Enunciado do Projeto de Sistemas Operativos 2015-16
SHELL PARALELA - EXERCÍCIO 2

LEIC-A / LEIC-T / LETI
IST

1 Shell paralela com monitorização de desempenho

O 2º exercício tem como objetivo estender a *par-shell* com a funcionalidade de monitorização de desempenho. Para tal, deve ser criada uma tarefa (*thread*) adicional na *par-shell*, que será responsável por monitorizar os tempos de execução de cada processo filho.

A tarefa de monitorização, aqui denominada tarefa monitora, coordena-se com a tarefa principal (que existe, por omissão, sempre em qualquer processo) da seguinte forma:

- A tarefa principal deverá passar a registar o *pid* de cada processo filho que lançou e o respetivo tempo numa estrutura de dados partilhada (entre as duas tarefas em causa, i.e. a principal e a monitora). A medição do tempo deverá ser feita usando a função `time`.
- A informação sobre processos filho, partilhada entre ambas as tarefas, deverá ser mantida numa lista baseada na que foi construída na primeira aula de laboratório.
- A tarefa monitora espera pela terminação dos processos filho; para cada processo filho terminado, a tarefa de monitora mede o tempo de terminação e adiciona-o no registo correspondente na estrutura partilhada.

Obviamente, é necessário que ambas as tarefas acima mencionadas se coordenem. Caso contrário, a tarefa monitora não sabe se há processos filhos que justifiquem que esta se bloqueie esperando pela sua terminação. Essa coordenação deve ser feita da seguinte forma:

- Além da lista com informação sobre os processos filhos, a tarefa principal partilhará com a tarefa monitora um inteiro (aqui denominado *numChildren*) que indica o número de processos filho em execução.
- Assim, obviamente, sempre que a tarefa principal lança um novo processo filho (chamando `fork`), incrementa o inteiro *numChildren*.
- A tarefa monitora consiste, basicamente, num ciclo infinito; em cada iteração do ciclo, consulta o valor de *numChildren* para saber se há pelo menos um processo filho em execução. Se não há, espera 1 segundo antes da próxima iteração (invocando a função `sleep`). Se há pelo menos um processo filho, espera pela terminação do(s) processo(s) filho, invocando `wait`.

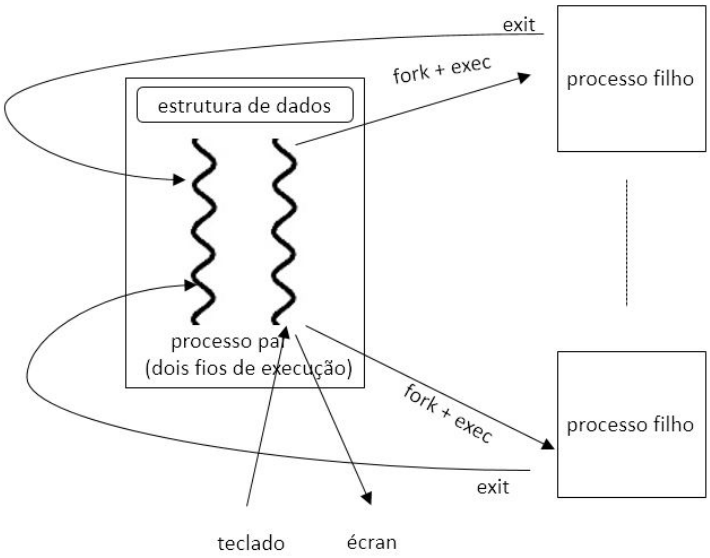

Figura 1: Representação dos processos e de algumas das chamadas sistema a utilizar.

Há um caso que requer especial cuidado: quando o utilizador da *par-shell* dá ordem de terminação (através do comando `exit`), a tarefa principal deve informar a tarefa monitora para que esta, depois de esperar que todos os processos filho terminem (registando o seu tempo de terminação), também termine. Assim, a tarefa principal deverá aguardar que a tarefa monitora termine.

Após este ponto de sincronização, antes do processo em causa terminar, a tarefa principal deverá imprimir a informação completa sobre todos os processo filho que foram lançados (e completados) ao longo da execução da *par-shell*: pid e tempo de execução em segundos (i.e. tempo de terminação - tempo de lançamento).

Devem ser as funções POSIX para gestão de tarefas e trincos lógicos, ensinadas nas aulas teóricas.

2 Experimente

Pode testar o funcionamento da nova par-shell recorrendo ao programa fibonacci fornecido no primeiro exercício. Experimente lançar o programa com diferentes valores e confirme que o tempo de execução aumenta com o valor do argumento.

Discuta o comportamento da tarefa monitora; em particular, considere o caso em que não há processos filho. Altere o valor do tempo de `sleep` e relacione com a matéria dada na aula teórica sobre a noção de espera ativa.

3 Entrega e avaliação

Os alunos devem submeter um ficheiro no formato zip com o código fonte e *makefile* através do sistema Fénix. O exercício deve obrigatoriamente compilar e executar nos computadores dos laboratórios.

A data limite para a entrega do exercício é 23 de Outubro até às 23h59m.

Após a entrega, o corpo docente disponibilizará a codificação da respetiva solução, que pode ser usada pelos alunos para desenvolverem os exercícios seguintes.

A demonstração da solução do exercício feita pelos alunos acontece durante aula laboratorial após a entrega (em semana a indicar no site da cadeira). No início dessa aula laboratorial, será dada aos alunos uma alínea adicional que complementa o enunciado apresentado neste documento. A alínea adicional é de resolução rápida para quem preparou e resolveu o enunciado base.

A nota é individual a cada membro do grupo. Membros que não compareçam na aula de demonstração têm nota nula neste exercício. As notas dos exercícios 1 a 5 são indicativas estando sujeitas a confirmação na discussão final na qual todo o *software* desenvolvido durante o semestre será tido em conta.

4 Cooperação entre Grupos

Os alunos são livres de discutir com outros colegas soluções alternativas para o exercício. No entanto, *em caso algum* os alunos podem copiar ou deixar copiar o código do exercício. Caso duas soluções sejam cópias, ambos os grupos reprovarão à disciplina.