#ASLI ENGINEERING

# Piece Selection Algorithm

SWIPE

BY

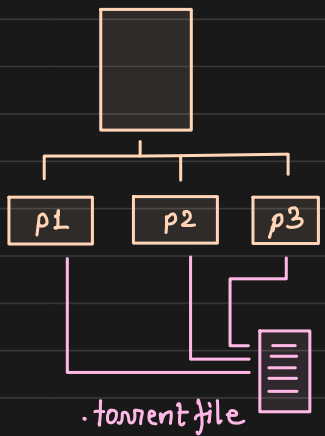
ARPIT BHAYANI

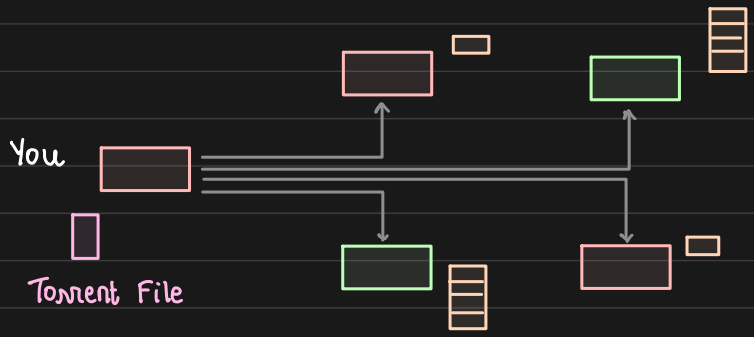
# The Piece Selection Algorithm

The original file that is to be shared in the network is split into equal sized pieces

256KB to 1MB

A piece for a torrent is identified by 20B SHA1 hash and to download the file, we have to ask for pieces from the peers participating in the network




The performance of the protocol depends on the order of pieces but why?

because, we don't want to be in a situation where every peer has the same of pieces and all are stalled on seeder for the next

↓

has the entire file

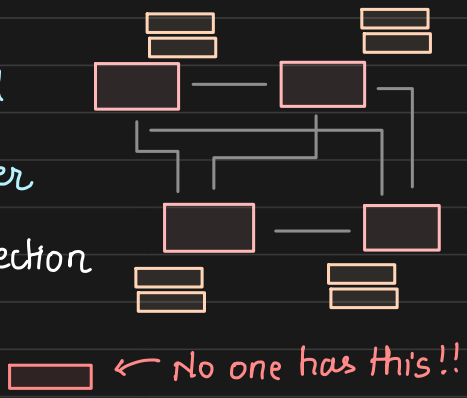
and what if the seeder leaves the network?

The download of the entire network would stop !!

Hence,

1. to maximize the download speed
2. minimize the dependency on seeder

we have to diversify the piece selection

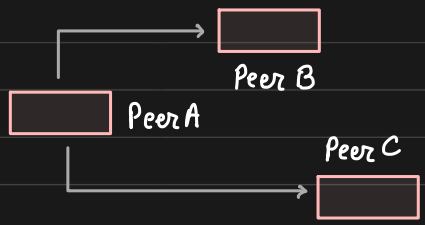



## Rarest-First Piece Selection Algorithm

Core idea: Prioritize downloading the piece that is rare

Say, peer A wants to download the file

So, it would start downloading pieces from its peer set { B, C }



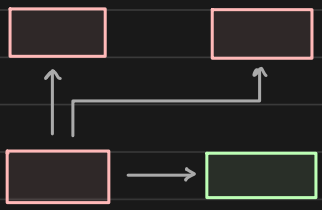

A would send req to B and C checking the pieces they have and B and C would respond

By keeping a track of pieces available with peers

A would prioritize requesting the rarer piece.

