Universidade Federal de Santa Catarina – UFSC
Departamento em Informática e Estatística
INE 5418 – Computação Distribuída
Semestre 2024/2

Definição do Trabalho 1: Sistemas Peer-to-peer

Comunicação *peer-to-peer (P2P)* é uma alternativa ao modelo cliente servidor, sendo amplamente utilizada para o compartilhamento de mídias digitais. Em redes *P2P*, os pares compartilham recursos por meio da transferência direta entre os pares, sem que o recurso em si passe por um servidor.

Em uma rede de compartilhamento de arquivos *peer to peer* **não estruturada,** todos os *peers* possuem um papel equivalente. Cada *peer* possui arquivos armazenados localmente que estão disponíveis para serem transferidos para outros *peers*. Para obter um arquivo, um *peer* inicia uma busca por meio de ***flooding*** (inundação) de mensagens. A busca é limitada pelo parâmetro **TTL** (*time to live*) da mensagem, que limita o número de saltos, ou difusões, realizados. Quando a mensagem de requisição chega a um *peer* com o recurso, ele responde diretamente ao *peer* que iniciou a solicitação, permitindo estabelecer um canal de comunicação direto e **transferir o recurso diretamente ao solicitante**.

Nesse tipo de sistema, a transferência de recursos pode **ser feita em *chunks*** (porções), sendo possível **obter cada *chunk* de *peers* diferentes** com o objetivo de acelerar a transferência do recurso. Ao fim da transferência, o solicitante pode **reconstruir** o arquivo concatenando as diferentes partes (*chunks*). Não é necessário que um par tenha todas as partes do arquivo para que ele possa transferir uma parte, basta que ele tenha a parte solicitada.

Neste trabalho, o objetivo é simular uma rede P2P não estruturada de compartilhamento de arquivos. A simulação pode ser feita localmente, mas é necessário que:

- Cada par seja uma instância diferente do programa;
- A comunicação seja feita com Berkeley *sockets*.

A topologia da rede é definida estaticamente e pode ser obtida por um arquivo de configuração. O arquivo deve ser nomeado como `topologia.txt` e cada linha do arquivo indica o id de um nodo, seguido por uma lista de ids dos nodos diretamente conectados a este nodo. Por exemplo, o quadro a seguir ilustra uma topologia com 5 nodos e as conexões e a representação de um grafo com a vizinhança de cada nodo:

```
0: 1, 3, 4
1: 0, 2
2: 4
3: 0
4: 0, 2
```

Na inicialização do sistema, é necessário iniciar os N nodos da rede (no exemplo acima, 5 nodos, com ids de 0 a 4). Cada nodo terá acesso ao arquivo `topologia.txt` e cada nodo consegue fazer consultas apenas para os seus vizinhos diretos. Para a comunicação, é necessário que cada processo atenda requisições em endereços distintos para as requisições UDP e TPC. Além disso, cada nodo terá uma capacidade de transferência, que impacta na velocidade com que dados são transmitidos. Para estas definições, será utilizado um arquivo com as configurações dos nodos, chamado `config.txt`, indicando, para cada nodo: o IP, porta para comunicação UDP e bytes/s, conforme ilustração abaixo.

```
0: 127.0.0.1, 6000, 100
1: 127.0.0.1, 6001, 50
2: 127.0.0.1, 6002, 80
3: 127.0.0.1, 6003, 200
4: 127.0.0.1, 6004, 100
```

A descoberta de arquivos deve ser feita sobre o paradigma de troca de mensagens com a comunicação por *sockets* UDP. A descoberta de arquivos envolve o *flooding* de mensagens e a resposta de um *peer* que possui o arquivo ao *peer* solicitante. Para efeito da simulação, cada nodo que for transmitir uma mensagem de descoberta de arquivo deve atrasar a retransmissão em **1s**.

O formato de requisições e respostas por solicitação de arquivo fica a critério do grupo, assim como a estratégia de decisão sobre a quem solicitar cada chunk.

A etapa de transferência de um *chunk* de um arquivo deve ser feita com uma conexão TCP, que deve ser aberta somente para esse propósito, sendo terminada ao fim da transferência do *chunk* em questão. A velocidade total de envio deve ser limitada pela capacidade definida no arquivo de configuração, ou seja, o **total de bytes transferidos por segundo** de um nodo para seus *peers* não pode exceder a quantidade definida no arquivo de configuração do nodo.

Cada nodo já contém um conjunto de arquivos na inicialização, que devem ser colocados em um **diretório de mesmo nome do id do nodo**. Para simplificar a simulação, os arquivos já estão divididos em *chunks*. Os nomes dos arquivos seguem o formato “<nome>.ch<chunk>”. Ou seja, o arquivo “Ex.txt” quando dividido em 3 *chunks*, é composto pelos arquivos “Ex.txt.ch0”, “Ex.txt.ch1” e “Ex.txt.ch2”. Um mesmo *chunk* pode estar presente em diferentes *peers*, sendo admissível arquivos duplicados na rede.

Durante a execução dos programas de *peer,* deve ser possível oferecer como entrada arquivos de metadados necessários para a transferência de um arquivo. Esses arquivos devem ser nomeados como `arquivo.p2p`, contendo na primeira linha o nome do arquivo, na segunda linha o número de *chunks* e, na terceira linha, o valor inicial de TTL da mensagem difundida. Por exemplo, o quadro seguinte representa o arquivo necessário para receber o arquivo *image.png,* que está dividido em 10 *chunks* e deve ser buscado com no máximo 7 saltos:

```
image.png
10
7
```

Execução

Para realizar as simulações, com os N nodos da rede já iniciados e configurados pelos arquivos `topologia.txt` e `config.txt`, você deve oferecer como entrada um arquivo de metadados do tipo `<metadados>.p2p`. Deve ser possível acompanhar a recepção dos *chunks* sendo transferidos por meio de impressões na tela, contendo o *peer* enviando o arquivo de *chunk* e a porcentagem do arquivo já transferida. Assim que todos os *chunks* forem recebidos, os arquivos devem ser concatenados, formando o arquivo completo. Você pode desenvolver o programa em qualquer linguagem de programação, desde que a implementação atenda os requisitos definidos neste enunciado.

Entrega

O trabalho consiste em:

1. Implementar o programa executado pelos *peers* compatível com os arquivos de configuração e entrada descritos neste enunciado para simular uma rede *peer-to-peer* de transferência de arquivos.
2. Breve relatório que indique as principais decisões e estratégias de implementação utilizadas, instruções sobre como compilar e executar o código produzido, além de exemplos de saídas de execução com diferentes parametrizações. Ao final, discuta quais conclusões você observa ao variar parâmetros do sistema.

O trabalho pode ser realizado em **grupos de até 3 participantes**. O trabalho será apresentado em laboratório.

O código-fonte e relatório devem ser enviados pelo Moodle para análise e avaliação.

Os nomes dos participantes do grupo devem constar no relatório entregue no Moodle. Participantes com nomes não referenciados não serão considerados como membros do grupo.