- `http://www.bittorrent.org/bittorrentecon.pdf` Description on BitTorrent's original incentive mechanism.
- `http://www.dcg.ethz.ch/publications/hotnets06.pdf` Description of BitThief, a client which tries to never upload any data.
- `http://www.cs.washington.edu/homes/arvind/papers/bittyrant.pdf` Paper describing the BitTyrant client.
- `http://www.cs.umd.edu/~dml/papers/bittorrent_sigcomm08.pdf` Paper describing the PropShare client.