LICENCIATURA EM ENGENHARIA INFORMÁTICA
LICENCIATURA EM CIÊNCIAS DA COMPUTAÇÃO

Sistemas Operativos

Teste

29 de maio de 2023 — Duração: 2h

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

I

1 Mesmo com um número de processadores limitado, um sistema operativo deve ser capaz de fornecer aos seus utilizadores a ilusão de suportar a execução simultânea de vários processos.

(a) Descreva sucintamente como funciona o mecanismo de comutação de processos e a sua contribuição para percepção da execução simultânea referida acima. Justifique a sua resposta.

(b) Com estratégias de escalonamento baseadas em *fatias de tempo*, o intervalo de tempo escolhido até forçar a comutação de processos tem impacto no desempenho das aplicações. Discuta a importância de escolher intervalos de tempo adequados à carga do sistema. Justifique a sua resposta.

2 Imagine que está a implementar uma aplicação que precisa de armazenar e ler informação de um conjunto de ficheiros de forma eficiente. Na escrita do código da aplicação deparou-se com diferentes abordagens. Com base no seu conhecimento sobre o funcionamento de sistemas operativos, sistemas de ficheiros e dispositivos de armazenamento, indique quais as que acha mais apropriadas.

(a) Para cada ficheiro pode efetuar várias escritas (através da chamada ao sistema *write*) com um tamanho reduzido (p.ex. 10 bytes) ou realizar escritas com um tamanho maior (p.ex. 4096 bytes). Qual das opções acha melhor para otimizar o desempenho da sua aplicação? Justifique a sua resposta.

(b) Ao ler cada ficheiro pode realizar leituras de forma sequencial ou aleatória sobre os blocos de dados contidos no mesmo. Qual o padrão de acessos, sequencial ou aleatório, mais eficiente em termos de desempenho? Justifique a sua resposta.

II

Considere um sistema de detecção de anomalias numa linha de produção baseado em algoritmos de Inteligência Artificial (IA). As anomalias são detectadas com base em fotografias do produto. Por exemplo, no caso do produto ser um chip, os modelos de IA irão tentar perceber se o chip tem algum defeito visível. Este algoritmos estão implementados no ficheiro executável `detectAnom` que recebe como argumento o caminho para um ficheiro de imagem e escreve no `stdout` uma linha por cada defeito encontrado.

1 Escreva a função `void defeitos(char* imagens[n], int n, int max)` que, para um conjunto de caminhos para `imagens` passados num array de `n` elementos, escreve para o `stdout` todos os defeitos encontrados. A função deve desencadear uma procura concorrente, recorrendo ao ficheiro executável `detectAnom`, nunca excendendo um total de `max` processos a executar simultâneamente.

2 Escreva a função `void conta(char* imagens[n], int n)` que, para um conjunto de caminhos para `imagens` passados num array de `n` elementos, devolve para o `stdout` o número total de defeitos que foram encontrados. Para resolver este exercício, deve tirar proveito da função `defeitos` desenvolvida na alínea anterior e também do comando unix `wc -l`, o qual lê linhas do `stdin` e retorna o número total de linhas para o `stdout`.

Algumas chamadas ao sistema relevantes

Processos

- pid_t fork(void);
- void exit(int status);
- pid_t wait(int *status);
- pid_t waitpid(pid_t pid, int *status, int options);
- WIFEXITED(status);
- WEXITSTATUS(status);
- int execlp(const char *file, const char *arg, ...);
- int execvp(const char *file, char *const argv[]);
- int execve(const char *file, char *const argv[], char *const envp[]);

Sistema de Ficheiros

- int open(const char *pathname, int flags, mode_t mode);
- int creat(const char *pathname, mode_t mode);
- int close(int fd);
- int read(int fd, void *buf, size_t count);
- int write(int fd, const void *buf, size_t count);
- int pipe(int filedes[2]);
- int dup(int oldfd);
- int dup2(int oldfd, int newfd);
- int mkfifo(const char *pathname, mode_t mode);