BitTorrent

Internet Technologies and Applications

Aims and Contents

- Aims
  - Show benefits of P2P for file downloads
  - Explain concepts of BitTorrent as a P2P file sharing application
  - Show how BitTorrent protocol works
- Contents
  - Example: File Transfer
    - P2P versus Client/Server
  - P2P File Sharing Systems
  - BitTorrent

Example: File Transfer

Client/Server

Peer-to-peer (P2P)

File Transfer using Client/Server

- A server has a 1000Mbit file for download
- Every user (including server) has a 1Mb/s full duplex connection in a network
- Initially, 5 users download the file from the server
  - Each user can receive at a maximum of 1Mb/s
  - Server can send at 1Mb/s, or 0.2Mb/s to each user
  - 1000Mbit @ 0.2Mb/s: 5,000 seconds
- Some time later, 100 users download the file from the server
  - Each user can receive at a maximum of 1Mb/s
  - Server can send at 1Mb/s, or 0.01Mb/s to each user
  - 1000Mbit @ 0.01Mb/s: 100,000 seconds

File Transfer using P2P

- The 5 “seed” users all download from the server (same as client/server approach)
  - 5,000 seconds
- But some time later, the 100 users download from the 5 seed users (not from the server)
  - Each user can receive at a maximum of 1Mb/s
  - Seed users can send at 1Mb/s, or 0.05Mb/s to each user
  - 1000Mbit @ 0.05Mb/s: 20,000 seconds

- Although the initial step is the same as client/server, subsequent users can download from the seed users, rather than single server
  - 5 servers to download from instead of 1

Factors affecting Performance

- Number of:
  - Servers
  - Seed users
  - Downloading users
- Connection speeds of each device:
  - Uplink
  - Downlink
- File structure
  - If the file is divided into pieces, then a user can start uploading pieces before they have downloaded the entire file

Technical Tradeoffs of P2P File Sharing

- Advantages of P2P file sharing:
  - Not always dependant on original server
    - Redundancy: content is available if server fails
    - Performance: server does not have to handle load from all users; server link does not have to support access from all users at once
- Disadvantages of P2P file sharing:
  - Relies on participation of other users
    - As the number of seeding users decreases, the performance decreases
    - Download performance varies over time
  - No central control of access to content
    - Difficult to control who accesses the content
    - Difficult to find the content (no central index of files)

- There are many other issues regarding:
  - Security, privacy, accounting, legality, …

P2P File Sharing Systems

P2P File Sharing Systems

- Protocols
  - Communication protocols for locating other users and transferring the data
  - Examples: BitTorrent, Gnutella, Naptser, FastTrack, eDonkey, …
- Software
  - Client programs that implement the protocols and provide a user-interface
  - Examples:
    - BitTorrent: BitTorrent, BitComet, uTorrent, Transmission, Azeurus Vuze, …
    - Gnutella: Limewire, Morpheus, Shareaza, gtk-gnutella, …
    - Napster: Napster, WinMX, …
    - FastTrack: Kazaa, Grokster, iMesh, …
    - eDonkey: eDonkey, eMule, …
  - Some clients also support multiple protocols
- Networks
  - Collections of users (including servers) using the same protocol

BitTorrent

Terminology and Concepts

- A file (or set of files) is referred to as a *torrent*
  - A *.torrent* file is a specific file that describes the torrent

- Users accessing a torrent are called *peers*
  - Downloading and/or uploading
- A peer that has downloaded the entire torrent is a *seed*
  - That user is *seeding* the torrent
- A peer that has not yet downloaded the entire torrent is a *downloader* or a *leech*
  - Normally a leecher has downloaded more than they have uploaded
- A *swarm* is the set of users connected to a torrent
  - All the peers, including seeds

