Lesson 6:
GAME THEORY PRINCIPLES
BITTORRENT

15/3/2019
OUTLINE OF THE LESSON

• Game theory: some basic concepts
  • applied in the Bittorrent chocking algorithm

• Content Distribution Networks

• Bittorrent: general architectures
  • Torrent
  • Tracker
  • Peer to peer protocol

• Peer to peer protocol
  • Choking algorithm
  • Piece selection algorithms
GAME THEORY

• an area of mathematics which studies problems arising where the players of a game
  • have to choose between different strategies
  • make choices to maximize their rewards
  • have a partial knowledge of the behaviour of other players
  • their behaviour is rational

• applied in economy, biology, psychology, sociology, philosophy, and computer science

• in our case:
  • players: peers
  • strategies: choice of the peers to interact with
  • goal of the strategy
    • BitTorrent: download the content as fast as possible
    • BitCoin: model mining strategies (different from the standard strategy)
Example: odd or even
• two players, 1 e 2
• they choose a number at the same time
  • if the sum is even, player 1 wins, otherwise player 2
• pay off: in game theory, is the reward
Strategic Form: tabular representation of the game
- rows and columns are paired with the possible game strategies
- at the cross of row R and column C the pay-off of the two players if they choose the strategies corresponding to that row and column
- rational behaviour: player chooses a strategy to maximize his/her pay-off
- peers are like players: they “play a game” where the reward is the amount of information (Bittorrent) or bitcoin (Bitcoin) they receive

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>P</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>1,0</td>
<td>0,1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0,1</td>
<td>1,0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
GAME THEORY

- **dominant strategy**: best choice regardless what other player do
- given two strategies ST1, ST2 for a player, if, whichever the strategic choice of other players be, the pay-off obtained by choosing ST1 is greater of that obtained by choosing ST2, then

  **ST1 strictly dominates ST2**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2,0</td>
<td>4,1</td>
<td>5,2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>3,1</td>
<td>3,2</td>
<td>2,1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1,4</td>
<td>2,2</td>
<td>0,3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Strategy A strictly dominates C for Player 1
EXAM OR PROJECT?

students needs to decide whether to study for an exam or preparing a project

• both not allowed

• project prepared jointly with a partner

• a student obtains two different votes, one for the examination, one for the presentation of the project

• knows a priori the final vote obtained if it studies for the examination and do not arrange the presentation of the project and the other way round.
EXAM OR PROJECT?

- **Exam** expected grade (payoff)
  - 92 if you study
  - 80 if you do not study

- **Project** expected grade (depends from the decision of the other student)
  - 100 if both of you work on the project
  - 92 if only one of you work on it
  - 84 if no one of the them prepare it

- Rational behaviour: each student is interested to obtain the best grade
- each of you have to decide independently, knowing that the other one will also make a decision, but no possibility to communicate
EXAM OR PROJECT?

- if both of you prepare the project, each of you obtain:
  - the maximum for the project: 100
  - the minimum for the exam: 80
  - average grade: 90
- if both of you prepare the exam, each of you obtain:
  - the maximum for the exam: 92
  - the minimum for the project: 84
  - average grade: 88
- if a student prepares the exam and the other the project:
  - who has prepared the project obtains:
    - 92 for the project and 80 for the examination
    - average grade 86.
  - who has prepared the exam obtains:
    - 92 for the exam and 92 for the project (benefits from the other)
    - average grade 92.
The payoff matrix can be used to show the average grade of students for all possible scenarios.
EXAM OR PROJECT?

- “Study for exam” is a **strictly dominant strategy** for both players, meaning each will get an average grade of 88
- the dominant strategy tells to not collaborate
- yet, there is an outcome that is better for both
  - both work on project and obtain an average grade of 90 that cannot be achieved by rational players
  - better situation for the individual is worst situation for the group

![Game theory payoff matrix]

- **You**
  - **Project**
    - 90, 90
  - **Exam**
    - 92, 86

- **Your partner**
  - **Project**
    - 90, 90
  - **Exam**
    - 86, 92

Laura Ricci
Game theory principles, BitTorrent
how many players
  • for the moment, only two players

how many encounters? For the moment, only one-shot game
  • in general dynamic, iterated games

what do players know?
  • everything about the structure of the game
    • structure of the game, strategies, payoff
  • nothing about the decisions of the other player

rational player: each player tries to maximize his/her own payoff, given her belief about the strategies used by the other players
THE PRISONER DILEMMA

