Estudo Complementar para aproveitamento da Disciplina de Sistemas Operacionais.

O aluno Wagner Loch, do curso de Bacharelado em Engenharia de Computação, deverá realizar as duas atividades descritas na sequência para complementar o pedido de aproveitamento solicitado.

Prazo para entrega: 25 de novembro de 2019

Gerson Cavalheiro
2019/2

Atividade 1: Pesquisa Bibliográfica

Entrada e Saída

Descrever, na forma de um artigo 3 a 4 páginas, os mecanismos de Entrada e Saída promovidos pelo Plattform Controller Hub e seu princípio de operação.

Atividade 2: Implementação

Escalonador de Processos

Especifique e implemente um simulador do mecanismo de escalonamento de processos do tipo multifilas, com as filas representando processos com diferentes prioridades. Em uma fila, os processos executam em round-robin, havendo preempção entre as filas. Este simulador possui como entrada quatro parâmetros:

- O número de CPUs disponíveis
- Duração do *slice* de CPU entregue a cada processo
- A quantidade de memória disponível (em GB)
- O nome de um arquivo com a descrição dos processos que serão executados

O modelo de simulador deve implementar "simulação discreta" e manter um relógio que contabilize o avanço do tempo. Nesta implementação, a unidade de tempo é o *slice*.

O arquivo de descrição dos processos descreve os processos que serão executados, sendo um processo descrito a cada linha, da seguinte forma:

```
chegada, duração, memória, prioridade
```

Onde:

- chegada: representa o tempo em que o processo é recebido pelo SO
- duração: número de slices necessários para executar o processo
- memória: quantidade de memória (em GB) necessário à execução do processo (use apenas múltiplos de 64 GB)
- prioridade: um valor entre 0 (zero) e 4, sendo 0 a mais alta prioridade

Assim, no exemplo:

```
0, 10, 128, 1
5, 35, 256, 1
7, 50, 64, 0
10, 20, 128, 2
```

São descritos 4 processos, que chegam nos tempos 0, 5, 7 e 10, respectivamente, sendo que a duração do primeiro é de 10 slices, ocupando 128 GB de memória e tendo prioridade 1. Já o segundo, tem duração de 35 slices e consome 256 GB de memória, também com prioridade 1.

Importante: os processos com prioridades 1, 2 e 3 tem prioridades variáveis. Assim, após passados 10 slices de execução com sua prioridade normal, o processo tem sua prioridade decrementada em uma unidade, até atingir o valor mínimo (que é 4). Então passa a incrementar uma unidade, a cada 10 slices, até atingir seu valor inicial. Neste modelo de prioridades, processos com a prioridade mais baixa (4) não se alteram.

A quantidade de memória livre deve ser considerada no lançamento de um processo. Caso não haja memória livre, ele deverá ser executado assim que a quantidade de memória necessária estiver disponível, não havendo nenhum outro processo pronto para ser executado com prioridade maior que a sua.

Como resultado, o programa deve fornecer um arquivo que descreva a execução dos processos com o seguinte formato:

```
chegada, lançamento, duração projetada, duração observada
```

Onde:

- chegada: representa o tempo em que o processo foi recebido pelo SO
- lançamento: tempo em que o processo foi lançado efetivamente (se igual à chegada, não houve atrasos)
- duração projetada: é a duração informada no arquivo de entrada para o processo
- duração observada: contabiliza o tempo necessário para execução do processo

```
0, 0, 10, 10
5, 5, 35, 35
7, 7, 50, 69
10, 12, 20, 27
```

Entrega, um arquivo zip (em arquivos em outros formatos serão desconsiderados):

- Os fontes do programa, devendo o programa estar comentado de forma a ser possível compreender a implementação;
- Um PDF descrevendo a implementação (sem reproduzir o código, apenas trechos se acharem relevante na explicação) e realizando uma análise do desempenho da execução dos processos (diferença entre as durações projetadas e observadas e diferença entre chegada e lançamento), considerando diferentes quantidades de CPUs e memória disponível.