



UNIVERSIDADE DO ESTADO DO RIO DE JANEIRO

DISCIPLINA: SISTEMAS DISTRIBUÍDOS – 2025/1ºSEMESTRE

CURSO: CIÊNCIA DA COMPUTAÇÃO – 2025.1

Alunos: Jessica Coutinho de Oliveira

Lucas Alberto Da Silva De Aquino

Lucas Ferreira Lima

Pamella Silva Japson

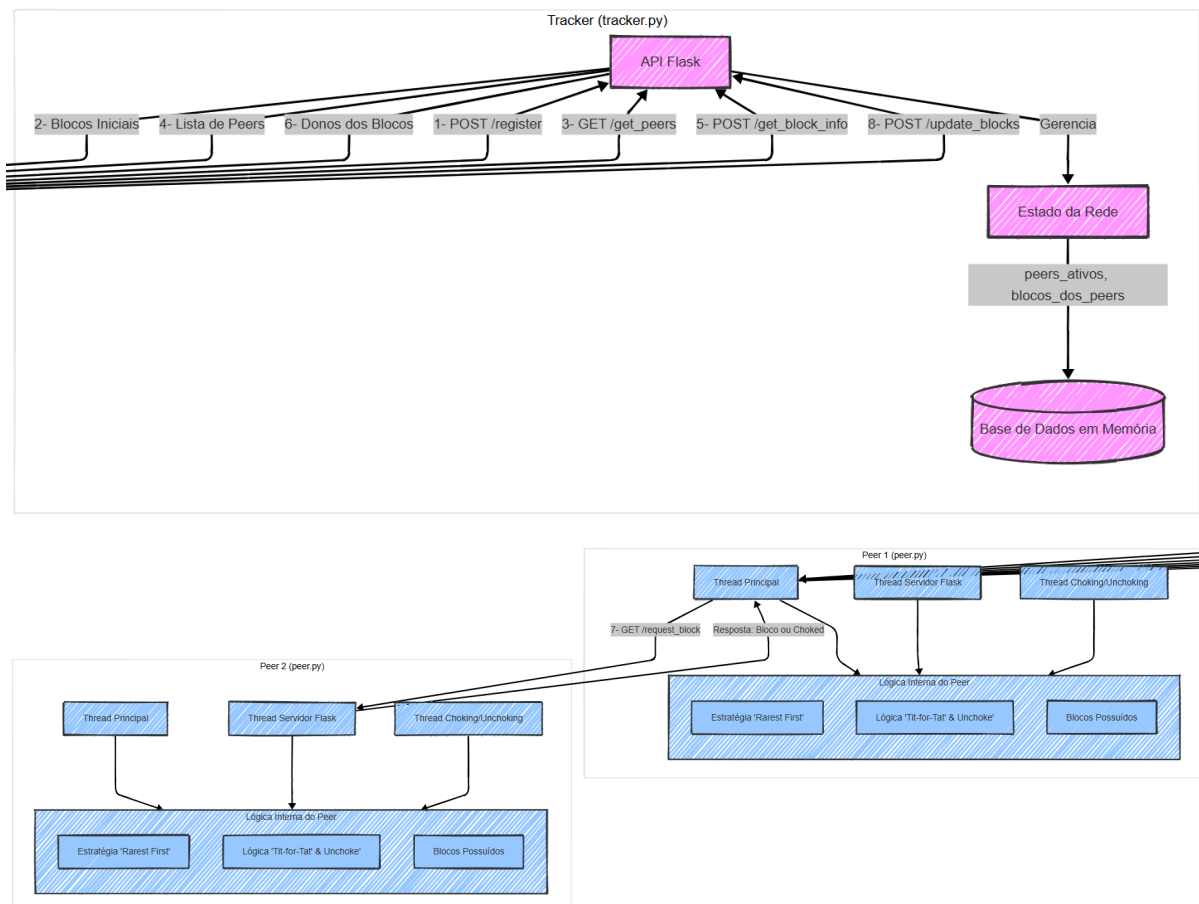
RIO DE JANEIRO-RJ

2025

• Objetivo

O objetivo do trabalho é desenvolver um sistema distribuído de compartilhamento de arquivos, inspirado no BitTorrent, que simula a divisão de arquivos em blocos, a troca direta de blocos entre peers e o uso de estratégias de distribuição como Rarest First e Tit-for-Tat simplificado, além de um tracker central responsável apenas pela descoberta de peers e distribuição inicial dos blocos. O sistema deve permitir que os peers cooperem na obtenção dos blocos de forma descentralizada, aplicando políticas de desbloqueio e bloqueio (choke/unchoke) para controlar as trocas, promovendo equilíbrio na distribuição e eficiência na replicação dos dados na rede.