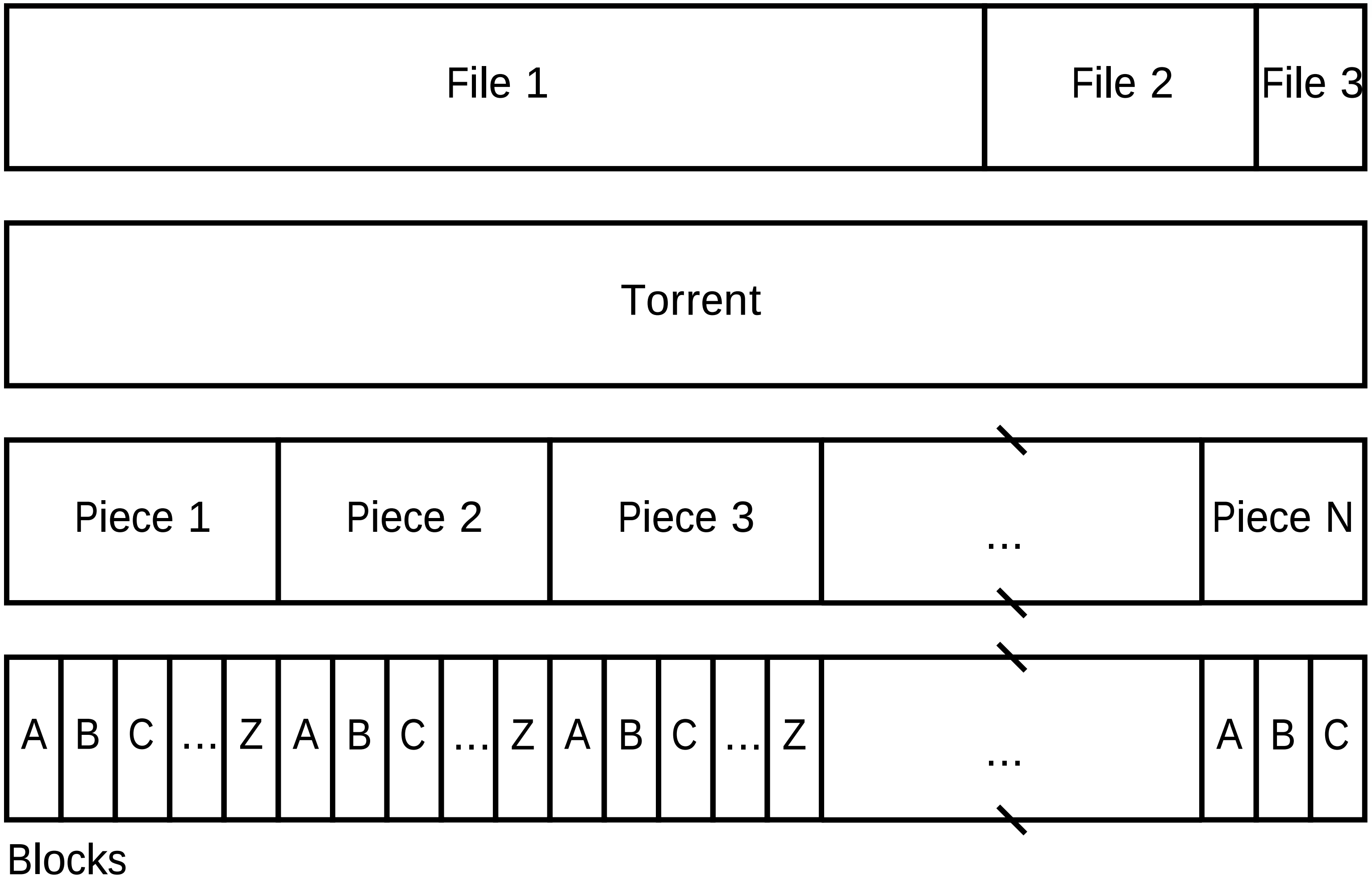
Terminology and Concepts

- Files are divided into *pieces*
  - When a piece is downloaded by a peer, it is then made available for upload
- Pieces are divided into *blocks*

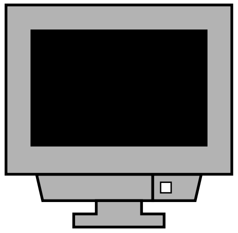
- A *tracker* is a server that manages the list of peers in a swarm
  - Keeps a list of peer IP addresses and port numbers

- An *indexer* is a server that manages a list of torrents
  - Normally a website that allows search for .torrent files by name and other descriptive data
  - A .torrent file also lists the address of the tracker

