UNIVERSIDADE ESTADUAL DE MATO GROSSO DO SUL
CURSO DE CIÊNCIA DA COMPUTAÇÃO
UNIDADE UNIVERSITÁRIA DE DOURADOS

Lincoln Martins Amorim

RGM: 41443

Vinicius da Silva Balbino

RGM: 43581

Documentação Trabalho P2P

DOURADOS - MS

2024

SUMÁRIO

1. SUMÁRIO DO PROBLEMA

O projeto visa a criação de um sistema de compartilhamento de arquivos distribuído usando uma rede de SuperNodes que coordena o acesso de clientes aos arquivos. Cada SuperNode armazena informações sobre quais clientes possuem cada arquivo e responde a requisições de download e upload. A rede é coordenada por um SuperNode mestre (coordenador), que gerencia a distribuição de arquivos e organiza uma nova eleição para eleger um coordenador substituto caso o nó mestre falhe. Este sistema é ideal para redes distribuídas com tolerância a falhas e grande demanda por compartilhamento de arquivos.

2. DESCRIÇÃO DO CÓDIGO

2.1. Estruturas de Dados, Constantes e Variáveis Globais

- **Struct SuperNode**: Representa um nó na rede e armazena seu ID (int) e endereço IP (Addr).
- **superNodes**: Mapa que armazena os SuperNodes registrados pelo coordenador. A chave é o ID do SuperNode e o valor é a instância SuperNode.
- **files**: Mapa que associa o nome do arquivo aos IPs dos clientes que possuem esse arquivo. Exemplo: files["file1.txt"]["192.168.1.5"] = true.
- **isMaster**: Booleano que identifica se o nó atual é o coordenador (mestre).
- **electionInProgress**: Booleano que indica se uma eleição está em andamento.
- **knownSuperNodes**: Lista dos IPs conhecidos de outros SuperNodes. Essencial para a transmissão de informações entre SuperNodes e para as eleições.
- **registerPort** (8080): Utilizada para registrar SuperNodes no coordenador.
- **releasePort** (8081): Permite ao coordenador liberar SuperNodes após o registro.
- **clientPort** (8082): Usada para comunicação entre clientes e SuperNodes.
- **broadcastPort** (8084): Porta para transmissões entre nós (broadcast de informações).

- **electionPort** (8085): Porta usada para a comunicação do processo de eleição.

2.2. Funções de Controle de SuperNodes e Coordenação

- **handleSuperNodeRegistration**: Esta função permite que o coordenador registre novos SuperNodes. Ele recebe a conexão, extrai o endereço IP do nó e envia uma confirmação de registro com o ID. Caso todos os SuperNodes se registrem com sucesso, a comunicação é liberada.
- **freeNode**: Envia uma mensagem de liberação para um SuperNode específico. Esse processo finaliza o registro do nó e permite que ele comece a se comunicar com o coordenador e outros SuperNodes.
- **freeSuperNodes**: Após todos os SuperNodes estarem registrados, envia uma mensagem para liberar a comunicação entre eles. Ele é executado em paralelo para evitar bloqueios no sistema.
- **broadcastSuperNodes**: Cria uma lista dos SuperNodes registrados e envia essa lista a todos os nós, mantendo-os informados sobre os nós ativos na rede.
- **startElection**: Inicia uma eleição para definir um novo coordenador, enviando mensagens a todos os nós com IDs maiores. Se nenhum nó responde, o nó que iniciou a eleição se torna o coordenador.
- **handleElection**: Gerencia a eleição, recebendo respostas dos nós contatados e confirmando se o nó atual é elegível como coordenador.
- **declareAsCoordinator**: Caso o nó atual seja escolhido como coordenador, essa função define seu status como mestre, atualiza o IP do coordenador e notifica todos os SuperNodes.
- **checkCoordinator**: Executa uma verificação periódica para confirmar se o coordenador está ativo. Em caso de falha, inicia uma nova eleição.

2.3. Funções de Comunicação com Clientes

- **handleUpload**: O cliente realiza o upload de um arquivo, e o SuperNode registra o arquivo e associa o IP do cliente ao arquivo em files.
- **handleDownload**: Gerencia as requisições de download de clientes, verificando localmente se o arquivo está disponível. Caso o arquivo não esteja

disponível, envia uma solicitação a outros SuperNodes (usando broadcastRequest) para encontrar o arquivo.

- **broadcastRequest**: Envia uma solicitação de broadcast para localizar o arquivo em outros SuperNodes. Retorna o IP do cliente que possui o arquivo ou uma mensagem de erro caso o arquivo não seja encontrado.
- **handleClientRequest**: Funciona como um servidor local do cliente, escutando na porta designada. Aceita conexões de outros clientes para servir arquivos por download direto.
- **uploadFile** e **downloadFile**: Essas funções permitem que o cliente interaja com o SuperNode. uploadFile envia um comando ao SuperNode para registrar o arquivo, e downloadFile solicita o arquivo ao SuperNode, recebendo o IP do cliente que possui o arquivo.
- **handleUserInteraction**: Cria um loop de interação que permite ao usuário enviar comandos ao sistema: realizar upload, download ou encerrar a conexão.