• Diagrama de Arquitetura



● Tracker

O tracker.py atua como coordenador da rede. Ele não armazena os arquivos, apenas mantém o estado da rede (quem está conectado e o que possui). Inicialmente, atua como seeder distribuindo os blocos entre os peers para simplificar a solução.

- **API Flask:** disponibiliza endpoints HTTP para os peers interagirem com o tracker.
- **/register:** registra um novo peer, atribui-lhe um conjunto inicial de blocos e informa o número total de blocos. A lógica garante que todos os blocos sejam eventualmente introduzidos na rede.
- **/get_peers:** fornece a um peer uma lista de outros peers ativos.
- **/get_block_info:** informa quais peers possuem um determinado conjunto de blocos.
- **/update_blocks:** permite que um peer informe ao tracker sobre novos blocos que adquiriu.
- **Estado da Rede:** Armazena em memória:
 - **peers_ativos:** um *dicionário* com os IDs e endereços dos peers online.
 - **blocos_dos_peers:** um *dicionário* que mapeia cada peer ao conjunto de blocos que ele possui.

● Peer

Cada instância de peer.py é um nó que participa do compartilhamento tanto como cliente, quanto como servidor (distribuindo os blocos solicitados):

Thread Principal:

- **Registro:** Inicia a comunicação com o tracker.
- **Download:** Entra em loop até baixar todos os blocos em falta.
- **Seleção de Bloco:** Usa a estratégia "Rarest First".
- **Pedido de Bloco:** Solicita o bloco mais raro a um dos peers que o possui.
- **Atualização:** Após baixar um bloco, informa ao tracker.
- **Modo Seeding:** Quando possui todos os blocos, entra em modo de "seeding", apenas distribuindo blocos a outros peers que pedem.

Thread do Servidor Flask:

Cada peer executa o seu próprio servidor para responder a pedidos de outros peers.

- **/request_block/<id_bloco>:** envia um bloco a outro peer.

Este endpoint implementa a política de "*choking*". Ele só enviará o bloco se o peer solicitante estiver na lista de "desbloqueados" (peers_desbloqueados) ou for o "otimista desbloqueado" (peer_otimista_desbloqueado).

Thread de Choking/Unchoking:

- Executa em segundo plano para gerir com quem o peer irá compartilhar blocos (upload).
- **Tit-for-Tat:** Periodicamente, reavalia quais peers lhe forneceram os blocos mais úteis (os mais raros) e os desbloqueia (unchoke) para que tenham prioridade de upload. Tipicamente, conforme o enunciado, os 4 melhores são escolhidos.
- **Optimistic Unchoke:** A cada 10 segundos, escolhe um peer aleatório e o desbloqueia. Dá aos novos peers uma oportunidade de começar a baixar algo, mesmo que não tenham blocos raros para oferecer em troca inicialmente.

● Fluxo de Comunicação Esperado

- 1) Um novo Peer 1 inicia e envia POST /register ao Tracker.
- 2) O Tracker responde com blocos iniciais e o total de blocos.
- 3) O Peer 1 pede uma lista de outros peers (GET /get_peers).
- 4) Para decidir qual bloco baixar, o Peer 1 pede ao Tracker informações sobre os blocos que faltam (POST /get_block_info).
- 5) O Tracker responde, avisando que o Peer 2 possui um bloco raro que o Peer 1 precisa.
- 6) O Peer 1 envia um GET /request_block diretamente ao Peer 2.
- 7) O servidor do Peer 2 verifica a lógica de "unchoke". Se o Peer 1 estiver desbloqueado, o Peer 2 envia os dados do bloco.
- 8) O Peer 1 recebe o bloco, armazena e informa o Tracker através de POST /update_blocks.
- 9) O ciclo se repete até que o Peer 1 tenha o arquivo completo e vire um "seeder". A thread de "unchoke" para de funcionar nesse momento e o Peer passa a enviar o bloco solicitado para qualquer Peer que pedir.