Files, Torrent, Pieces and Blocks

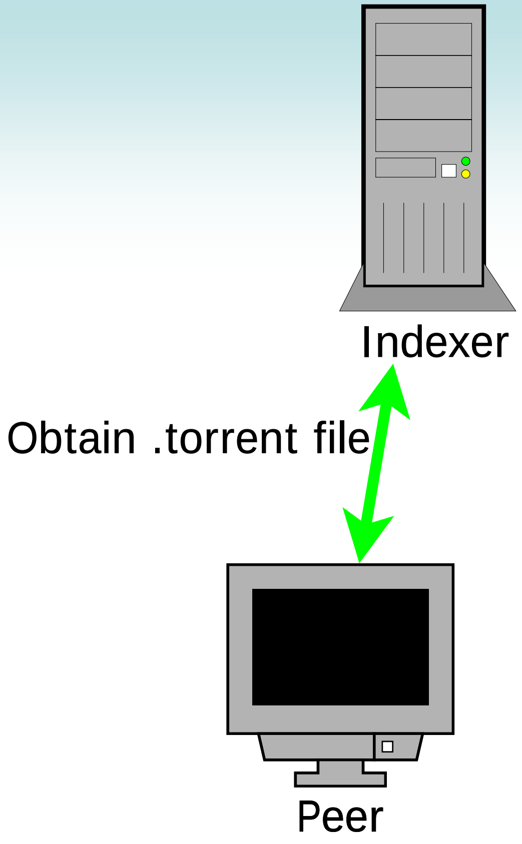


Steps using BitTorrent



A peer is a BitTorrent client software running on a computer. It can be identified by the IP address of the computer and the applications port number.

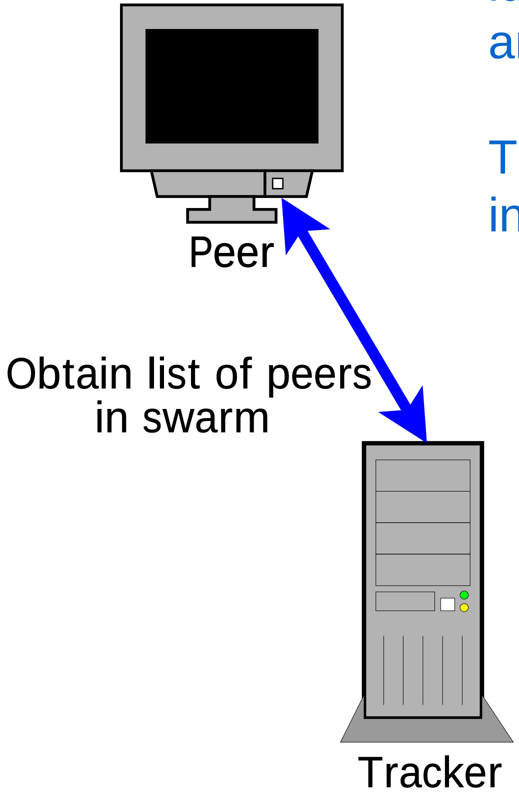
Initially the peer does not know any other peers.



To start downloading a file, the corresponding .torrent file is needed.

The peer contacts an indexer, which has a list of known torrents.