2.4. Inicialização e Configuração dos Nós

- **initializeNode**: Inicializa o nó determinando se ele é o coordenador. O nó coordenador registra os SuperNodes, libera a comunicação e mantém a estrutura da rede. O nó comum se registra com o coordenador, espera a liberação para iniciar a comunicação e monitora o coordenador para iniciar eleições em caso de falha.
- **receiveBroadcast**: Recebe a lista de SuperNodes do coordenador e atualiza a lista conhecida de nós.

3. DECISÕES DE IMPLEMENTAÇÃO

- **Uso de Goroutines**: Várias funções, como freeSuperNodes, listnerOtherNodes e receiveBroadcast, são executadas em goroutines para permitir processamento concorrente, melhorando a escalabilidade e a eficiência do sistema.
- **Uso de Mutex (mu)**: Para garantir que variáveis compartilhadas, como superNodes e files, sejam acessadas de maneira segura, o mutex sincroniza o acesso, evitando condições de corrida.

- **Transferência de Arquivos em Blocos**: No processo de download, os arquivos são transferidos em blocos de 1024 bytes, minimizando o uso de memória e reduzindo o risco de falhas durante a transmissão.

4. COMUNICAÇÃO ENTRE MÁQUINAS

O sistema é estruturado para garantir comunicação eficiente e segura entre os diferentes nós e clientes usando sockets TCP. Cada porta de comunicação possui uma função específica:

- **Cliente ↔ SuperNode**: Os clientes usam a porta supernoPort (8082) para registrar uploads e requisitar downloads ao SuperNode.
- **Cliente ↔ Cliente**: Cada cliente escuta na porta clientPort (8081) para atender a solicitações de download de arquivos diretamente de outros clientes.
- **SuperNodes e Coordenador**: A troca de informações entre SuperNodes e o coordenador ocorre em diferentes portas designadas para cada tipo de operação, incluindo registerPort para registro e electionPort para o processo de eleição.

5. TESTES REALIZADOS E ANÁLISE

Para validar o funcionamento do sistema, foram realizados testes abrangentes nas seguintes funcionalidades principais:

- **Registro de SuperNodes no Coordenador**: SuperNodes se conectam ao coordenador para realizar o registro. Após a confirmação do registro, recebem uma resposta "ACK". **Resultado**: Os SuperNodes foram registrados corretamente e liberados para comunicação com sucesso.
- **Liberação de SuperNodes para Comunicação**: Após o registro completo, todos os SuperNodes foram liberados para iniciar a comunicação entre si. **Resultado**: O coordenador enviou mensagens de liberação, permitindo a troca de informações entre SuperNodes e garantindo a continuidade do sistema.
- **Processo de Eleição para Coordenador**: Simulando uma falha no coordenador, um novo processo de eleição foi iniciado. SuperNodes com IDs maiores foram contatados, e um novo coordenador foi eleito. **Resultado**: A eleição não é concluída devido à obstrução da porta 8080.

- **Upload de Arquivo para o SuperNode**: Clientes enviaram arquivos para o SuperNode, que registrou o upload e adicionou o cliente ao mapa files. **Resultado**: O upload foi registrado corretamente e o SuperNode confirmou a operação com uma resposta positiva.
- **Download de Arquivo a Partir de Outro Cliente**: O cliente requisitou um download ao SuperNode, que identificou o IP de um cliente que possui o arquivo e forneceu a localização. **Resultado**: O arquivo foi transferido corretamente entre os clientes, e o processo foi concluído sem falhas.
- **Solicitações de Arquivos entre Clientes**: Clientes solicitaram diretamente a outros clientes um arquivo específico e o servidor de cliente atendeu à solicitação. **Resultado**: O arquivo solicitado foi encontrado e transferido entre os clientes com sucesso, demonstrando que o servidor do cliente responde corretamente a solicitações externas.

6. PRINTSCREENS DE EXECUÇÃO

Não foram coletados printscreens pois os testes foram feitos apenas no laboratório.

7. CONCLUSÃO

Este sistema distribuído de compartilhamento de arquivos proporciona uma estrutura robusta e tolerante a falhas para gerenciar o compartilhamento de arquivos entre clientes em uma rede de SuperNodes. A arquitetura permite ao sistema eleger um novo coordenador em caso de falha do nó mestre, garantindo a continuidade das operações. O uso de Go e a arquitetura concorrente possibilitam um sistema eficiente e escalável para redes distribuídas.

REFERÊNCIAS BIBLIOGRÁFICAS

Golang Documentation. **net.** Disponível em: <https://golang.org/pkg/net/>. Acesso em: 13 nov. 2024.

Golang Documentation. **sync.** Disponível em: <https://golang.org/pkg/sync/>. Acesso em: 13 nov. 2024.

Golang Documentation. **io.** Disponível em: <https://golang.org/pkg/io/>. Acesso em: 13 nov. 2024.

Golang Documentation. **bufio.** Disponível em: <https://golang.org/pkg/bufio/>. Acesso em: 13 nov. 2024.