● Estados de um Peer:

- **Registrando:** Estado inicial em que o peer anuncia ao tracker sua entrada na rede.
- **Downloading:** Estado ativo onde o peer ainda não possui todos os blocos.
- **Seeding:** Estado alcançado quando o peer baixou 100% dos blocos.

● Relações entre Peers:

- **Choked:** O estado padrão. O peer não fará upload de blocos para um peer "choked", que fez um pedido.
- **Unchoked:** O peer está disposto a fazer upload de blocos para um peer "unchoked". Esta condição é determinada pelas lógicas de "Tit-for-Tat" e "Optimistic Unchoke".

● Estratégia Rarest First

É usada pela Thread Principal do Peer para decidir qual bloco baixar.

O mecanismo implementado consiste em:

- **Identificar Blocos em Falta:** O peer determina quais blocos ainda não possui.
- **Consultar o Tracker:** Ele envia a lista de blocos em falta para o tracker através da mensagem *POST /get_block_info*.
- **Analisar a Raridade:** O tracker responde informando quais peers na rede possuem cada um desses blocos. O peer analisa esta resposta para encontrar o bloco que é possuído pelo menor número de peers.
- **Solicitar o Bloco:** Uma vez identificado o bloco mais raro, o peer envia uma mensagem *GET /request_block* para um dos peers que o possui para iniciar o download.

● Estratégia Tit-for-Tat

Implementada pela Thread de Choking/Unchoking do Peer que gere a quem ele pode enviar blocos.

- **Avaliação Periódica:** a cada 20 segundos, o peer reavalia a sua lista de desbloqueados.
- **Cálculo de Pontuação:** Para cada peer conhecido, é calculada uma pontuação baseada na utilidade desse peer (quantos blocos raros ele possui).
- **Seleção dos Melhores Peers:** Os peers são ordenados com base nessa pontuação.
- **Desbloqueio:** O peer desbloqueia (unchoke) os 4 melhores peers da lista.
- **Bloqueio:** Todos os outros peers permanecem no estado "choked", o que significa que os seus pedidos de blocos serão temporariamente negados.
- **Exceção (Optimistic Unchoke):** um peer aleatório é "desbloqueado otimisticamente" a cada 10 segundos.

● Testes

Os testes demonstraram que o sistema funciona de forma eficiente, com os peers conseguindo completar o download dos blocos em um tempo razoável e mantendo boa distribuição dos dados na rede. As estratégias de Rarest First e Tit-for-Tat simplificado mostraram-se eficazes para equilibrar a troca de blocos entre os peers, evitando gargalos.

O script de execução foi configurado para rodar em 8 peers simultaneamente, em abas do terminal diferentes, permitindo testar o comportamento do sistema em uma rede com múltiplos nós atuando de forma distribuída. Esse cenário possibilitou observar a

eficiência das estratégias de distribuição de blocos, a dinâmica do choke/unchoke e a capacidade dos peers em completar o download do arquivo de maneira cooperativa.

```
C:\Windows\system32\cmd.exe
Iniciando o Tracker na porta 5000...
2025-06-16 06:05:42,017 - INFO - Rastreador iniciado com 50 blocos. Aguardando peers para distribuição inicial...
* Serving Flask app 'tracker'
* Debug mode: on
2025-06-16 06:05:42,095 - INFO - WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
2025-06-16 06:05:42,095 - INFO - Press CTRL+C to quit
```