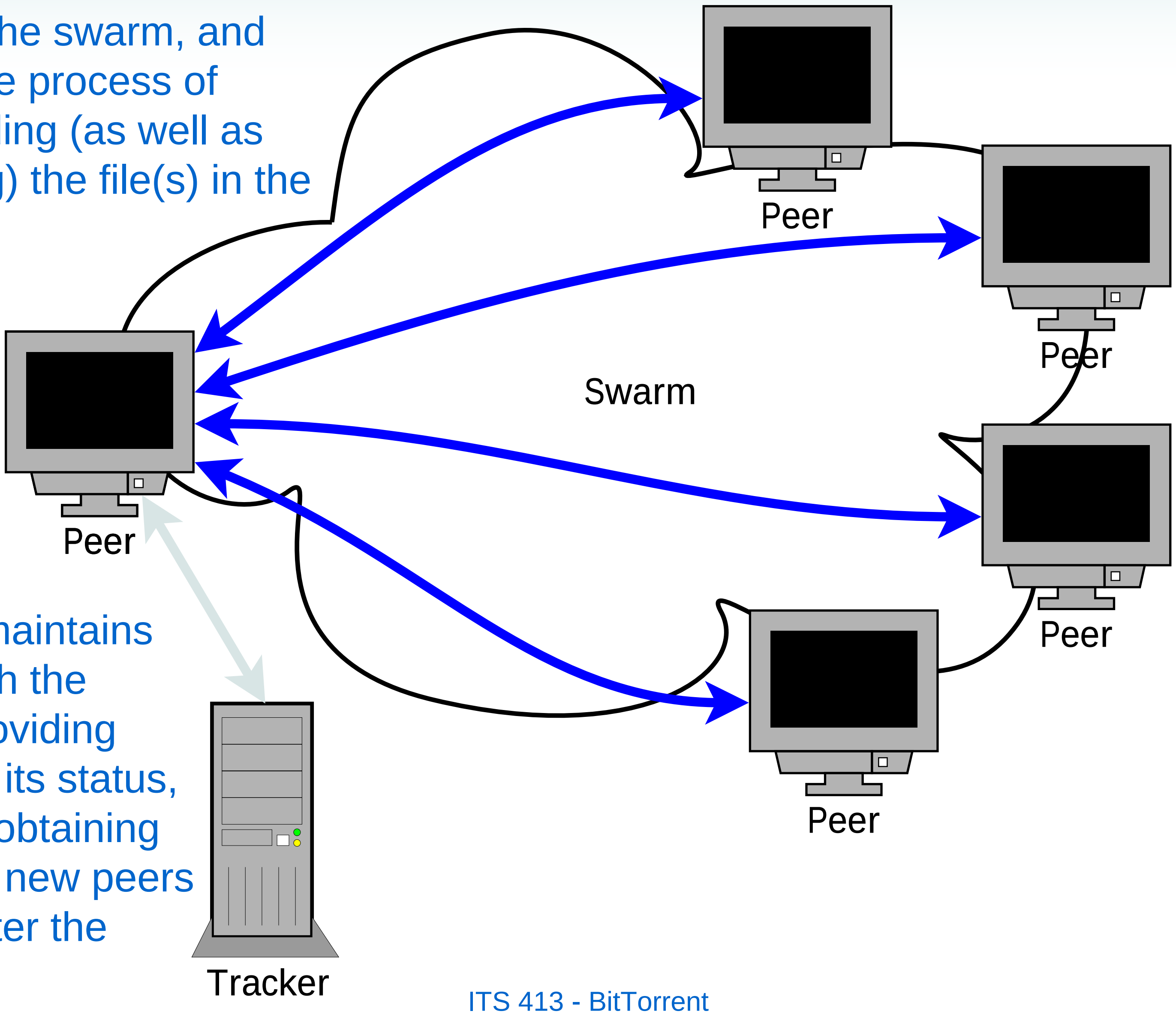
The process of obtaining the .torrent file is *not* part of the BitTorrent protocol. Normally it is performed using a web browser contacting a web server (the index). The .torrent file may be downloaded using HTTP, and then loaded in the BitTorrent client software.



The .torrent file describes the file(s), as well as identifies the tracker for the torrent (by an IP address and port number)

The peer contacts the tracker to obtain the list of peers in the swarm for the torrent.

The peer contacts other peers in the swarm, and begins the process of downloading (as well as uploading) the file(s) in the torrent.



The peer maintains contact with the tracker, providing updates of its status, as well as obtaining address of new peers as they enter the swarm.

.torrent File

- The .torrent file describes the contents of the files as well as the tracker that manages the swarm
- A text file with various (encoded) fields:
  - info: describes the file(s) of the torrent. Includes:
    - File names, file length, length of pieces in torrent, and for each piece the SHA1 hash of the piece
  - announce: URL of the tracker
    - An optional list of backup trackers may also be included
  - creation date: when the torrent was created
  - created by: name of software that created torrent
  - comment: textual description of the contents

Example .torrent File

```
d8:announce39:http://torrent.ubuntu.com:6969/announce
7:comment29:Ubuntu CD releases.ubuntu.com
13:creation datei1225365524e
4:infod
          6:lengthi732766208e
          4:name28:ubuntu-8.10-desktop-i386.iso
          12:piece lengthi524288e
          6:pieces27960:
          þý“!├ .¶ ÈTO ™┼ ▯ îðÿB2ÅÉÒ▯óý¨úþ9tz>©‘U…
```

