**Relatorio:**

**Algoritmos de escalonamento de processos**

**Disciplina:** Sistemas Operativos

**Professor:** Estanislau Lima

**Universidade**: IECM- UTA

**Realizado por:**

-Raffaele Fiorillo;

-Rui Jorge Varela;

**Introdução**

Um dos principais capítulos ao serem tratados na disciplina de Sistemas Operativos é o capitulo sobre processos.

Um processo é gerido dentro do computador, pelo sistema operativo, de forma a ter o que se pode considerar um ciclo de vida.

Há diversos modos de gerir o como é tratado um processo. o responsável por decidir quando, por quanto e como será executado um processo é o tal chamado **escalonador.**

Este trabalho consiste em reproduzir o funcionamento de um escalonador, simulando os diversos algoritmos de escalonamentos estudados durante as aulas.

**Considerações:**

Este trabalho foi desenvolvido de **zero**, sem qualquer ajuda externa (seja ela da internet, livros ou pessoas).

O programa apenas visa ser um **simulador**, logo alguns aspetos dos algoritmos de escalonamento originais foram deixados de fora por não serem pertinentes para a estrita visualização do funcionamento deles.

**Constituição do programa:**

O programa é dividido em 6 ficheiros, 5 contem um algoritmo de escalonamento cada, e o restante ficheiro representa o menu de seleção do algoritmo.

**Modo de uso, conselhos e observações:**

Por preferência deve abrir-se o programa no **prompt de comandos**. Algumas funcionalidades são ativadas por comandos que só tem efeito no prompt, esses comandos são responsáveis por melhorar a experiência do utilizador simplificando a visualização dos resultados obtidos.

O primeiro ficheiro que deve ser aberto é o “**Menu.py**”. Este dará acesso ao menu de escolha dos algoritmos.

Ao escolher um algoritmo não se pode imediatamente sair para o menu. Um estratégia para terminar o funcionamento do algoritmo, caso por algum motivo for selecionado o algoritmo errado, é inserir o valor 0 (zero) quando o programa pede o número de processos que se deseja criar. De pois aparecerá automaticamente a opção para voltar ao menu.

Teoricamente o programa aceita dados do tipo float (em parâmetros que possam ser tal), mas os algoritmos foram pensados para trabalharem com dados do tipo inteiro (int), então (mesmo funcionarem com dados float) é **preferível** serem inseridos dados do tipo inteiro.

**Nenhuma** das funcionalidades espera tomar como input valores do tipo **string** ou **char.** Introduzir valores deste tipo resultará automaticamente em um erro e na interrupção do programa.

Ao acionar um erro o programa é logo interrompido, e dever-se-á recomeçar de novo.

Se existirem processos com um tempo de chegada **inferior** ao instante inicial, estes processos não entrarão na simulação (prestar atenção ao inserir os valores).

Foram comentadas linhas de código que criam **aleatoriamente** um número de processos e suas informações. Caso queira-se, estas linhas podem ser “descomentadas” (talvez para poupar tempo).

Usar este programa em sistemas operativos que não sejam o Windows, resultará em um erro.

**Funcionamento:**

**Menu:**

Ao abrir o ficheiro “Menu.py” são mostrados no ecrã os possíveis algoritmos para realizar a simulação, ou a opção ”sair” com valores numéricos associados. Ao introduzir um valor, chama-se a opção que está associada a ele. Introduzir carateres que não estão no menu resultará em um erro (com a exceção de números inteiros maiores que 6, que serão interpretados como um pedido para sair).

Ao selecionar um algoritmo começa o procedimento para simular o escalonamento. Cada algoritmo tem as suas variações. Sempre é mostrado no limite superior em qual algoritmo nos encontramos (para evitar confusões).

**FCFS (First Come First Served):**

Ao entrar neste modo é pedido para introduzir o **número de processos** que deseja-se criar (deve ser obrigatoriamente um inteiro).

De seguida será pedido para introduzir, para cada processo, o **tempo de chegada** (o instante em que chegou no escalonador) e o **tempo de execução** (quanto tempo levará para ser executado).

Terminando criar todos os processos, é criada uma tabela de processos com todos os seus dados e esta é mostrada no ecrã (para que os valores possam ser conferidos e estudados para avaliar a simulação).

Depois pede-se para introduzir o **instante inicial** (o instante em que começa o escalonamento).

Ao terminar as operações acima indicadas começa a simulação.

A simulação consiste em indicar a cada segundo os acontecimentos que ocorreram. Estes acontecimentos podem ser:

- A execução de um processo (indicando o instante atual, o nome do processo, o tempo de espera e o tempo de execução);

- Um instante vazio (instante em que não existem processos para executar);

Quando todos os processos terminarem de ser executados, aparece uma **linha de resumo** que indica os resultados da execução (o instante em que terminou a execução, e o tempo medio de espera dos processos)

**SJF (Shortest Job First) não preemptivo:**

Ao entrar neste modo é pedido para introduzir o **número de processos** que deseja-se criar (deve ser obrigatoriamente um inteiro).

De seguida será pedido para introduzir, para cada processo, o **tempo de chegada** (o instante em que chegou no escalonador) e o **tempo de execução** (quanto tempo levará para ser executado).

Terminando criar todos os processos, é criada uma tabela de processos com todos os seus dados e esta é mostrada no ecrã (para que os valores possam ser conferidos e estudados para avaliar a simulação).

Depois pede-se para introduzir o **instante inicial** (o instante em que começa o escalonamento).

Ao terminar as operações acima indicadas começa a simulação.

A simulação consiste em indicar a respetiva execução de um processo (indicando o instante atual, o nome do processo, o tempo de espera e o tempo de execução);

Quando todos os processos terminarem de ser executados, aparece uma **linha de resumo** que indica os resultados da execução (o instante em que terminou a execução, e o tempo medio de espera dos processos)

**SJF (Shortest Job First) preemptivo:**

Essa é uma variante do algoritmo anterior, difere apenas (visualmente) nas informações que fornece no que diz respeito à execução dos processos.

Mostra (por cada intervalo) um bloco que contém na primeira linha: - o tempo em que este processo foi interrompido (caso nunca for interrompido mostra 0); - o tempo de chegada deste processo.

Na segunda linha mostra: - o instante de tempo atual; -o nome do processo; - o tempo de espera; - o tempo de execução que lhe falta para terminar;

Cada vez que um processo termina a sua execução, esta informação é mostrada no ecrã (referindo o nome do respetivo processo)

**Por Prioridade:**

Esse algoritmo é visualmente igual ao algoritmo anterior, apenas pede um valor adicional no processo de criação de processos: - o valor da prioridade.

**Round Robin:**

Esse algoritmo não pede a prioridade, mas sim o quantum de tempo. É também o mais completo em termos de informações que fornece.

A cada ciclo fornece (entre cardinais) o número de processos que faltam, no total, para serem processados, e o número de processos que serão executados no ciclo atual.

Mostra os processos executados e os dados atuais deles (como nos casos anteriores).

Mostra (caso aconteça) se não estão sendo executados processos nos instantes atuais.

No fim mostra o resumo dos dados da simulação.

Ao fim de cada um dos algoritmos anteriores é sempre mostrado se se quer voltar ao menu, ou se se quer terminar o programa.

**Conclusão:**

Pude notar ao fazer este trabalho a grande complexidade dos algoritmos de escalonamento. E considerando que estes foram apenas algoritmos de simulação evidencia-se ainda mais a complexidade destes.

De forma geral foi muito satisfatório concluir este trabalho. Gostei muito da experiência.