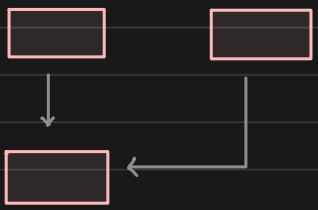
## Advantages of Rarest First

1. Spreading the seed: It ensures only 'new' pieces are downloaded from the seeder, while peers trade other pieces among themselves

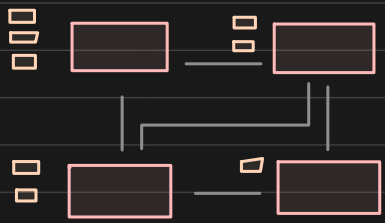


2. Increased download speed: More peers have the pieces, faster download we'll get

3. Enabling upload: If you have a rare piece, other peers would want it from you, and thus due to Choke algorithm, you'll get unchoked often.



4. Prevent rarer piece missing:

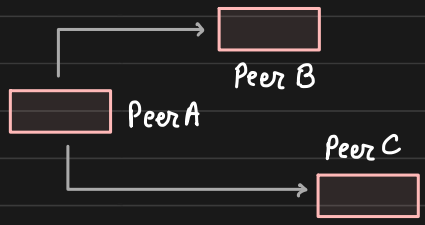


By prioritizing rarest piece first we ensure that it does not go missing

↳ seeder left and now no peer has it

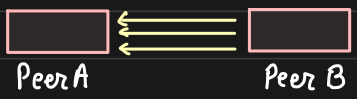
## How does a peer compute rarest piece?

Every peer has a peer set that it got from the tracker.

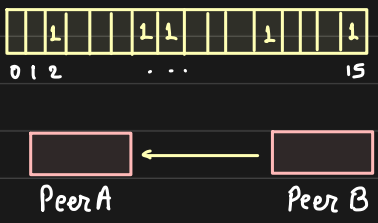



There are two ways by which a peer can tell about the pieces they have

1. 'Have' message : a series of Have messages can be sent by peers (B and C), one for each piece




2. 'Bitfield' message : at the start of the connection, the peer can send a Bitfield message to A that has bits marked as 1 for the pieces the peer holds



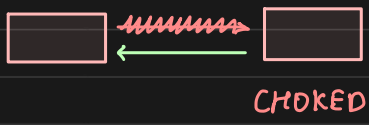

Thus, the peer A can compute the rarest piece and fetch it on priority from the network

Rarest piece is computed everytime we receive 'Have' message from the peers.

## Random First Policy

When a peer joins a network, it would need the piece as soon as possible because of reciprocation

Peer will allow you to download, only when you have something to upload



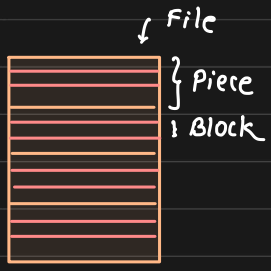

Because rare pieces are slower to download

if we have downloaded < 4 pieces, we choose the next piece to request at random. This would ensure we quickly have 4 pieces and start actively contributing to the network

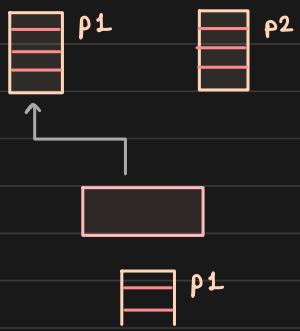
* Once the first 4 pieces are downloaded, the peer switches from random first to rarest first.

## Strict Priority Policy

A file is split into pieces and pieces are split into blocks. In each transfer, a block of a piece is transferred.



Given that a peer cannot contribute back to the network untill it has one complete piece




The peer will always prioritize downlooding of all blocks of a piece before downloading a block of other piece

This ensures we prioritize completion of a piece before moving to next.

## End Game Mode

This mode starts at the end when a peer has requested for all the blocks it needs and the request is in transit i.e. download is about to finish and peer is just waiting for responses

Request Made waiting for response

very heavy on n/w but happens only at the end.

To complete the download quickly, the peer sends request to ALL the peers in the peerset and everytime it receives a block, it broadcasts CANCEL.