Universidade Federal do Pará
Instituto de Ciências Exatas e Naturais
Faculdade de Computação
Laboratório de Sistemas Operacionais

Experiência 07

Nome: Alfredo Gabriel de Sousa Oliveira
Matrícula: 202004940020
Data: 25/01/2022

Introdução

Na experiência 07, foi implementado um algoritmo utilizando o conceito de pthread e mutex, no qual deveríamos simular o saque e o deposito. Além disso, foi pedido a analise de um algoritmo disponibilizado no arquivo da atividade.

O código fonte está disponível nesse link: https://github.com/alverad-katsuro/laboratorio_de_sistemas_operacionais

Etapas da experiência

Para a realização desta experiência primeiro foi feito a analise do algoritmo disponibilizado e após essa analise, foi desenvolvido o algoritmo solicitado seguindo as etapas descritas a seguir:

- Desenvolver de uma struct capaz de simular uma conta bancaria
- Criar uma função responsável por realizar o saque
- Criar uma função responsável por realizar os depósitos
- Criar uma função que retorna valores entre 100 e 500
- Criar a função main responsável por chamar as demais funções

Resultados

Para a primeira parte da atividade, foi concluído que esse algoritmo a cada instancia de thread, ou seja, a cada novo pthread soma o valor da constante ‘VECLEN’ ao valor total, isto é, a variável ‘sum’ presente na struct ‘DOTDATA’. Já para o controle desta soma foi utilizado o mutex para restringir a quantidade de acessos a variável ‘sum’ por

vez, um lock antes do acesso a fim de solicitar o recurso (variável) e um unlock após uso para liberar o recurso solicitado, de modo que não seja possível dois ou mais processos interagirem ao mesmo tempo com está variável que é compartilhada.

Já para segunda parte, após a implementação do algoritmo seguindo os passos supracitados e disponível no endereço acima, possui o comportamento esperado pela atividade.
A seguir serão demonstrados alguns exemplos de execução.

```
saldo inicial 10000.00
saldo apos o saque 9885.00
saldo apos deposito 10202.27
saldo apos deposito 10637.91
saldo apos deposito 10754.00
saldo apos o saque 10408.91
saldo apos o saque 10074.18
saldo apos deposito 10459.27
saldo apos o saque 10044.55
saldo apos deposito 10221.27
saldo apos o saque 10015.64
saldo final 10015.64
```

Como visto acima, o saldo inicial era R$10.000,00 e após as operações de saque e deposito o saldo final foi de R$10.015,64

```
saldo inicial 10000.00
saldo apos o saque 9779.91
saldo apos deposito 9985.45
saldo apos deposito 10136.09
saldo apos o saque 9777.45
saldo apos deposito 10142.55
saldo apos o saque 9799.73
saldo apos deposito 9988.73
saldo apos o saque 9782.64
saldo apos deposito 10152.18
saldo apos o saque 9872.18
saldo final 9872.18
```

Já para esta interação tivemos o saldo final foi de R$ 9872,18, ou seja, houve mais saques do que depósitos.

Conclusão

Com esse trabalho foi possível criar uma familiaridade maior com a implementação do mutex e do pthread. Ademais, é importante ressaltar que não houve problemas na utilização do mutex e do pthread visto que seus conceitos são bem objetivos, contudo, houve alguma dificuldade em implementar a função que gerava os números aleatórios, visto que inicialmente foi implementado o algoritmo com o saldo igual a zero, logo havia a necessidade de gerar um saque com valor menor ao saldo atual, e, havia interações em que o valor gerado

para ser sacado era maior que o saldo atual e mesmo chamando novamente a função que gerava o valor aleatório o “drand48” sempre gerava o mesmo valor.