Escola de Artes, Ciências e Humanidades

ACH2044

Sistemas Operacionais

Relatório EP 1

Autores:
Bernardo Chagas Araújo — 10367210
Daniel Bolsonaro Vaz Filho — 10346867
Leonardo Soares Santos — 10284782

5 de Novembro de 2018

Conteúdo

Lista de Figuras

1 Código Fonte

1.1 Pacote Sistema

O pacote Sistema engloba todos os arquivos de um Sistema Operacional no que se diz respeito a execução de uma lista de processos que são dados em forma de batch (Lote de Dados), seus componentes são o BCP (Bloco de Controle de Processos), o comparaBCP (que é complemento do Escalonador), a CPU (que executa os procesoss), o Despachante (faz os pedidos de salvamento e retomada de contexto), o Escalonador em si e o SO, que age como agente regulador do sistema.

Todas as classes estão encapsuladas com um sistema de get e set, protegendo assim a integridade dos dados dentro de um Sistema Operacional.

1.1.1 BCP

No BCP estão todas as estruturas responsáveis por atuar em um controlador de processos, além de seus 2 registradores rotulados como X e Y. Os únicos métodos dessa classe são os Gets e Sets de todos os atributos.

1.1.2 comparaBCP

Nos vimos diante do problema para organizar os processos que entravam com os que já estavam na fila de processos prontos, então decidimos usar a classe PriorityQueue do java, que requer um método comparador método complementar para organizar a ordem da fila prioritária, que é, basicamente esse método em si.

1.1.3 CPU

Classe em que ocorre a execução do processo e interpretação dos comandos que são passados pelo processo. Na CPU também se encontram informações como o contador de programa, o quantum, o nome do processo e coisas do tipo sobre o processo que está dentro da CPU.

1.1.4 Despachante

Na classe despachante você encontra chamadas as funções para salvar e atualizar contexto dentro da CPU e dentro de processos

1.1.5 Escalonador

É na classe Escalonador que ocorre a seleção dos processos que devem ser rodados e da lista de bloqueio. O método carregaProcessos() cria o log de carregamento dos processos na ordem em que eles devem ser processados. O método escolheProximo(), como o nome sugestivo propõe, carrega a lista de processos prontos e pega o primeiro da fila, analisa se ele é válido para rodar o programa e, se for, é submetido à CPU para análise de conteúdo. Caso ele não seja, significa que ele tem crédito 0 e é preciso que a lista de bloqueio rode, para certificar de que nenhum processo dentro dessa lista possúi algum crédito para rodar. Se depois que a lista de bloqueados for esvaziada não tivermos nenhum processo com crédito maior que 0, aí resetamos os créditos de todos os processos.

1.1.6 SO

O SO basicamente faz as chamadas para o Logger e o Escalonador para rodar os processos e escrever o arquivo de log, além disso, ele faz a restauração de créditos da lista de processos.

1.2 Pacote Suporte

1.2.1 CriaTeste

Como o nome sugere, é uma classe que cria testes automatizados de acordo com os problemas do enunciado, ele cria diversos arquivos de processos, um arquivo de prioridades e um arquivo de quantum dentro de uma pasta e roda o programa de escalonador naquela pasta. Foi muito utilizado na criação de casos de teste para construção de gráficos e análise de quantum ideal.

1.2.2 Leitor

O leitor nada mais é do que a classe de leitura criada externamente ao SO pois, teoricamente, não faz parte do sistema em si. Ele é utilizado para ler todos os

arquivos de entrada dos casos de teste e tem nele, todas as leituras para poder rodar os processos com perfeição.

1.2.3 Logger

A classe Logger usa da classe PrintWriter do java para poder escrever e acessar o arquivo para escrita dentro da pasta de log. Nele estão todas as funções que ativamente e diretamente escrevem algo no log.