- A famous example of game theory
- Two robbery A e B are captured from the police
  - they are interrogated by the police in separate rooms and they cannot communicate
  - they can:
    - resist to the questioning and proclaim himself/herself innocent
    - betray the other one and confess the crime
- According to their behaviour, the police will behave in the following way:
  - if both of them do not confess, remaining silent (cooperate) they are imprisoned for 1 years for minor crimes.
  - if A confess (betrays) and B remains silent, the betrayer (A) is set free and the other one (B) is imprisoned for 20 years. (and viceversa).
  - if both of them confess, they are both imprisoned for 5 years.
- A e B ask: Which strategy brings to the lower punishment and is so better?
THE PRISONER DILEMMA

<table>
<thead>
<tr>
<th>Prisoner A</th>
<th>Prisoner B</th>
</tr>
</thead>
<tbody>
<tr>
<td>confess</td>
<td>remain silent</td>
</tr>
<tr>
<td>5 years</td>
<td>0 year</td>
</tr>
<tr>
<td>20 years</td>
<td>1 year</td>
</tr>
</tbody>
</table>
THE STRATEGIC FORM OF THE PRISONER DILEMMA

<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>19,19</td>
<td>0,20</td>
</tr>
<tr>
<td>D</td>
<td>20,0</td>
<td>15,15</td>
</tr>
</tbody>
</table>

- **D** = deny, **C** = confess
- **pay-off**: number of spared years
- for both players **D** dominates **C**
  - denial leads to higher individual pay-off – **selfishness**
  - confession leads to higher global pay-off - **cooperation**
- dominant strategy is betraying, even if a larger number of years could be spared by agreeing upon a common decision

- **PD** captures the conflict between individual rationality (selfishness) and common good (cooperation)
• **Dilemma games** games where no matter what choice a player makes, there exists the possibility that he/she feels “regret”, given the possibility, he/she would have acted differently

![Dilemma Game Table]

- generic two-player dilemma game:
  - $R$: reward for mutual cooperation
  - $S$: sucker’s payoff
  - $T$: temptation to defect
  - $P$: punishment for mutual defection
DILEMMAS

Condition for the dilemma

\[ T > R > P > S \]

with four different pay-offs \( (R, S, T, P) \),
- there are \( 4! = 24 \) possible orderings
- most of them correspond to games with no dilemma
- for instance \( R > S > T > P \) always cooperate, never regret
ITERATED PRISONER DILEMMA (IPD)

• what does it happen if the game is iterated? Let us suppose to maximize the sum of the pay-offs
  • cooperation may be possible, since it is possible to decide the behaviour on the basis of the past history

• consider a sequence of rounds, at each round play the prisoner dilemma
  • no a-priori knowledge of the number of game rounds (it is not possible to know when the game ends)
  • players accumulate the pay-off corresponding to the pay-off matrix of the prisoner dilemma
  • winner is the player which has obtained the maximum pay-off
  • each player is able to remind the behaviour of the adversary in the last game session
IPD: OPPONENT WITH A RANDOM BEHAVIOUR

DIFFERENT STRATEGIES AGAINST OPPONENT

- Always defecting is still the better strategy
- Defection has no consequence on the adversary and is better payed
**IPD: OPPONENT IS A “GRUDGER”**

- Grudger: when the opponent defects, switch to defect and defect for ever
IPD: OPPONENT IS A "GRUDGER"

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRUDGER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>24</td>
</tr>
</tbody>
</table>

- Always defecting is no more the better strategy: now is better to collaborate
- Defecting now may have a consequence on multiple games
- There is the opportunity of influencing other player for future games
• in 1980, Robert Axelrod, professor of political science at the University of Michigan, held a tournament of various strategies for the iterated prisoner's dilemma.

• he invited a number of well-known game theorists to submit strategies to be run by computers.

• in the tournament, programs played games against each other strategies and against their same strategy, repeatedly.

