Enunciado do Projecto de Sistemas Operativos 2016-17
Serviço de banca paralelo - Exercício 2

LEIC-A / LEIC-T / LETI
IST

1 Operações bancárias paralelas no `i-banco`

No segundo exercício pretende-se paralelizar todas as operações que o processo principal do `i-banco` oferece sobre as suas contas (creditar, debitar, ler saldo). Dessa forma a aplicação fica apta a explorar o paralelismo da máquina onde corre de uma forma mais eficaz.

Ao iniciar, o `i-banco` deve criar uma *pool* com `NUM_TRABALHADORAS` tarefas, sendo esse valor definido através de uma constante.

Sendo assim, o `i-banco` passa a ter `NUM_TRABALHADORAS+1` tarefas: a tarefa inicial, que recebe comandos do *stdin*, e as `NUM_TRABALHADORAS` tarefas, que executam os comandos recebidos pela tarefa inicial.

A comunicação entre a tarefa inicial e as restantes tarefas é feita por um *buffer* circular de `2 * NUM_TRABALHADORAS` entradas. A tarefa inicial recebe cada comando e, caso exista vaga, coloca-o no *buffer* para posterior leitura e execução por uma das restantes tarefas. Cada tarefa trabalhadora deve esperar pela chegada de um comando no *buffer* partilhado. Naturalmente, as tarefas devem ser sincronizadas de forma a assegurar que:

- Para cada comando que a tarefa inicial coloque no *buffer*, apenas uma tarefa trabalhadora o recebe e executa a respetiva operação.
- Caso uma tarefa trabalhadora esteja livre e não haja nenhum comando disponível no *buffer*, a tarefa deve bloquear-se até receber um novo comando.
- Caso a tarefa inicial receba novo comando e o *buffer* esteja cheio, esta deve bloquear-se até surgir uma posição livre no mesmo.
- A execução de cada operação sobre as contas deve ser sincronizada, já que pode haver contenção no acesso à mesma conta por diferentes tarefas trabalhadoras.

Ao receber os comandos `"sair"` ou `"sair agora"`, o `i-banco` só deverá terminar quando todos os comandos anteriormente colocados no *buffer* tenham sido executados e completados.

Relativamente às simulações, que são realizadas por processos filho, o `i-banco` deve manter o comportamento já implementado no primeiro exercício.

2 Detalhes de implementação

Para assegurar uniformidade na interacção entre a tarefa inicial e as tarefas trabalhadoras, deve ser seguida a seguinte abordagem:

- Cada comando a ser passado a uma tarefa trabalhadora é definido pela seguinte estrutura:

```
typedef struct
{
  int operacao;
  int idConta;
  int valor;
} comando_t;
```

- O *buffer* circular mencionado na secção anterior será constituído por um vector de estruturas *comando_t*, existindo um índice de escrita e outro de leitura:

```
#define NUM_TRABALHADORAS 3
#define CMD_BUFFER_DIM  (NUM_TRABALHADORAS * 2)

comando_t cmd_buffer[CMD_BUFFER_DIM];

int buff_write_idx = 0, buff_read_idx = 0;
```

- A tarefa inicial lê os comandos de *stdin* e, para cada comando a enviar para uma tarefa trabalhadora, preenche uma estrutura *comando_t* com a informação relevante. As tarefas trabalhadoras lêem comandos de *cmd_buffer* e processam o seu conteúdo (analisam *operacao* e, consoante o caso, usam os valores contidos nos demais campos que sejam necessários à execução do comando).
- As mensagens resultantes da execução de cada comando devem ser impressas directamente no *stdout* por cada tarefa trabalhadora.

Devem ser usadas as funções da API do Unix/Linux estudadas na teoria para criação das tarefas e realização dos mecanismos de sincronização necessários. Sugere-se a consulta das funções: *pthread_create*, *pthread_join*, *pthread_mutex_init*, *pthread_mutex_lock*, *pthread_mutex_unlock*, *sem_init*, *sem_post* e *sem_wait*.

3 Entrega e avaliação

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

A data limite para a entrega do segundo exercício é 28/10/2016 até às 23h59m. A submissão é feita através do Fénix.

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

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.