2 Resultados Obtido

2.1 Teste com processos dados pelo professor

2.1.1 Resultado com base nas orientações do professor

Com base nas orientações dadas pelo professor em sala de aula, via email e no enunciado do trabalho. O calculo da média de trocas(1) e Media de Instrução(2) são:

$$MédiadeTrocas = \frac{NúmerodeTrocas}{NúmerodeProcessos} \tag{1}$$

$$MediadeInstruções = \frac{NúmerodeInstruçõesTotais}{QuantaReservado} \tag{2}$$

Quantum	Média de Trocas	Média de Instruções
1	12,2	0,9
2	6,9	1,571429
3	5,4	2,041237
4	4,8	2,224719
5	4,1	2,712329
6	3,8	2,869565
7	3,7	2,955224
8	3,5	3,142857
9	3,5	3,142857
10	3,5	3,142857
11	3,4	3.193548
12	3,4	3.193548
13	3,4	3.193548
14	3,4	3.193548
15	3,3	3.3
16	3,3	3.3
17	3,3	3.3
18	3,3	3.3
19	3,3	3.3
20	3,3	3.3
21	3,3	3.3

Tabela 1: Resultados: Processos dado pelo professor

Com base na tabela 1 e no gráfico da figura 1, o quantum ideal é 15 pois, com esta quantidade de quanta, a média de instruções e a média de trocas tem pelo menos o mesmo valor. A partir disso é possível ver que o valor de quantum não altera nenhuma das variáveis, também é visível que os testes foram feitos apenas até 21 quanta, pois no enunciado do trabalho havia a condição de que os processos teriam apenas 21 instruções, limitando assim a quantidade de quanta possível para resolução dos casos teste. Todavia, houve testes disponibilizados pelo professor tinha 24 de quantum e, apesar disso, não obtivemos variação de quaisquer médias. Por isso decidimos não colocar esse resultado com os outros.

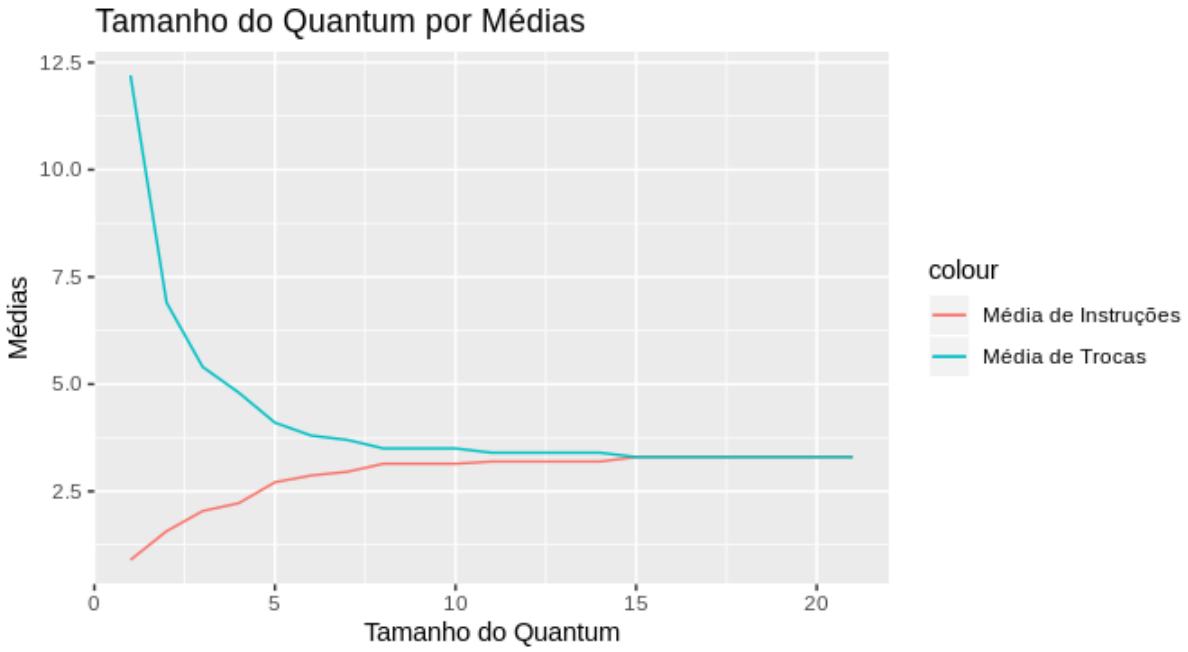

Figura 1: Gráfico de Médias por Quantum

2.1.2 Com base no Log do Monitor

Foi impossível realizar testes com os logs do monitor devido a inconsistência de informações. Nos foi fornecido vários arquivos de logs ao longo de uma semana e devido ao tempo de entrega e diversas alterações pendentes, optamos por não reproduzir o resultado desses testes.