Imagem: inicialização do Tracker

```
Peer 8 - python peer.py peer_8_5008
* Serving Flask app 'peer'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5008
Press CTRL+C to quit
2025-06-16 06:06:40,675 - peer_8 - INFO - Registrado com sucesso. Recebi 10 blocos. Total na rede: 50
2025-06-16 06:06:40,675 - peer_8 - INFO - Blocos atuais (10/50): [0, 2, 9, 10, 16, 25, 27, 35, 36, 42]
2025-06-16 06:06:40,737 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_1 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:40] "GET /request_block/0?peer_id=peer_1 HTTP/1.1" 403 -
2025-06-16 06:06:40,753 - peer_8 - WARNING - Pedido do bloco 1 para peer_2 negado (choked).
2025-06-16 06:06:40,816 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_3 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:40] "GET /request_block/0?peer_id=peer_3 HTTP/1.1" 403 -
2025-06-16 06:06:40,816 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_7 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:40] "GET /request_block/0?peer_id=peer_7 HTTP/1.1" 403 -
2025-06-16 06:06:40,831 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_5 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:40] "GET /request_block/0?peer_id=peer_5 HTTP/1.1" 403 -
2025-06-16 06:06:40,878 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_4 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:40] "GET /request_block/0?peer_id=peer_4 HTTP/1.1" 403 -
2025-06-16 06:06:40,907 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_7 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:41] "GET /request_block/0?peer_id=peer_7 HTTP/1.1" 403 -
2025-06-16 06:06:42,170 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_1 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:42] "GET /request_block/0?peer_id=peer_1 HTTP/1.1" 403 -
2025-06-16 06:06:42,211 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_5 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:42] "GET /request_block/0?peer_id=peer_5 HTTP/1.1" 403 -
2025-06-16 06:06:42,467 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_4 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:42] "GET /request_block/0?peer_id=peer_4 HTTP/1.1" 403 -
2025-06-16 06:06:42,600 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_3 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:42] "GET /request_block/0?peer_id=peer_3 HTTP/1.1" 403 -
2025-06-16 06:06:43,288 - peer_8 - WARNING - Rejeitando pedido do bloco 0 de peer_5 (choked).
127.0.0.1 - - [16/Jun/2025 06:06:43] "GET /request_block/0?peer_id=peer_5 HTTP/1.1" 403 -
```

Imagem: inicialização instável do Peer que pode ocorrer em algumas simulações, mas que melhora após alguns segundos.

```
Peer 8 - python peer.py peer_8 5008
2025-06-16 06:10:09,291 - peer_8 - INFO - Sucesso! Bloco 22 recebido de peer_3. Total: 28/50
2025-06-16 06:10:10,239 - peer_8 - INFO - Sucesso! Bloco 23 recebido de peer_4. Total: 29/50
2025-06-16 06:10:12,203 - peer_8 - INFO - Sucesso! Bloco 24 recebido de peer_4. Total: 30/50
2025-06-16 06:10:14,215 - peer_8 - INFO - Sucesso! Bloco 26 recebido de peer_5. Total: 31/50
2025-06-16 06:10:14,870 - peer_8 - INFO - Sucesso! Bloco 28 recebido de peer_7. Total: 32/50
2025-06-16 06:10:16,703 - peer_8 - INFO - Sucesso! Bloco 3 recebido de peer_5. Total: 33/50
2025-06-16 06:10:17,456 - peer_8 - INFO - Peers desbloqueados (fixos): {'peer_1', 'peer_5', 'peer_6', 'peer_2'}
2025-06-16 06:10:18,209 - peer_8 - INFO - Sucesso! Bloco 30 recebido de peer_2. Total: 34/50
2025-06-16 06:10:20,119 - peer_8 - INFO - Sucesso! Bloco 31 recebido de peer_2. Total: 35/50
2025-06-16 06:10:21,077 - peer_8 - INFO - Sucesso! Bloco 32 recebido de peer_6. Total: 36/50
2025-06-16 06:10:22,182 - peer_8 - INFO - Sucesso! Bloco 33 recebido de peer_7. Total: 37/50
2025-06-16 06:10:23,584 - peer_8 - INFO - Sucesso! Bloco 34 recebido de peer_5. Total: 38/50
2025-06-16 06:10:25,493 - peer_8 - INFO - Sucesso! Bloco 37 recebido de peer_1. Total: 39/50
2025-06-16 06:10:26,540 - peer_8 - INFO - Sucesso! Bloco 38 recebido de peer_5. Total: 40/50
2025-06-16 06:10:27,457 - peer_8 - INFO - Unchoke otimista: peer_7
2025-06-16 06:10:28,556 - peer_8 - INFO - Sucesso! Bloco 39 recebido de peer_5. Total: 41/50
2025-06-16 06:10:29,108 - peer_8 - INFO - Sucesso! Bloco 4 recebido de peer_3. Total: 42/50
2025-06-16 06:10:30,402 - peer_8 - INFO - Sucesso! Bloco 41 recebido de peer_3. Total: 43/50
2025-06-16 06:10:31,699 - peer_8 - INFO - Sucesso! Bloco 43 recebido de peer_2. Total: 44/50
2025-06-16 06:10:33,293 - peer_8 - INFO - Sucesso! Bloco 44 recebido de peer_2. Total: 45/50
2025-06-16 06:10:34,258 - peer_8 - INFO - Sucesso! Bloco 46 recebido de peer_4. Total: 46/50
2025-06-16 06:10:35,772 - peer_8 - INFO - Sucesso! Bloco 47 recebido de peer_6. Total: 47/50
2025-06-16 06:10:37,463 - peer_8 - INFO - Peers desbloqueados (fixos): {'peer_1', 'peer_5', 'peer_6', 'peer_2'}
2025-06-16 06:10:37,608 - peer_8 - INFO - Sucesso! Bloco 48 recebido de peer_4. Total: 48/50
2025-06-16 06:10:39,285 - peer_8 - INFO - Sucesso! Bloco 5 recebido de peer_7. Total: 49/50
2025-06-16 06:10:41,264 - peer_8 - INFO - Sucesso! Bloco 7 recebido de peer_3. Total: 50/50
2025-06-16 06:10:42,581 - peer_8 - INFO - --- ARQUIVO COMPLETO! ---
2025-06-16 06:10:42,581 - peer_8 - INFO - Todos os 50 blocos foram baixados. Entrando em modo de seeding.
```

