Enunciado do Projecto de Sistemas Operativos 2015-16
SHELL PARALELA - EXERCÍCIO 1

LEIC-A / LEIC-T / LETI
IST

O projecto de Sistemas Operativos 2015/16 está organizado em 5 exercícios encadeados. Esta secção introdutória apresenta uma visão global do projecto e dos tópicos abordados em cada um dos exercícios.

O projeto consiste em desenvolver uma *shell*, chamada `par-shell`, que permite executar e monitorizar lotes de programas em paralelo numa máquina *multi-core*.

Os 5 exercícios que constituem o projecto abordam os tópicos seguintes (aqui apresentados de forma resumida) :

- Exercício 1 - Desenvolverá o programa base da `par-shell`, que permite ao utilizador lançar múltiplos programas em paralelo.
- Exercício 2 - Estenderá a `par-shell` com monitorização do desempenho de cada processo filho.
- Exercício 3 - Estenderá a `par-shell` com limitação do paralelismo.
- Exercício 4 - A monitorização do desempenho dos comandos executados é periodicamente escrita em ficheiro.
- Exercício 5 - Suportará o acesso à shell através de terminais remotos e a redireção do *output* dos comandos.

As secções seguintes deste documento descrevem apenas o 1^o exercício. Serão disponibilidados documentos similares para cada um dos exercícios seguintes.

1 Shell paralela

O 1^o exercício consiste em desenvolver o módulo inicial da shell paralela.

Tal como qualquer shell de linha de comandos, a `par-shell` é um programa que permite a um utilizador emitir ordens para executar programas existentes no sistema de ficheiros da máquina. A caracteristica relevante da `par-shell` prende-se com o facto de permitir a execução de múltiplos comandos em paralelo.

No 1^o exercício, os alunos deverão codificar um programa `par-shell` que aguarda por ordens do teclado. Devem ser suportados os seguintes comandos:

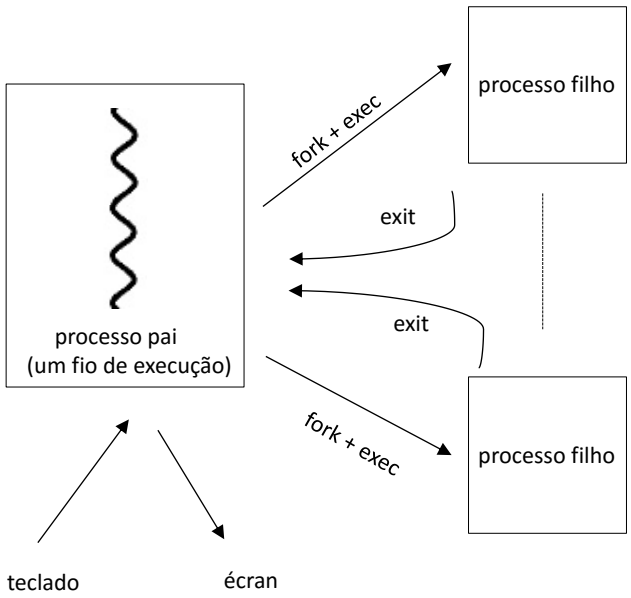

Figura 1: Representação dos processos e de algumas das chamadas sistema a utilizar.

- `pathname [arg1 arg2 ...]`, que executa o programa contido no ficheiro indicado pelo *pathname*, como processo fiho, passando-lhe os argumentos opcionais que sejam indicados (até um máximo de 5 argumentos permitidos). O processo filho é lançado em *background*, ou seja a `par-shell` não espera pela terminação dos processos filho que são lançados e fica pronta a lançar novos processos filho. Caso o lançamento de algum processo falhe (por exemplo, devido a um pathname inválido ou a erro na criação de processo filho), a `par-shell` deve apresentar uma mensagem reportando o erro no *stderr* mas a `par-shell` não deve terminar.

- `exit`, que termina a `par-shell` de forma ordeira. Em particular, espera pela terminação de todos os processos filho (incluindo aqueles que não tenham ainda terminado aquando da ordem de `exit`). Após todos os processos filho terem terminado, a `par-shell` deve apresentar no *stdout* o *pid* e o inteiro devolvido por cada processo filho.

Para este exercício devem ser usadas as seguintes funções da API do Unix/Linux: `fork, exec*, wait, exit` além das funções habituais da biblioteca *stdio*.

A leitura da linha comandos deve ser feita usando a biblioteca *commandlinereader*, que pode ser obtida no site dos laboratórios.

A solução deverá incluir uma *makefile* com um alvo chamado `par-shell`que gera um ficheiro executável da solução, com o mesmo nome. Soluções que não cumpram este requisito não serão avaliadas.

2 Experimente

Use o programa `fibonacci` fornecido no site dos laboratórios como exemplo para testar a sua `par-shell`. Executando esse programa com argumento elevado, os processos filhos demoram vários segundos a terminar, o que permite testar situações em que mútiplos processos filhos se executam em simultâneo.

Para experiências com múltiplos programas, sugerimos que componha ficheiros de texto com sequências de comandos. Por exemplo:

```
fibonacci 10000
fibonacci 10001
fibonacci 10002
exit
```

Para correr estes lotes de comandos, basta depois lançar na linha de comandos: `par-shell < input.txt` (em que input.txt é um ficheiro com uma sequência de comandos tal como a apresentada acima).

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 primeiro exercício é 9 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 a aula laboratorial de cada grupo na semana de 12-16 de Outubro. 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.