2.2 Testes criados

2.2.1 Manualmente de acordo com o professor

Quantum	MediaTrocas	MediaIntrucoes
1	9.9	0.9430379746835443
2	5.8	1.568421052631579
3	4.6	2.041095890410959
4	4.1	2.257575757575758
5	3.7	2.4833333333333334
6	3.3	2.98
7	3.3	2.98
8	3.2	3.0408163265306123
9	3.2	3.0408163265306123
10	3.2	3.0408163265306123
11	3.2	3.0408163265306123
12	3.2	3.0408163265306123
13	3.2	3.0408163265306123
14	3.1	3.1702127659574466
15	3.0	3.3863636363636362
16	2.9	3.5476190476190474
17	2.9	3.5476190476190474
18	2.9	3.5476190476190474
19	2.9	3.5476190476190474
20	2.9	3.5476190476190474
21	2.9	3.5476190476190474

Tabela 2: Resultados: Processos criados manualmente de acordo com o padrão de processos

Os resultados obtidos nesta parte corroboram os resultados da seção 2.1.1, pois o valor ideal neste exemplo é 14, muito próximo do valor obtido na sessão citada. Logo podemos tomar o valor de quantum ideal sendo entre 14 e 16.

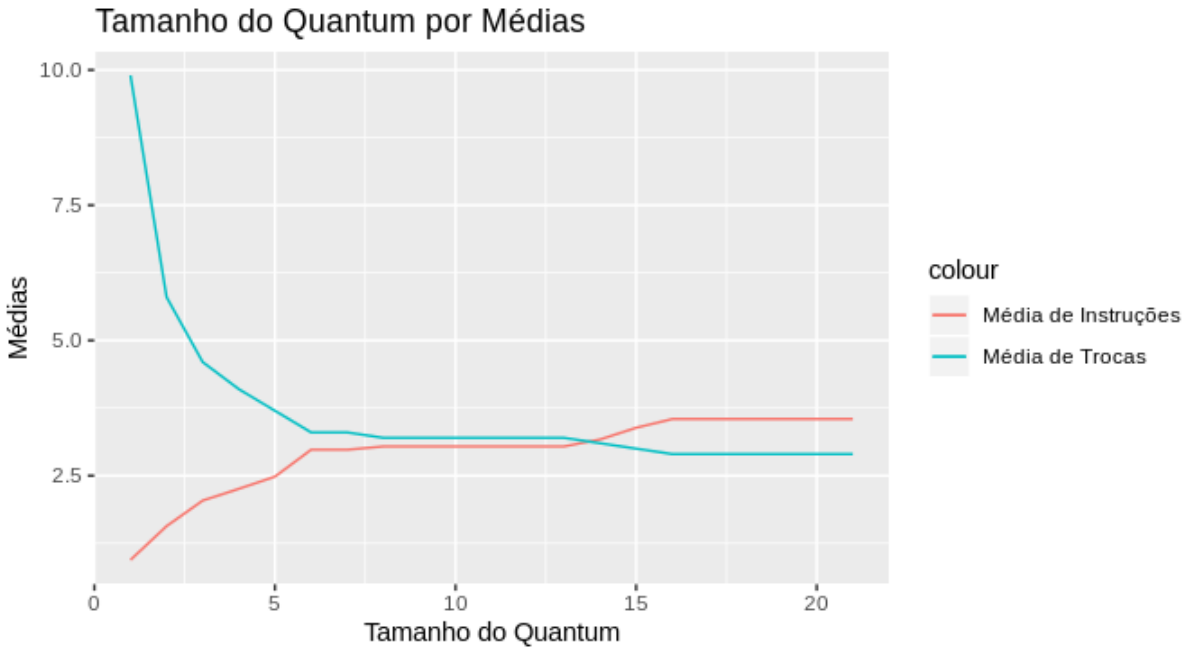

Figura 2: Gráfico: Resultado obtido de teste próprios

3 Conclusão

De acordo com os resultados obtidos em todos os testes feitos, chegamos a conclusão de que o quantum ideal para processos de até 21 instruções está entre 14 e 16.

4 Observações finais

Os casos testes automatizados não obtiveram resultados representativos o bastante para o restante do problema por isso foram omitidos.

Tentamos deixar o sistema o mais otimizado possível e com todos os critérios pedidos, todavia, diante de fatos de mudança estrutural do problema e atraso na divulgação de gabaritos de casos teste, é possível que alguns resultados estejam incompatíveis com o pedido