• each strategy
  • specified whether to cooperate or defect based on the previous moves of both the strategy and its opponent.
  • made 200 rounds against each other strategy and itself
AXELROD'S TOURNAMENT

14 strategies where submitted + Random Strategy. Examples:

- Always defect: defects on every turn.
  - what game theory advocates, but it misses the chance to gain larger payoffs by cooperating with an opponent who is ready to cooperate.

- Always cooperate: does very well when matched against itself. However, if the opponent chooses to defect, then this strategy will do badly.

- Random: cooperates 50% of the time.

...and many others.....some examples of the results:
AND THE WINNER WAS... TIT FOR TAT!
**IPD: “TIT FOR TAT”**

- **always cooperate at the first turn**

TIT FOR TAT

- From now on, copy what the adversary has done in the last round

TIT FOR TAT

Dipartimento di Informatica
Università degli Studi di Pisa

Laura Ricci
Game theory principles, Bittorrent
IPD: “TIT FOR TAT”

- from now on, copy what the adversary has done in the previous round
- the adversary which defeats
  - reacts quickly against defectors
  - even if he/she takes advantage of defeating, he/she is instantaneously punished.
- disincentives the opponent from defeating
“TIT FOR TAT”

- namely: give and take, “pan per focaccia”, in italian
- has been the best strategy in several tournaments where game theory players have taken part
- is not able to defeat a single adversary, because it always collaborate as first move
- is able to win the tournaments, hence globally it results a winning strategy
- is at the basis of the chocking algorithm defined by BitTorrent
**TIT FOR TAT: CHARACTERISTICS**

- **Cooperative**: always cooperate at the first turn.
- then, when the other person is cooperating, may be you are tempted to defect
- but if the other begins to defect
- ignore the temptation and maximize the long term cooperation

• in BitTorrent this corresponds to upload content, as first move (optimistic move) and never try to stop uploading
• cooperation: giving bandwidth to the community
TIT FOR TAT: CHARACTERISTICS

- **Retaliating:**
  - stop collaboration if the partner does not collaborate.
  - otherwise, partner could defeat more times

- In BitTorrent “chock” the data upload if the partner does not send data or it sends data at a low rate
- defection: choke the upload toward a non collaborating peer
TIT FOR TAT: CHARACTERISTICS

- **Forgiveness:**
  - restart collaboration if the partner begins to collaborate, even if he/she was not previously collaborating.
  - In BitTorrent a peer restarts data upload when the partner collaborate

- **Not envious:** do not try to do better than the other
TIT FOR TAT CHARACTERISTICS IN A SNAPSHOT

Laura Ricci
Game theory principles, Bittorrent
**TIT FOR TAT CANNOT BEAT THE OPPONENT**

<table>
<thead>
<tr>
<th>Tit for Tac</th>
<th>0</th>
<th>5</th>
<th>3</th>
<th>0</th>
<th>5</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Always Defect</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tit for Tac</th>
<th>0</th>
<th>5</th>
<th>3</th>
<th>3</th>
<th>0</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Always Defect</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

| 0 | 1 | 1 | 1 | 1 | 4 |

- Tit for Tac cannot beat the opponent, because it just copies the opponent: can tie or lose: depends if the opponent defeat on the last round
- The opposite is true for always defect
Bittorrent is basically a Content Distribution Network (CDN): a distributed set of hosts cooperating to distribute large set of data (for instance multimedia data) to the end users.

The flash crowd phenomena:
- a web server is accessed by a large number of persons in a small time period, due to an important event
- the server must manage, in this time period, a very high degree of unexpected traffic

The CDN solution:
- data and/or service replication on different mirror servers
- goal: optimize the bandwidth usage

Examples
- Commercial: Akami, AppStream, Globix, Bittorrent
- Academic: Coral (P2P), Globule
Content Distribution Networks

File is stored on the server.
CONTENT DISTRIBUTION NETWORKS

Server distributes the first 2 pieces of the file
Server distributes two further pieces: in parallel, previous pieces are distributed.
CONTENT DISTRIBUTION NETWORKS

Upload of the last pieces
Server is no more involved in the file distribution
CONTENT DISTRIBUTION NETWORKS

