

Sistemas Operativos

Teste

24 de maio de 2024

Duração: 2h

Por favor responda ao grupo I e a cada exercício do grupo II em folhas de teste separadas. Obrigado.

I

1 No trabalho prático da cadeira de Sistemas Operativos era pretendido desenvolver um serviço de orquestração (*i.e.*, execução e escalonamento) de tarefas (programas) num computador. Este serviço sabe, à partida, qual o tempo de execução de cada tarefa mas não possui quaisquer mecanismos de desafetação forçada de processos (*i.e.*, não consegue interromper um processo durante a execução de uma tarefa). Tendo em conta as considerações anteriores, que algoritmo de escalonamento escolheria para o serviço em questão. Justifique a sua resposta indicando **uma vantagem** e **uma possível desvantagem** da sua escolha.

2 Os sistemas operativos modernos utilizam paginação para gerir a memória dos processos em espaço do utilizador. Ainda, a paginação é muitas vezes combinada com mecanismos de *swapping* de páginas em memória para disco.

- (a) Indique **uma vantagem** de combinar paginação com mecanismos de *swapping*. Indique também **uma preocupação** que o sistema operativo deve ter para tornar o *swapping* de páginas eficiente. Justifique a sua resposta.
- (b) Ao configurar a área de *swap* num dado computador é preferível utilizar um sistema de ficheiros, ou utilizar diretamente **uma partição (raw)** de disco, para suportar a mesma? Justifique a sua resposta.

II

Considere um motor de pesquisa baseado em Inteligência Artificial que dada a interrogação de utilizadores (argumento *prompt*) responde em formato textual. Este motor é composto por um conjunto encadeado de programas.

- Para cada pedido, o ficheiro executável *filter* (*i.e.*, *filter <prompt>*) seleciona ficheiros que contêm conteúdo relacionado com a interrogação, escrevendo em formato binário para o *stdout* os seus caminhos.
- Entretanto, o ficheiro executável *execute* (*i.e.*, *execute <prompt>*) lê do *stdin* os caminhos dos ficheiros filtrados, um a um, e gera conteúdo em binário de acordo com a interrogação do utilizador, o qual é escrito no *stdout*.
- O ficheiro executável *merge* (*i.e.*, *merge <prompt>*) lê do *stdin* conteúdo binário resultante de interrogações e gera uma resposta para o utilizador em formato textual (escrita no *stdout*).

1 Escreva o programa SOGPT de forma a implementar o comportamento acima descrito.

Valorização: otimize a fase de *execute* desencadeando o seu processamento concorrente com N processos.

2 Considere um programa cliente (*search_prompt*) que recorre ao pipe com nome *fifo_server* para enviar um pedido de pesquisa ao programa servidor. O pedido (estrutura *Req q*) contém o *pid* do programa cliente e o *prompt* a ser processado pelo servidor. Por fim, o programa cliente notifica o utilizador quando o pedido é completado, indicando-lhe o seu identificador.

```
1 int main (int argc, char * argv[]) {
2     Req q;
3     q.pid=getpid();
4     q.prompt=strdup(argv[1]);
5
6     char fifoc_name[30];
7     sprintf(fifoc_name, "fifo_client_%d", q.pid);
8     mkfifo(fifoc_name, 0666); ←
9
10    int fds = open("fifo_server", O_WRONLY);
11    write(fds, &q, sizeof(q));
12
13    close(fds);
14
15    int req_id;
16    int fdc = open(fifoc_name, O_RDONLY);
17    read(fdc, &req_id, sizeof(int));
18    printf("Request %d completed\n", req_id);
19
20    close(fdc);
21
22 }
```

(a) Escreva o programa *servidor* que, através do pipe com nome *fifo_server*, deverá receber pedidos de pesquisa de clientes (*i.e.*, *search_prompt*) e executar os mesmos sequencialmente, utilizando o programa SOGPT. Para cada pedido, o servidor deve atribuir-lhe um identificador único, a ser enviado ao cliente assim que o mesmo termina.

(b) Com base na sua solução para a alínea (a), indique se a mesma funcionaria caso a abertura do pipe com nome *fifoc_name* (linha 15) fosse feita logo após a criação do mesmo (linha 8). Justifique a sua resposta.