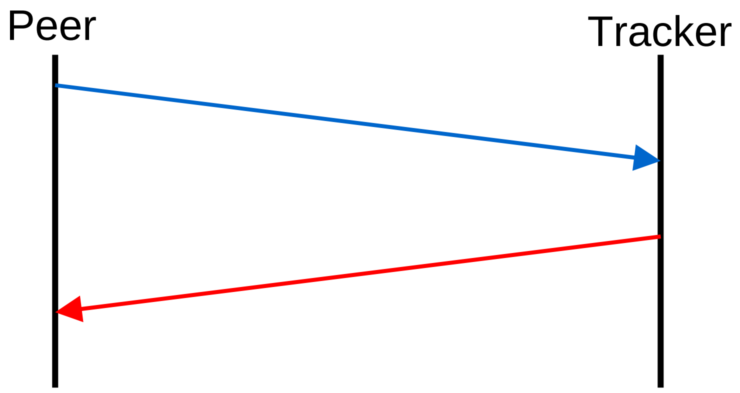
Hash values of all 27960 pieces

Peer to Tracker Communications

- The Tracker implements a HTTP web server
- Peers communicate with the tracker using HTTP
  - Peer sends HTTP GET requests. URL contains information about the peers activities in the torrent:
    - ID of peer
    - Port number the client is listening on
    - Bytes uploaded/downloaded so far; Bytes remaining to download
    - Status of download (started, stopped, completed)
    - …
  - Tracker sends HTTP responses, containing plain text:
    - Interval that client should send requests to tracker
    - Number of peers completed (seeds) and incomplete (leechers) in the swarm
    - A list of peers (peer ID, IP address, port number)
    - Warning/error messages
    - …

Example Peer/Tracker Exchange

```
GET /announce?info_hash=3%82%0d%b6%dd%5eY%28%d2%3b%c8%11%bb%ac%2fJ%e9L%b8%82
&peer_id=-UT1810-_1%02A%ba%d4%9a%0c%e8JBB
&port=11541          &uploaded=0         &downloaded=0       &left=732766208
&corrupt=0           &key=82D434AB       &event=started      &numwant=200
&compact=1           &no_peer_id=1       &ipv6=192.168.1.2
HTTP/1.1
Host: torrent.ubuntu.com:6969
User-Agent: uTorrent/1810
Accept-Encoding:gzip
```



```
HTTP/1.0 200 OK
Content-Length: 362
Content-Type: text/plain
Pragma: no-cache

d8:completei1328e10:incompletei97e8:intervali1800e5:peers300:<.#...Z .....
```

Peer Exchange Protocol

- A TCP connection is created from a peer (client) to a remote peer and they exchange messages to inform of current status and download blocks
  - At the start, the client sends a Handshake message to remote peer
    - Informs remote peer of PeerID
  - This may be followed by a Bitfield message, where the remote peer indicates the set of pieces it has available for download

- Once the Handshake is sent, a peer (client or remote peer) may send:
  - Request message to request a specific block
  - Piece message which includes a block
    - Note: although called "Piece" message, it is sending a block
  - Have message to indicate a piece a peer has
  - Cancel message to cancel a previous Request for a block
  - Keepalive message to keep the TCP connection open
    - In the case nothing has been sent between the peers recently

Peer Exchange Protocol

- BitTorrent aims to implement a ‘tit-for-tat’ algorithm for data transfer
  - For every one piece the client downloads, it uploads one piece
  - Aim to maintain equal amount of upload/download
  - BitTorrent applies this while a peer is downloading a torrent
    - Once the peer has completed (i.e. becomes a Seed), there is no way for the protocol to enforce the peer to keep uploading
- Peers uses two concepts:
  - Choking: when a client is choked by a remote peer, then no requests will be handled by the remote peer
  - Interested: when a remote peer is interested in the client, then it would like to receive a piece that the client has
  - Client maintains status of remote peer (values may be true or false):
    - am_choking: the client is choking the remote peer
    - am_interested: the client is interested in the remote peer
    - peer_choking: the remote peer is choking the client
    - peer_interested: the remote peer is interested in the client