Laura Ricci
Game theory principles, Bittorrent
CONTENT DISTRIBUTION NETWORKS

Laura Ricci
Game theory principles, Bittorrent
CONTENT DISTRIBUTION NETWORKS
CONTENT DISTRIBUTION NETWORKS

File complete on all peers
the peer which is going to share a file generates a descriptor of the file, the file .torrent, and publishes it on a server.

the descriptor includes a reference to a tracker, an active entity which coordinates the peers sharing the file

swarm: set of peers collaborating to the distribution of the same file coordinated by the same tracker

seeder: peer which owns all the file parts
   - at starting time: a single seeder S exists, the peer publishing the content, creates the file .torrent e publishes it
   - if S leaves the network after having “seeded” a set of peers, in the following, new peers become seeders. These are the peers which have downloaded the whole file and are still in the network

leecher:
   - the peer which has some part or no part of the file and downloads the file from the seeders and/or from other lechers.
   - at the beginning, it looks for the .torrent file to start the download
BITTORRENT: OVERVIEW

• Key Idea: In 2002, Bran Cohen
  • popularity exhibits temporal locality (Flash Crowds)
  • E.g. CNN on 9/11, new movie/game release,...
  • efficient content distribution system using file swarming

• initial release do not perform all the functions of a typical p2p system, like searching
  • later introduce Kademlia for searching

• Initially Focused on Efficient Fetching, not Searching
  • Distribute the same file to all peers
    • throughput increases with the number of downloaders via the efficient use of network bandwidth
  • Single publisher, multiple downloaders

• Has many “legal” publishers
  • Blizzard Entertainment using it to distribute the beta of their new game
• The peer Seeder (grey) is going to share a file
1. upload the .torrent on a Torrent Server
2. opens a connection to the tracker and informs it of its own existence: it is the only peer having the file
PROTOCOL OVERVIEW

1. Upload torrent file
2. Provide first seed
3. Post search request and retrieve link to torrent file
4. Contact tracker
5. Contact seeder for pieces
6. Trade pieces with peers

• the peer Peer (grey) is going to download file,
  3. downloads the file descriptor (.torrent) and opens it through the BitTorrent client
  4. opens a connection to the tracker and informs it of its own existence and receives from the tracker a list of peers of the swarm
  5. + 6. opens a set of connections with the peers of the swarm.
PROTOCOL OVERVIEW

- the peer Peer (grey) is going to download file,
  5. + 6. ask other peer about the parts they own
deares its interest in some part of the file
goess on exchanging information with the peers in the swarm
- if Peer remains online when it has finished the file download, it goes on distributing the file, becoming a seeder
PROTOCOL OVERVIEW

Laura Ricci
Game theory principles, Bittorrent
A SNAPSHOT OF A SWARM

2. Retrieve and trade pieces

1. Obtain list of peers

Tracker
PIECES AND BLOCKS

• Content is split into pieces (256KB - 2MB)
  • the smaller unit of retransmission: get small amounts of verifiable data from several peers at a time. Each time a piece is fully downloaded, check it (using the SHA1 algorithm) against what the torrent file tells.
    • typically 256/512/1024/2048 kByte
    • one SHA-1 hash per piece in the .torrent file
    • size adapted to have a reasonably small .torrent file

• Pieces are split into blocks (16KB)
THE TORRENT FILE

• .torrent file encoded using bencoding

• Info key
  • length on the content in bytes
  • md5 hash of the content (optional)
    • not used by the protocol
    • pieces SHA-1 hash are enough
  • File Name
  • Piece length (256kB, 512kB, 1024kB, etc.)
  • Concatenation of all pieces SHA-1 hash
  • Announce URL of the tracker (HTTP)
    • Possibility of announce list for backup trackers
  • Some optional fields
    • creation date, comment, created by,....
THE TORRENT FILE

- Automatically generated by proper tools (maketorrent) or directly by the Bittorrent client
- When the .torrent is generated, the tool requires to the client
  - All the information to set up the .torrent
  - In particular the address of a tracker. Two alternatives
    - choose the address of the tracker from a predefined list
    - insert the address of a known tracker, for instance:
      http://www.smarttorrent.com:2710/announce
PEER ID

