Primeiro Trabalho de Sistemas Operacionais – INF 1316 / 2019_2

O primeiro trabalho consiste em programar em linguagem C programas que implementem três **interpretadores de comandos** e **escalonadores de programas**. O interpretador deverá ler a especificação do escalonamento de vários processos concorrentes e solicitar ao escalonador a execução dos mesmos. Os interpretadores e escalonadores serão processos Unix de programas em linguagem C seus.

O seu Interpretador deverá ser um processo Unix diferente do escalonador, e como eles precisam se comunicar, você deve usar algum tipo de comunicação inter-processo (shared memory, pipes, Named pipes ou Message Queues) para passar os dados dos programas a serem escalonados para o Escalonador.

O Escalonador então inicia e passa a controlar a execução de todos os processos simultaneamente de acordo com a política de escalonamento. Cada escalonador deverá ter uma das seguintes diretrizes de escalonamento:

1) PRIORIDADE (prioridade de execução, um número de 0 a 7 sendo 0 a maior prioridade e 7 a menor)
2) ROUND-ROBIN (com fatia de tempo de 1 segundo)
3) REAL-TIME (nesse caso o interpretador deve indicar ao escalonador em qual momento deve ser iniciado e por quantos segundos deve executar)

No sistema Unix o escalonador executa como parte do núcleo (kernel) mas esse escalonador que você irá implementar no trabalho (Escalonador) é um simples processo que vai controlar e coordenar a execução dos processos lidos pelo interpretador de comandos. Ou seja, o seu escalonador vai gerenciar a ordem de execução dos programas utilizando os **sinais** para controlar quando devem ser suspensos e quando devem prosseguir com a sua execução

Cada Escalonador deve ser um programa capaz de lidar com cada uma das 3 políticas acima, ou seja, deverá implementar as estruturas de dados para tal gerenciamento.

Na política de escalonamento REAL-TIME cada processo deverá executar periodicamente (uma vez por minuto), iniciando sua execução, em determinado momento de tempo (I) e deve permanecer executando apenas durante um certo período de tempo (D). Tanto o momento de início como a duração da execução no minuto, são indicados pelos seus parâmetros. Por exemplo, P1 I=20 D=5, indica que P1 deve ser executado a cada minuto e 20 segundos e deve executar por 5 segundos, ou seja, do segundo 20 até o segundo 25, dentro de cada minuto.

Atenção: Para esse escalonamento REAL-TIME o seu escalonador deverá verificar se um novo processo iniciado não possui parâmetros de escalonamento conflitantes com os dos demais processos já em execução. Por exemplo, se seu sistema já está executando P1 I=0 D=10 e P2 I=20 D=5, então não será possível executar um processo P3 I=11 D=20, pois não haveria como permitir a execução simultânea de P2 e P3. Você também deverá verificar se I+D <= 60 segundos, pois o período de execução de um processo não deve ir além do início do início do próximo minuto.

Além disso, como é muito comum no processamento de tempo real, pode existir também *uma dependência causal entre processos*, ou seja, um determinado processo, P4 só inicia quando outro, digamos P2, tiver terminado. Nesse caso, especifica-se o processo sucessor (P4) assim: P4 I=P2 D=10. Então, o escalonador precisa garantir **que entre a execução de P2 e de P4 nenhum outro processo execute**.

O interpretador vai ler um comando (1 processo) por linha de um arquivo **exec.txt** (ASCII). A sintaxe dos comandos a ser analisada pelo interpretador é a seguinte:

> **Run** <nome_programa> PR=<número de 0 a 7>, para escalonamento PROPRIDADE
>
> **Run** <nome_programa>, para o ROUND-ROBIN
>
> **Run** <nome_programa> I=<momento-início> D=<tempo-duração>, para REAL-TIME

Onde: momento-início == inteiro em [0-59] OU <nome_programa> já declarado em Run anterior.

Os processos a serem escalonados/ controlados devem ser de dois tipos: CPU-bound e IO-bound que devem ser criados por você. O interpretador irá ler de **exec.txt** quais são os programas a serem executados, e deverá iniciá-los exatamente na ordem em que aparecem nesse arquivo, **com um intervalo de 1 segundo entre cada um deles**. Ou seja, não será possível ler toda a entrada e ordenar a lista antes de passar para o escalonador. Você poderá assumir que processos causalmente dependentes sempre virão aos pares (exemplo p2 e P4) e nunca mais do que isso, e que P2 sempre aparecerá antes de P4 em exec.txt. Os programas devem ser passados um-a-um para o escalonador. A saída deve ser clara o suficiente para mostrar como e porque ocorre a preempção na execução dos processos. Esses arquivos serão objeto de avaliação.

O trabalho pode ser feito por grupos de até 3 alunos (são 3 programas). O trabalho será avaliado em **22/10,** na aula de laboratório, através de entrevista do grupo e deve ser enviado por Email até **21/10 às 23:59:59** para: luizfernandobessaseibel@gmail.com com subject indicando INF1316 – SO - TRAB1 e os nomes dos integrantes do grupo. Cada dia de atraso acarreta um desconto de 1 ponto na nota máxima. Devem ser entregues os códigos fonte e um relatório indicando os programas que serão executados em seu teste, a ordem de entrada para o escalonador e a ordem de execução determinada pelo escalonador, juntamente com uma análise crítica sobre o que, de fato, ocorreu. Essa explicação também será objeto de avaliação e deverá ser feita para cada interpretador/escalonador.

OBS: Cabe informar que haverá avaliação automática de cópias de programas ou de trechos de programas. Caso seja verificada uma cópia, os programas iguais receberão grau zero.