

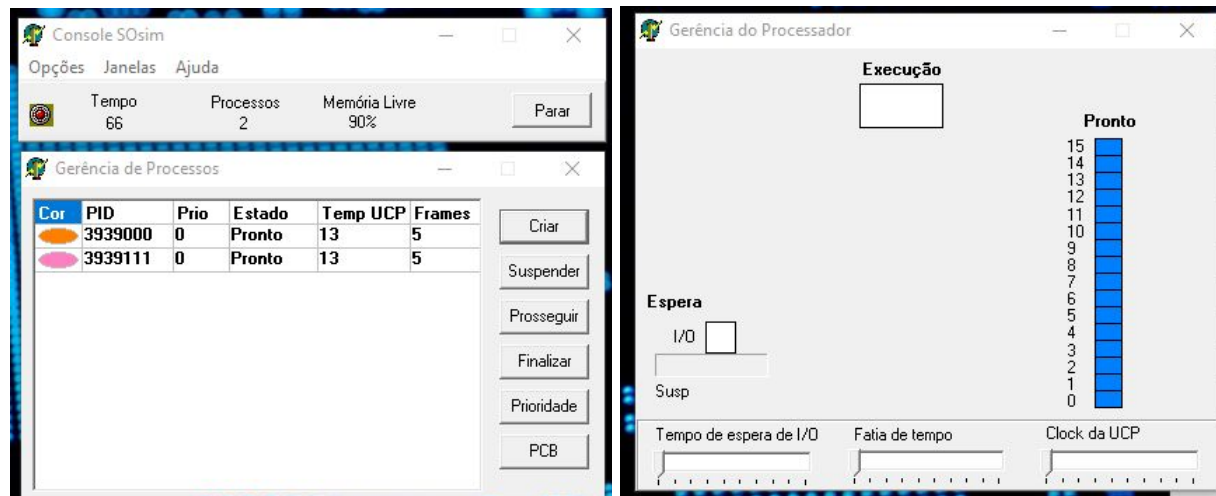
Luigi Muller Sousa Linhares

Sistemas Operacionais 2019.1

23 de maio de 2019

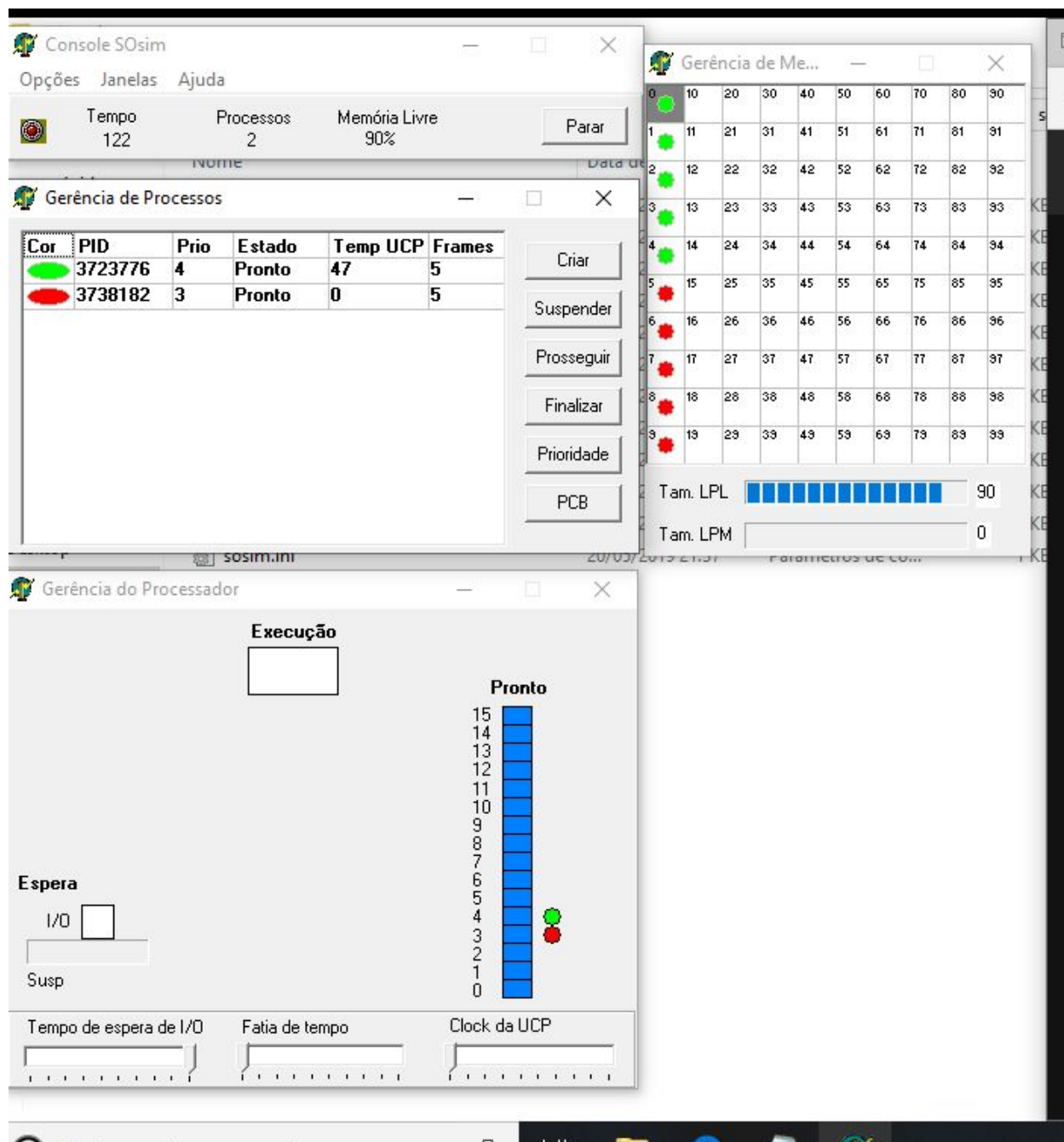
Laboratório de Sistemas Operacionais

PRÁTICA 01



Quando o processo que está em "Execução" termina (por exemplo, o processo rosa), ambos os processos ficam no estado "Pronto". O escalonador agora precisa escolher qual processo será executado agora, como o processo laranja é o próximo na fila de processos, ele é escolhido.

PRÁTICA 02



Ocorre starvation porque o processo verde tem uma prioridade maior que o outro fazendo com que ele sempre seja executado.

Uma ideia seria o escalonador reduzir a prioridade do processo a cada tique do relógio, até que sua prioridade se torne menor que de outro processo e a partir daí executar o processo seguinte. Outra ideia seria definir tempo máximo de execução e após esse tempo ser esgotado, o processos seguinte seria executado.

PRÁTICA 03

Qual o espaço de endereçamento real máximo de um processo?

Dependendo da arquitetura do processador, por exemplo, na arquitetura x86 o espaço máximo teórico é de 2 elevado a 32 endereços de memória e na arquitetura x64 o espaço máximo teórico é de 2 elevado a 64 endereços de memória sendo que alguns processadores limitam esse endereçamento para 48 bits.