- Peer ID = client ID + random string
  - Client ID: name of the client + its version
  - Random string: different each time the client is restarted (take as seed the system clock + the IP address of the peer)

Examples:
- M4-0-2--41dabfd4760b
- AZ2306-LwkWkRU95L9s
- AZ2402-YbqhPheosA4a
- BC0062->*
- UT1500-
- xfdMm\xb9\x96\xf0\xb8\xdf
PEER BOOTSTRAP

A peer downloads a .torrent file from a web server
- The peer client retrieves the tracker’s URL and connects to it (HTTP GET)

- The peer sends to the tracker
  - Info_hash (SHA-1 of the info key)
  - Peer ID
  - Port the client listen on
  - Number of peers wanted (default 50)
  - Various statistics (periodically computed)
    - Uploaded, downloaded, left, event (started, stopped, completed)

- The tracker returns
  - Failure raison
  - Interval between statistics
  - A random list of peers already in the torrent
    - Peer ID, peer IP, peer port
    - Typically 50 peers
  - Statistics
    - Complete/incomplete (seed/leechers)
PEER BOOTSTRAP

- Tracker is a web server acts as a certification authority
  - If a .torrent file is certified by the web server, there is no way to corrupt the torrent
    - each piece SHA-1 hash is in the .torrent file
    - peers exploits the .torrent received from the tracker to check the integrity of the pieces received from the P2P network
- Tracker is not involved in file distribution
- Peer connects to each peer returned by the tracker
  - at most 40 outgoing connections
  - remaining peers kept as a pool of peers to connect to
- Peer set (neighbor set) size 80: list of peers that known by the peer P
  - received them from the tracker
  - which have contacted the peer later
- In the last versions, use Kademlia DHT to implement a trackerless Bitcoin
Each peer maintains for each remote peer it is connected to the following state:

- **am_choking**: the local peer is choking the remote peer
  - it does not want to send data to the remote peer because the local peer is not satisfied about how this peer collaborates

- **amInterested**: the local peer is interested in at least one piece on the remote peer

- **peer_choking**: the remote peer is choking the local peer
  - the remote peer does not want to send data to the local peer, because it is not satisfied by the collaboration with the local peer

- **peerInterested**: the remote peer is interested in at least one piece of the local peer

The **local** peer can receive data from a remote peer if:

- the **local** peer is interested in the **remote** peer
- the **remote** peer unchoked the **local** peer
A is not willing to upload to B
A is not interested in B’s Pieces

B is willing to give to A
B is interested in A’s Pieces
Download the Bittorrent client from https://www.bittorrent.com/lang/it/
Download control
- d – interested and choked
- D – interested and unchoked
- K – uninterested and unchoked
- S – snubbed (no data received in 60 seconds)
- F – piece(s) failed to hash

Most peers are d or D. No need to connect with uninteresting peers.
Error states. Connection should be closed.
- K – Interested and unchoked
- S – snubbed (no data received in 60 seconds)
- F – piece(s) failed to hash
- **Download control**
  - d – interested and choked
  - D – interested and unchoked
  - K – uninterested and unchoked
  - S – snubbed (no data received in 60 seconds)
  - F – piece(s) failed to hash

- **Upload control**
  - u – interested and choked
  - U – interested and unchoked
  - O – optimistic unchoke
  - ? – uninterested and unchoked
BITTORRENT: CONNECTION STATUS

- Download control
  - d – interested and choked
  - D – interested and unchoked
  - K – uninterested and unchoked
  - S – snubbed (no data received in 60 seconds)
  - F – piece(s) failed to hash

- Upload control
  - u – interested and choked
  - U – interested and unchoked
  - O – optimistic unchoke
  - ? – uninterested and unchoked

- Connection information
  - I – incoming connection
  - E/e – Using protocol encryption

- Flags
  - ud – used UDP
  - h – used UDP hole punching
  - P – connection uses µTP

- How was this peer located?
  - H – DHT (distributed hash table)
  - L – local peer discovery (multicast)
  - X – peer exchange
PEER TO PEER PROTOCOL

