Relatório Técnico: Sistema de Impressão Distribuída

Trabalho Prático 1 - Sistemas Distribuídos

Autores: Antonio Soares Couto Neto e Thales Matheus

---

1. Como Executar

Iniciar os serviços:

```
bash

docker compose up --build
```

Executar os testes:

```
bash

python test_cases.py
```

---

2. Visão Geral da Arquitetura

O sistema implementa um serviço de impressão distribuído utilizando o algoritmo de exclusão mútua de Ricart-Agrawala. A arquitetura é composta por:

- **Servidor de Impressão (Burro)**: Responsável apenas por receber e registrar as mensagens de impressão.
- **Clientes Inteligentes (3 instâncias)**: Implementam o algoritmo de exclusão mútua e coordenam o acesso ao recurso compartilhado.
- **Comunicação**: Utiliza gRPC para comunicação entre os componentes.

---

3. Algoritmo de Ricart-Agrawala

3.1 Princípio de Funcionamento

O algoritmo implementa exclusão mútua distribuída sem a necessidade de um coordenador central, baseando-se em:

1. **Relógios Lógicos de Lamport**: Cada mensagem carrega um timestamp lógico.

2. **Sistema de Prioridades**: O acesso é concedido com base nos timestamps (menor timestamp tem prioridade).
3. Troca de mensagens de requisição e liberação.

3.2 Implementação

Estruturas de Dados:

- `lamport_clock`: Contador lógico para ordenação de eventos.
- `request_queue`: Fila de requisições pendentes.
- `replies_received`: Conjunto de clientes que já responderam à requisição atual.

Protocolo de Comunicação:

- RequestAccess: Solicita acesso à seção crítica.
- ReleaseAccess: Libera o acesso após o uso.
- SendToPrinter: Envia mensagem para impressão.

---

4. Resultados dos Testes

4.1 Cenário 1: Funcionamento Básico

Objetivo: Verificar o fluxo básico de impressão sem concorrência.

Resultado:

- Cliente A solicitou e obteve acesso.
- Mensagem foi enviada para impressão com sucesso.
- Acesso foi liberado corretamente.

4.2 Cenário 2: Concorrência

Objetivo: Validar a ordem de acesso com múltiplos clientes.

Resultado:

- Cliente C obteve acesso primeiro (timestamp 1).
- Cliente A (timestamp 2) obteve acesso antes do Cliente B (timestamp 3).
- A ordem de execução respeitou os timestamps lógicos.

---

5. Dificuldades e Soluções

5.1 Comunicação entre Containers

Problema: Dificuldade na resolução de nomes entre containers Docker.

Solução: Configuração adequada da rede no docker-compose.yml e uso de portas expostas.

---

6. Conclusão

O sistema implementado demonstra os conceitos de exclusão mútua distribuída, utilizando o algoritmo de Ricart-Agrawala. A arquitetura permite a expansão para mais clientes e garante a ordem correta de acesso ao recurso compartilhado.