OS - TP 1 - PUC MINAS S1 2025

Edgard de Paiva Melo Filho

2 de maio de 2025

Repositório do Projeto

github.com/Kodvik/AcademicCode-OS-TP1

1 Introdução

O Algoritmo do Banqueiro, proposto por Dijkstra, é utilizado para prevenção de deadlocks em sistemas operacionais. conforme solicitado esse trabalho prático tem como objetivo implementar esse algoritmo usando linguagem C, dando um grande foco na utilização dos conceitos de programação concorrente e controle de acesso crítico com `pthread_mutex`.

2 Objetivos

- Simular clientes solicitando e liberando recursos de forma concorrente.
- Garantir que o sistema esteja sempre em estado seguro após cada requisição.
- Utilizar `mutex` para evitar condições de corrida no acesso às estruturas compartilhadas.

3 Bibliotecas Utilizadas

Alem das bibliotecas padrão do C, foram utilizadas as seguintes bibliotecas:

- `pthread.h` – Manipulação de threads e locks mutex.
- `unistd.h` – Controle de tempo (função `sleep`).
- `string.h` – Funções de manipulação de strings.

4 Descrição do Funcionamento

Ao iniciar o programa, os recursos disponíveis são definidos por linha de comando. O vetor `available[]` é inicializado com esses valores.

Em seguida, são criadas múltiplas threads de clientes. Cada cliente entra em um loop onde:

1. Gera uma requisição de recursos aleatória, respeitando sua necessidade máxima.
2. Chama a função `request_resources`, que:
   - Verifica se a requisição pode ser atendida.
   - Simula a alocação.
   - Verifica o estado de segurança do sistema com o algoritmo de segurança.
3. Após o uso, libera os recursos com `release_resources`.

5 Compilação e Execução

5.1 Compilação

Para compilar o projeto, utilize o comando `make`, que se apoia em um `Makefile` configurado para compilar automaticamente o código-fonte com `gcc` e suporte a `pthread`. Isso evita a necessidade de compilar manualmente com instruções longas como: (sei que não foi requisitado usar o makefile, mas como é só para automatizar o processo de compilação, mesmo não sendo de tudo necessário eu estou tentando criar o habito de usar makefile, resolvi fazer um então para facilitar um pouco).

```
gcc -pthread -o banqueiro src/main.c
```

5.2 Execução com Argumentos de Linha de Comando

Após a compilação, o programa é executado com:

```
./banker 10 5 7
```

Esses números representam a quantidade de instâncias disponíveis para cada tipo de recurso. No exemplo acima:

- `10` é o número de instâncias do recurso tipo 0;
- `5` é o número de instâncias do recurso tipo 1;
- `7` é o número de instâncias do recurso tipo 2.

Você pode passar qualquer outra sequência de números inteiros positivos, como:

```
./banker 8 3 4 2
```

Esse exemplo simula um sistema com quatro tipos de recursos.

Importante: É necessário passar pelo menos um argumento, e os valores devem ser inteiros positivos. Caso contrário, o programa exibirá uma mensagem de erro.

Para compilar o projeto, utilize o comando `make`.

A execução do programa é feita com:

```
./banker 10 5 7
```

Onde os números representam os recursos disponíveis para cada tipo.