Qual o espaço de endereçamento real mínimo de um processo?

O tamanho da página, porque é a menor parte transferida entre a memória física e disco.

Qual o tamanho da página virtual?

Por exemplo, 4 KB, 2 MB e 1 GB na arquitetura x86-64.

PRÁTICA 04

Quais os critérios utilizados pelo simulador para selecionar o processo a ser transferido para o arquivo de paginação (swap out)?

Se a página tiver o bit R como 0, ele se torna um candidato para a remoção, se houverem candidatos com esse critério é marcada para remoção o que tiver mais tempo sem ser referenciada. Em casos mais extremos, se todos forem referenciados no momento da escolha(ou seja, tem bit R como 1), é escolhida aleatoriamente uma para remoção.

Quando o processo deve ser transferido novamente para a memória principal (swap in)?

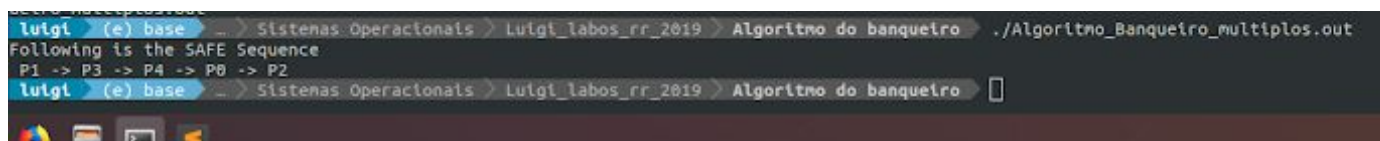
Como o simulador está em paginação por demanda, o processo só é chamado para a memória principal quando a CPU detecta a falta das página que esse processo demanda para executar suas instruções, a CPU gera uma interrupção para o sistema operacional carregar as páginas faltantes, após isso a CPU reassume o controle e reinicia a instrução que causou a interrupção.

