Sistemas Operacionais

Relatório do Trabalho: Escalonador de Processos

Prof. Gerson Cavalheiro

Thiago Heron Albano de Ávila

1. Descrição do Trabalho

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. Além disso, vale ressaltar algumas observações que devem ser respeitadas.

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.

2. Dados de Entrada

A configuração dos dados de entrada do programa é passada através do terminal ao realizar a execução. Abaixo, estão descritas os dados necessários e o formato de execução.

- **NumberCPU:** O número de CPUS disponíveis.
- **SizeMemory:** A quantidade de memória disponível (em GB), múltiplos de
- **NameFile:** O nome do arquivo que contém a descrição dos processos que serão executados. Neste trabalho, utilizou-se o arquivo com o nome **"processos.txt"** , sendo que é formatado onde um processo é descrito a cada linha da seguinte forma:
    - chegada, duração, memória, prioridade

Logo abaixo, está descrito o formato da linha de comando para a execução do trabalho com os parâmetros de entrada:

./main NumberCPU MemorySize NameFile

./main 2 512 processos.txt

3. Dados de Saída

Os dados de saída são escritos em um arquivo chamado **"results.txt"** que descreve 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 S.O.
- **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

4. Configurações do Trabalho

O seguinte trabalho foi implementado através de algumas listas encadeadas que representam as filas de prioridades executando em *round-robin*, ou seja, possui cinco listas que representam as prioridades de 0 à 4, além de uma outra lista para auxiliar no controle dos processos que não foram executados devido a falta de memória.

Partindo do pressuposto que este trabalho é um simulador de escalonamento de processos, paras controlar a unidade de tempo (*slice*) foi utilizado uma estrutura de repetição que reproduz a execução do número de CPUs determinada, após a execução destas, é incrementado um na unidade de tempo.

5. Funcionamento da Programa

Em cada uma das repetições para controlar o número de CPUs executadas por unidade de tempo, é chamado a função denominada **void CPU(),** essa tem a finalidade de encontrar qual o processo deve ser executado, desde qual tem preferência e até a checagem se houve algum processo anterior que deveria ter sido executado dado a falta de memória.

Após a escolha do processo ideal para a execução, é feito a atualização do número de slices do processo através da função **updateProcess,** e também os recursos do sistema, como a memória, através da função **updateSystemResources.**

Por fim é verificado se o processo em análise ainda possui *slices*, e dependendo da situação, ou é finalizado o processo, ou é verificado se deve ocorrer uma troca de prioridade caso sua prioridade inicial tenha sido definido entre 1 e 3 inclusives, e tenha passado 10 slices na sua prioridade atual.

6. Análise de Desempenho da Execução dos Processos

Para a realização das análises da execução dos processos será considerado um arquivo de entrada com variações de memória e CPUs, identificando a relação com a unidade de tempo (*slices*) total para ser concluída, e processos bloqueados devido a falta de memória para a execução de processos em um *slice.*

Tabela 1. Dados de entrada referente ao arquivo com descrições de processos denominado “processos.txt”.

Chegada	Duração	Memória	Prioridade
0	5	128	0
5	4	64	0
7	3	256	1
8	5	256	1
10	4	64	2
12	3	256	2

6.1 Comparação do Impacto de Número de CPUs

Essa análise foi feita comparando o programa com 1 CPU e 4 CPUs, consequentemente com tamanho de memória fixa, e observado o número total de unidade de tempo para a execução.

Na **Tabela 2**, é observado com apenas 1 CPU, onde pode-se observar nos primeiros processos que apesar de houver memória disponível, há uma diferença do tempo de chegada em relação ao lançamento devido ao fato que apenas um *slice de processo* é consumido por uma CPU em *slice de tempo*.

Em relação a **Tabela 3,** onde é testado com 4 CPUs, pode-se observar que como há a disponibilidade de mais CPUs para executar *slices de processos* na unidade de tempo, é possível lançar os processos sem atraso desde que haja memória disponível, que é o caso.

Pode-se observar também que a duração observadas é coerente, tomando como base a **Tabela 3,** a duração observada é exatamente o tempo de lançamento adicional a quantidade de slices (duração projetada) necessária para a execução, não tendo atrasos.

Tabela 2. Dados de Saída do programa com as entradas, **Memória: 1024 GB, Número de CPUs: 1, Tempo Total: 24 slices.**

Chegada	Lançamento	Duração Projetada	Duração Observada
0	0	5	5
5	5	4	9
7	9	3	14
8	10	5	17
12	18	3	23
10	17	4	24

Tabela 3. Dados de Saída do programa com as entradas, **Memória: 1024 GB, Número de CPUs: 4, Tempo Total: 15 slices.**

Chegada	Lançamento	Duração Projetada	Duração Observada
0	0	5	5
5	5	4	9
7	7	3	10
8	8	5	13
10	10	4	14
12	12	3	15

6.2 Comparação do Impacto do Tamanho de Memória

Essa análise foi feita comparando o programa com 256 GB de memória e 512 GB, além disso foi utilizado 2 CPUs para ambos os testes, a fim de explorar a execução de mais de um processo na mesma unidade de tempo impactando consequentemente na memória.

Na **Tabela 1,** pode-se observar que os processos com tempo de chegada em 7 e 8 possuem 256 GB para execução, consequentemente no primeiro teste que pode ser visualizado na **Tabela 4,** não será possível executar ambos na mesma unidade de tempo devida a falta de memória. Como consequência disto, o segundo processo a ser executado será bloqueado. Isso pode ser analisado onde o processo com chegada em 7 apenas consegue ser lançado após a finalização do processo 8 que ocupava toda a memória.

Na **Tabela 5,** pode-se observar que não ocorre esse problema, tomando como base que há memória disponível para execução de ambos os processos e como consequência o tempo de chegada é igual o tempo de lançamento.

Tabela 4. Dados de Saída do programa com as entradas, **Memória: 256 GB, Número de CPUs: 2, Tempo Total: 24 slices.**

Chegada	Lançamento	Duração Projetada	Duração Observada
0	0	5	5
5	5	4	9
8	9	5	14
7	15	3	18
12	14	3	20
10	20	4	24

Tabela 5. Dados de Saída do programa com as entradas, **Memória: 512 GB, Número de CPUs: 2, Tempo Total: 16 slices.**

Chegada	Lançamento	Duração Projetada	Duração Observada
0	0	5	5
5	5	4	9
7	7	3	11
8	8	5	13
12	12	3	15
10	11	4	16