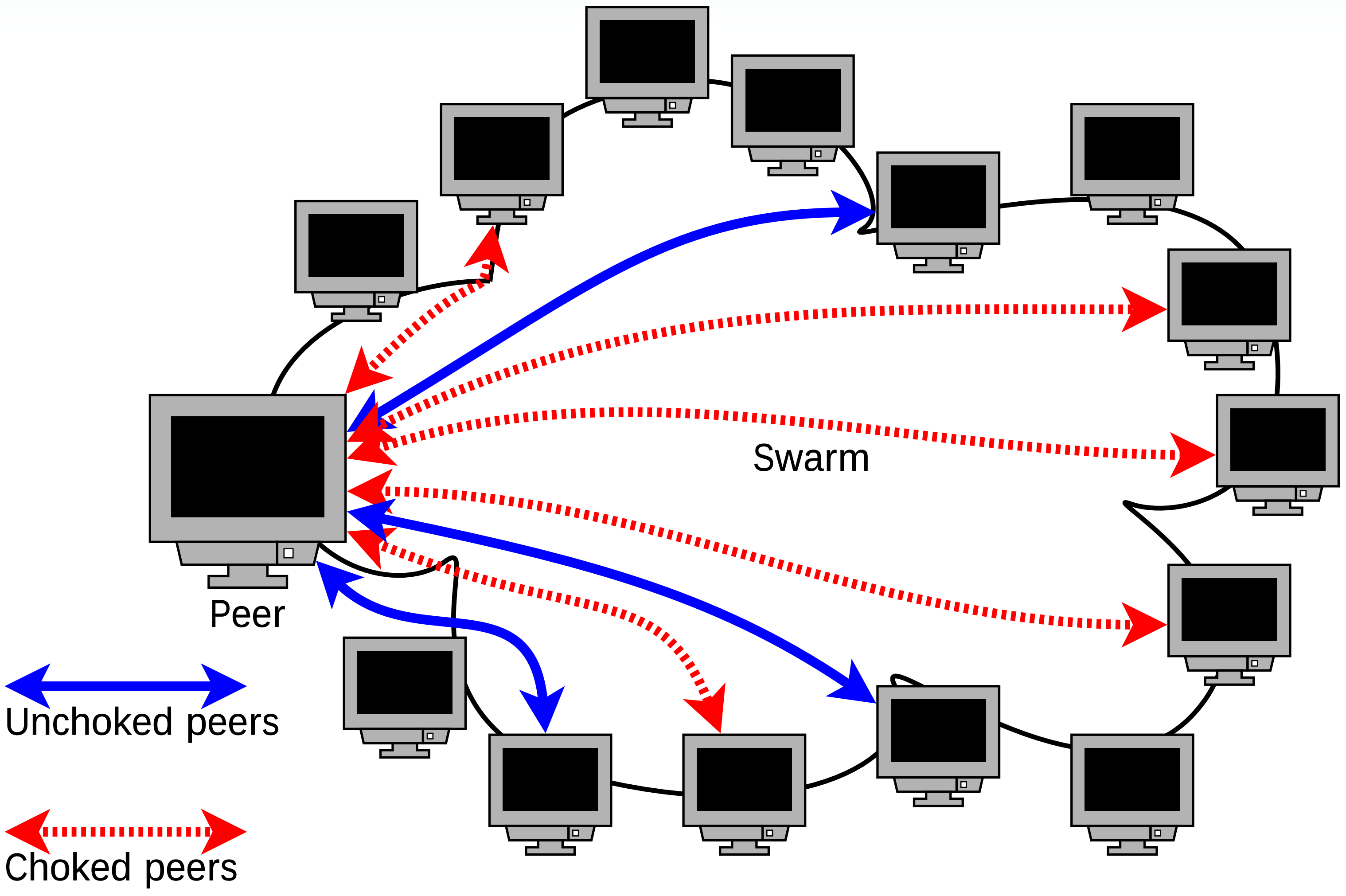
Peer Exchange Protocol

- The peers may send following messages to change state:
  - Choked message
  - Unchoked message
  - Interested message
  - Uninterested message

- Each peer implements a choking algorithm to determine when to choke/unchoke peers
  - Need to consider many factors, including the download rate from different peers

- For a torrent, a client may have connections with many (10’s of peers), but generally only several (4 to 10) will be unchoked at anyone time

Peers in a Swarm



Torrent, Pieces and Blocks

- Although there is no size limits, using BitTorrent is usually most efficient for large sizes, e.g. MBytes to GBytes
- Pieces are normally 256KB, 512KB or 1MB
- Blocks are normally 16KB

- Which pieces does a client download first?
  - Random is one option
  - Rarest-first is another (more common) option
    - A client peer keeps track of the pieces all other peers in the swarm have
      - Original Bitfield message, and regular Have messages
    - Client peer downloads the pieces which are least common, so that it will become more common
- *Availability* of a torrent: number of copies of torrent in swarm
  - 1 = all pieces are available at peers in the swarm; 0.5 = only 50% of pieces are available (you will not be able to download entire file); 20 = 20 copies of the torrent are available

Performance Issues

- How many peers to connect to?
  - About 10 to 30 for each torrent
- How many peers to download from (unchoked)?
- What are the optimal piece and block sizes?
- What is the best choking algorithm?
  - When do you try a different peer (in the hope that they will offer faster download)?
- How to ensure users upload content?
- How to maintain fairness with other applications?
  - Remember, TCP aims to provide fairness amongst connections. If BitTorrent has 100 connections and web browser has only 10 connections, there may be extreme unfairness
- …