QUESTÃO 02)

O algoritmo do banqueiro é usado para alocação de recursos de forma que a ordem de escalonamento seja segura e não ocorra impasses. O algoritmo analisa se a solicitação do processo por um recurso levará o sistema para um estado inseguro, e possivelmente para um estado de impasse. Se ocorrer, a solicitação é negada, caso contrário, é aceita e levada a diante. O algoritmo vem da analogia de um banqueiro que possui uma linha de crédito para os seus clientes, sendo que ele somente oferecerá o crédito se puder oferecer também para as demais pessoas.

Implementação do algoritmo do banqueiro para múltiplos recursos

O algoritmo do banqueiro foi compilado com o gcc versão 6.3.0 do MinGW. O arquivo se encontra na pasta "Algoritmo do banqueiro", o arquivo com código se chama "Algoritmo_Banqueiro_multiplos.cpp" e o executável para Linux se chama "Algoritmo_Banqueiro_multiplos.out".



```
luigi (e) base ~ > Sistemas Operacionais > Luigi_labos_rr_2019 > Algoritmo do banqueiro > ./Algoritmo_Banqueiro_multiplos.out
Following is the SAFE Sequence
P1 -> P3 -> P4 -> P0 -> P2
luigi (e) base ~ > Sistemas Operacionais > Luigi_labos_rr_2019 > Algoritmo do banqueiro
```

QUESTÃO 03)

O algoritmo do barbeiro é um problema relacionado em sistemas operacionais sobre a comunicação entre processos e regiões críticas. O problema é sobre um barbearia que possui um barbeiro, uma cadeira para o barbeiro e n cadeiras de espera para os clientes. O barbeiro analisa se há clientes nas cadeiras, se não houver, ele dorme. Caso contrário, o cliente o acorda. Se o barbeiro estiver cortando o cabelo de um cliente, ele espera na cadeira, caso não haja, ele vai embora.

Uma solução seria usar três semáforos:

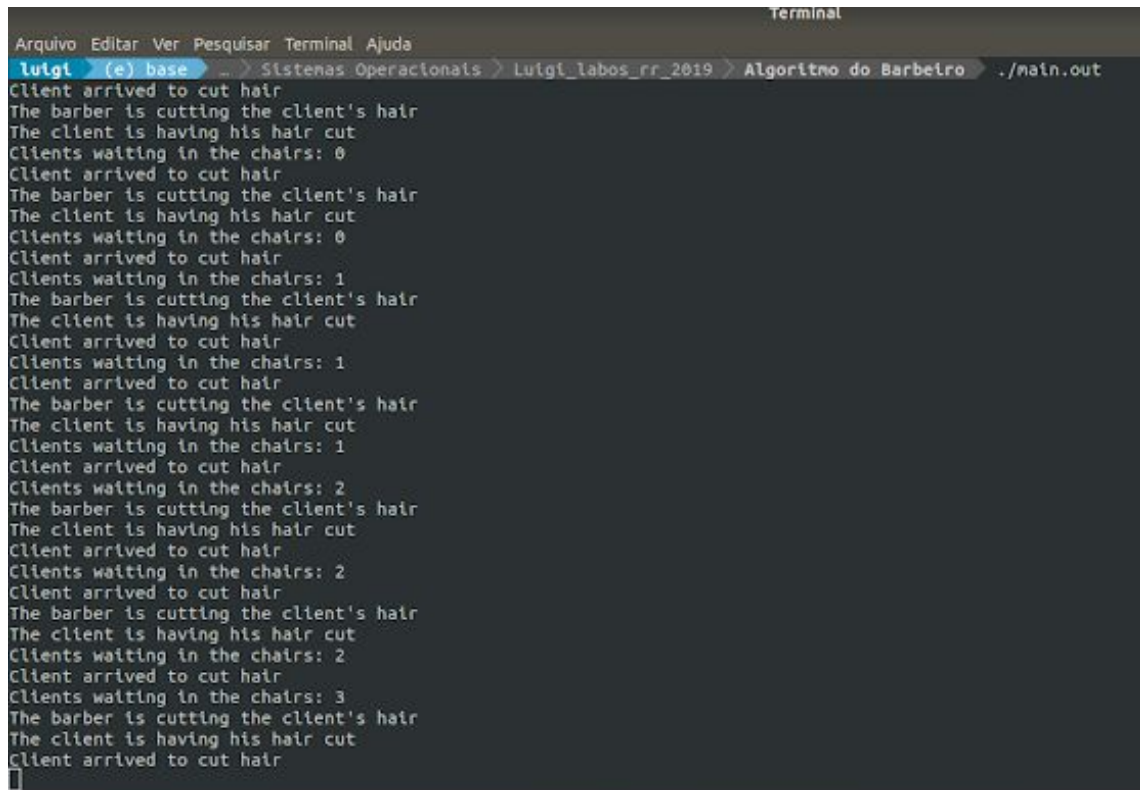
1. Um para o cliente que contará o número de clientes presentes;