- **Peer wire protocol**: peer to peer protocol to exchange pieces of files

- Unlike client-server architectures, this is not the client who decides when to receive data:
  - the peer which receives a connection request may refuse the connection
  - if the hash does not correspond to any hash of the files it is participating to download
  - if the ID of the peer does not correspond to those received by the tracker

- when the connection is established, each peer communicates to the partner the indexes of the pieces of the file which it owns

- each peer of a swarm knows the peers distribution within the swarm
PEER WIRE PROTOCOL

• **Handshake**
  - two way handshake
  - to initiate a connection between two peers
  - once initiated, the connection is symmetric
  - contains (68 bytes)
    - Pstrlen=19 (protocol string identifier length)
    - Pstr=“BitTorrent protocol”
    - Reserved (8 bytes)
    - Info_hash
    - Peer ID

• **Keep Alive**
  - sent each two minutes if no message is received from the connection
  - to check if the client has closed the connection
**Bitfield** <bietfield>

- first message sent after the handshake
- no more sent in the following
- sent by both peers once the connection is initialized
- Bit i in the bitfield is set to 1 if the peer has piece i, 0 otherwise

Soon, "B" summarized the pieces available in its swarm.

<table>
<thead>
<tr>
<th>piece</th>
<th>Total number in the swarm of B</th>
<th>Corresponding peers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>(A, D, E)</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>(A, B, C)</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>(A, D)</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>(A, C, E)</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>(A, C, D, E)</td>
</tr>
</tbody>
</table>
**Have**

- a message sent from a peer to all other peers in its swarm, it indicates that a new piece is completely downloaded and it is available for sharing
- sent to all the peers a peer is connected to
- information is useful for torrent monitoring: exactly knows who has what
- when a peer receives a HAVE message, it updates its swarm, indicating that a new peer became a provider of this piece

**Mainline 5.0.5 comment in source code**

```
# should we send a have message if peer already has the piece?
# yes! it is low bandwidth and useful for that peer.
```
PEER WIRE PROTOCOL: MESSAGES

- **Interested:**
  - Sent from a peer A to another peer B to indicate that it is interested in any of the receiver’s resources (pieces). Contains the piece index.

- **Not Interested**
  - A message sent after receiving a HAVE message regarding a specific piece to indicate that the sender is not interested in that piece anymore. Contains a peer index.

- **Choke/Unchoke** required to implement the unchocking algorithm
  - **choke**: sent from A to B when A chokes B
    - no request of download from B is accepted by A from now on.
  - **unchoke**: sent from A to B when A unchokes B
    - a message that is sent from a peer to another to indicate that it is allowed to request pieces from the sender at the moment.
PEER WIRE PROTOCOL: MESSAGES

Message to exchange blocks:

• **Request** <index, begin, length>
  • a message that is sent from a peer to another peer requesting a specific block in a specific piece,
  • sent from peer A to peer B to request to peer B the block
    • of the piece with index <index>
    • starting with an offset <begin> within the piece
    • of length <length>
  • it can be sent only after receiving UNCHOKE message.

• **Piece** <index> <begin> <block>
  • only one message used to send blocks
  • sent from peer A to peer B to send a block of data to peer B
    • of the piece with index <index>
    • starting with an offset <begin> within the piece
    • payload is <block>
PEER WIRE PROTOCOL: MESSAGES

- **Cancel**
  - a peer sent from a peer to another to indicate that it already got a piece and it is not interested in it anymore.
  - used in **end game mode** only
  - sent from peer A to peer B to **cancel** a request already sent to peer B for the block
    - of the piece with index \(<\text{index}>\)
    - starting with an offset \(<\text{begin}>\) within the piece
    - of a length \(<\text{length}>\)
PEER WIRE PROTOCOL: MESSAGES

Peer C (Leecher)  Peer A (Leecher)  Peer B (Seeder)

BITFIELD  BITFIELD  BITFIELD

INTRESTED

UNCHOKE

REQUEST

PIECE

HAVE

REQUEST

PIECE
• Algorithms for the selection of peers
  • Chocking/ Unchocking (based on the game theory)
    • rewards the most “collaborative” peers, those which have given a larger number of content in the past
  • Optimistic Unchocking

• Algorithms For the selection of the pieces of the files
  • Strict Priority
  • Random First Piece
  • Rarest First
  • End Game