Imagem: término do download com o Peer entrando em modo seeding.

● Dificuldades enfrentadas

Entre as principais dificuldades superadas ao longo do desenvolvimento estão:

- problemas iniciais de sincronização entre os peers;
- erros ocasionais de conexão devido à ordem de inicialização dos nós;
- desafios na implementação do mecanismo de choke/unchoke para garantir que os peers respeitassem corretamente as regras de bloqueio e desbloqueio durante as trocas;
- o modo seeding foi implementado pois os peers que terminavam de baixar os blocos estavam saindo e fazendo com que os demais ficassem incapazes de completar o download;
- Por falta de tempo tentando fazer o código rodar corretamente, o grupo não conseguiu organizar melhor o código ou realizar testes mais robustos.

- **Comentários do grupo**

- **Pamella:**

Durante o desenvolvimento do projeto MiniBit, atuei principalmente na implementação e ajustes do peer, contribuindo para que ele funcionasse corretamente e se comunicasse de forma eficiente com o tracker e os outros peers. Participei ativamente dos testes, validando a comunicação e o funcionamento do sistema, além de colaborar na documentação, sendo responsável pela escrita do README e por partes do relatório técnico, buscando deixar as informações claras e acessíveis. Tive algumas dificuldades iniciais para entender o funcionamento da comunicação entre os peers, mas, com apoio do grupo e pesquisa, consegui superar esses desafios e aprender bastante sobre sistemas distribuídos e compartilhamento de arquivos. O trabalho em equipe foi bem organizado, com boa divisão de tarefas, e sinto que contribui de forma significativa para o sucesso do projeto.

- **Lucas Alberto:**

Durante o desenvolvimento do projeto, senti dificuldades em entender como os diferentes módulos de cada componente do grupo iriam se comunicar e conversar de uma maneira geral. Fiz testes e participei da elaboração da documentação. Entendo que o trabalho foi bastante positivo para todos, visto que possibilitou a visualização do sistema peer-to-peer na prática, mesmo com as limitações do grupo mencionadas anteriormente. O grupo me auxiliou a entender como a função atuava e mesclando com o material do professor, pude compreender um pouco do funcionamento de um sistema P2P.

- **Jessica:**

No desenvolvimento do projeto, participei principalmente da implementação e dos ajustes no peer, garantindo que ele funcionasse corretamente e se comunicasse bem com o tracker e os outros peers. Contribuí na validação das estratégias de distribuição de blocos, como Rarest First e Tit-for-Tat simplificado, além de ajudar a aplicar corretamente a lógica de choke e unchoke. Trabalhei bastante na identificação e correção de problemas, como dificuldades na sincronização entre os peers e na fase inicial de comunicação. Também ajudei na organização e produção da documentação.

- **Lucas Ferreira**

Atuei na prototipagem inicial, criando uma versão funcional com bibliotecas de alto nível, o que nos ajudou a entender o fluxo entre peers e tracker. Com o avanço do projeto, refatorei partes para maior controle da rede. Também colaborei na definição da arquitetura, preparação do ambiente de testes distribuídos e resolução de problemas de concorrência entre threads durante requisições simultâneas ao tracker.