2. Um semáforo binário para o barbeiro sendo 0 para "dormindo" e 1 para trabalhando;
3. Um para a exclusão mútua que é necessária para o processo executar.

Nessa solução é registrado também o número de consumidores que estão esperando na barbearia.

Implementação do algoritmo do barbeiro

O algoritmo do barbeiro foi compilado com o gcc versão 6.3.0 do MinGW. O arquivo se encontra na pasta "Algoritmo do Barbeiro", o arquivo com o código se chama "main.cpp" e o executável para Linux se chama "mian.out". Não há necessidade de entradas.



```
Arquivo Editar Ver Pesquisar Terminal Ajuda
luigi (c) base > > Sistemas Operacionais > Luigi_labos_rr_2019 > Algoritmo do Barbeiro ./main.out
Client arrived to cut hair
The barber is cutting the client's hair
The client is having his hair cut
Clients waiting in the chairs: 0
Client arrived to cut hair
The barber is cutting the client's hair
The client is having his hair cut
Clients waiting in the chairs: 0
Client arrived to cut hair
Clients waiting in the chairs: 1
The barber is cutting the client's hair
The client is having his hair cut
Client arrived to cut hair
Clients waiting in the chairs: 1
Client arrived to cut hair
The barber is cutting the client's hair
The client is having his hair cut
Clients waiting in the chairs: 1
Client arrived to cut hair
Clients waiting in the chairs: 2
The barber is cutting the client's hair
The client is having his hair cut
Client arrived to cut hair
Clients waiting in the chairs: 2
Client arrived to cut hair
The barber is cutting the client's hair
The client is having his hair cut
Clients waiting in the chairs: 2
Client arrived to cut hair
Clients waiting in the chairs: 3
The barber is cutting the client's hair
The client is having his hair cut
Client arrived to cut hair
```

O programa inicia com o cliente entrando para cortar o cabelo, logo, ele acorda o barbeiro e já tem o seu cabelo cortado. Com o tempo começa a se acumular clientes nas cadeiras, como aparece no final da imagem.



```
Atividades Terminal ▾
Arquivo Editar Ver Pesquisar Terminal Ajuda
The barber is cutting the client's hair
The client is having his hair cut
Client arrived to cut hair
Clients waiting in the chairs: 3
Client arrived to cut hair
The barber is cutting the client's hair
The client is having his hair cut
Clients waiting in the chairs: 3
Client arrived to cut hair
Clients waiting in the chairs: 4
The barber is cutting the client's hair
The client is having his hair cut
Client arrived to cut hair
Clients waiting in the chairs: 4
Client arrived to cut hair
The barber is cutting the client's hair
The client is having his hair cut
Clients waiting in the chairs: 4
Client arrived to cut hair
Clients waiting in the chairs: 5
The barber is cutting the client's hair
The client is having his hair cut
Client arrived to cut hair
Clients waiting in the chairs: 5
The client gave up (the hall is full)
The barber is cutting the client's hair
The client is having his hair cut
Clients waiting in the chairs: 4
Client arrived to cut hair
Clients waiting in the chairs: 5
The barber is cutting the client's hair
The client is having his hair cut
Client arrived to cut hair
Clients waiting in the chairs: 5
```

Aqui acontece quando todas as cadeiras são ocupadas e o cliente desiste. Após isso, o barbeiro continua cortando cabelo, até um momento que uma cadeira fica vazia e outro cliente a ocupa.

Trabalhos citados

ALVAREZ, Diego. **O Problema do Barbeiro Dorminhoco (com threads)**. Disponível em:

<http://ces33.blogspot.com/2009/05/o-problema-do-barbeiro-dorminhoco-com_07.html>.

Operating System | Sleeping Barber problem. Disponível em:

<<https://www.geeksforgeeks.org/operating-system-sleeping-barber-problem/>>.

Operating System | Banker's Algorithm. Disponível em:

<<https://www.geeksforgeeks.org/operating-system-bankers-algorithm/>>