Laboratório de Sistemas Operacionais - Trabalho 4

Prof. Sérgio Johann Filho

Objetivos

- Compreender o escalonamento de processos no Linux.
- Implementar uma aplicação *multithread* e avaliar o seu comportamento em ambientes mono e multiprocessados.
- Configurar diferentes políticas e prioridades de execução.

Descrição

Este trabalho consiste na implementação de um programa multithread chamado *thread_runner*, que cria um número específico de threads e aloca um buffer global de tamanho especificado. Um ponteiro global é configurado para o início do buffer e cada thread terá um caractere atribuído a ela (exemplo, A, B, C, D...). Cada thread deverá escrever seu caractere na posição corrente do ponteiro global e incrementá-lo (o acesso ao buffer e seu ponteiro deve ser sincronizado com um semáforo binário, compartilhado entre threads). Para evitar que o buffer seja preenchido rápido demais, utilize a função *nanosleep()* com um atraso de 1ms ou menos (experimente com os valores). O uso dessa função não causa troca de contexto (espera ocupada) e evita que as threads fiquem o tempo todo em sua seção crítica. Desta forma, no final da execução, o buffer global irá representar o padrão de execução de cada thread.

O programa deverá imprimir o buffer ao final da execução e apresentar a sequência de execução (ABCD, por exemplo), juntamente com a contagem de quantas vezes cada thread foi escalonada durante o tempo de execução. Abaixo é apresentado um exemplo de execução esperado como saída do seu programa:

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCDDDDDDDDDDDDDDDDDDDDDDDDD
DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
DDDDDDDDDDDDDDDD

ABCD

A = 1
B = 1
C = 1
D = 1
```

O programa que você irá desenvolver deverá ter como formato de entrada os parâmetros:

```
$ ./thread_runner <número_de_threads> <tamanho_do_buffer_global_em_kb>
   <politica> <prioridade>
```

Testes

Verifique como configurar políticas e prioridades para threads nos exemplos das Aulas 18 e 19. Deverão ser executados testes no emulador QEMU utilizando um único core (execução padrão) e quatro cores (passar a flag *-smp 4* ao QEMU).

Entrega

Você deve utilizar o programa desenvolvido para escrever um relatório incluindo padrões de execução, figuras e descrições que mostram a comparação da execução com diferentes algoritmos de escalonamento, incluindo o CFS (SCHED_OTHER, SCHED_BATCH e SCHED_IDLE) e escalonadores de tempo real (SCHED_DEADLINE, SCHED_RR e SCHED_FIFO). É importante que você leia a respeito de cada classe de escalonamento no Linux e as políticas associadas, e mostre, através de experimentos, que os algoritmos se comportam de acordo com o esperado. Compare a saída do seu programa com o padrão de escalonamento real, que pode ser obtido com o uso das ferramentas *trace-cmd* e *KernelShark*.

Este trabalho deverá ser realizado em duplas ou trios e apresentado no dia 25/06 (apresentação em torno de 5 a 7 minutos). O relatório deverá ser utilizado como roteiro para apresentação. Para a entrega, é esperado que apenas um dos integrantes envie pelo Moodle um arquivo *.tar.gz* do projeto com o nome dos integrantes, contendo:

1. Programa desenvolvido para os experimentos;
2. Relatório;