Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores

Sistemas Operativos, Verão de 2020/21

Quinta Série de Exercícios

Pthreads

1. Implemente, como um monitor nas `pthread`, o sincronizador `semaphore_t`, que implementa a semântica de semáforo contador, com as operações:

```
// inicia o semáforo com o número de unidades especificado em initial.
void semaphore_init(semaphore_t *sem, int initial);

// adquire, com timeout infinito, o número de unidades especificado em units.
void semaphore_acquire(semaphore_t *sem, int units);

// adquire, com timeout de millis milisegundos, o número de unidades especificado em units.
bool semaphore_acquire_timed(semaphore_t *sem, int units, long millis);

// entrega o número de unidades ao semáforo especificado em units.
void semaphore_release(semaphore_t *sem, int units);
```

2. Realize, como um monitor nas `pthread`, o sincronizador `count_latch_t`, utilizável em cenários em que uma ou mais *threads* distribuem, de forma dinâmica, trabalho por *worker threads*, e esperam pela sua conclusão, com eventual limite do nível de paralelismo (máximo de *worker threads* a trabalhar em simultâneo).

```
// Inicia o count latch com o nível máximo de paralelismo especificado em par_level. Caso o valor de
// par_level seja 0, considera-se não haver limite ao nível de paralelismo.
void cl_init(count_latch_t *latch, int par_level);

// Espera que a contagem das unidades de trabalho seja 0.
void cl_wait_all(count_latch_t *latch);

// Contabiliza mais uma unidade de trabalho. Bloqueia a thread invocante caso o nível máximo de
// paralelismo tenha sido atingido.
void cl_up(count_latch_t *latch);

// Contabiliza menos uma unidade de trabalho. Caso se tenha descido abaixo do nível máximo de paralelismo
// liberta umas das threads bloqueadas na operação cl_up. Caso o número de unidades de trabalho chegue a
// 0 desbloqueia todas as threads bloqueadas na operação cl_wait_all.
void cl_down(count_latch_t *latch);
```

3. Considere o programa `search.c` em anexo, que obtém, de forma sequencial, os ficheiros com determinada extensão, existentes na pasta indicada ou em suas subpastas, em que determinada palavra ocorre. Escreva uma versão do programa que explore a multiplicidade de processadores do sistema onde é executado, considerando como unidade de trabalho o processamento de um ficheiro. A *thread* principal tem como única responsabilidade distribuir trabalho, esperando depois pela sua conclusão. Na sua solução utilize o sincronizador `count_latch_t` realizado na questão anterior e o *thread pool* desenvolvido nas aulas.

ISEL, 2 de junho de 2021

Data de entrega: 16 de junho de 2021