

Arquiteturas Paralelas e Distribuídas

Ferramentas para programação paralela: Pthreads

Eduardo Furlan Miranda

Baseado no material do Prof. FERRETO, T.
Programação Paralela. 2006.

Pthreads (POSIX Threads)

- Modelo de execução que existe independentemente de uma linguagem de programação, bem como um modelo de execução paralela
- Permite que um programa controle vários fluxos diferentes de trabalho que se sobrepõem no tempo
- Cada fluxo de trabalho é chamado de thread, e a criação e o controle sobre esses fluxos são obtidos por meio de chamadas para a **API POSIX Threads** (definida pelo padrão IEEE Std 1003.1c-1995)
- Existem implementações para vários SOs, incluindo Linux e Windows
 - No Linux geralmente se usa a biblioteca **libpthread**

- Define um conjunto de tipos, funções e constantes, para a linguagem C
- É implementado com um cabeçalho `pthread.h` e uma biblioteca de threads
- Existem cerca de 100 procedimentos de threads, todos prefixados `pthread_`

- **Pthreads - 5 grupos ou categorias**
- Gerenciamento de threads – criação, junção de threads, etc.
- **Mutexes** garante que apenas uma thread por vez tenha acesso a uma região crítica do código ou a um recurso compartilhado
- Variáveis de condição
- Sincronização entre threads usando bloqueios e barreiras de leitura e gravação
- **Spinlocks** não suspende a thread, evita troca de contexto

Semáforos

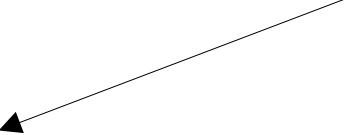
- A API de semáforo POSIX funciona com threads POSIX, mas não faz parte do padrão de threads, tendo sido definida no padrão POSIX.1b, Real-time extensions (IEEE Std 1003.1b-1993)
- Consequentemente, os procedimentos de semáforo são prefixados por `sem_` em vez de `pthread_`

Semáforo: permite que mais de uma thread acesse o mesmo recurso

Exemplo

```
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
```

arquivo de cabeçalho que
fornece acesso à API do
sistema operacional POSIX



```
#define NUM_THREADS 5
```

```
void * perform_work(void * arg){
    int index = *((int *)arg);
    int sleep_time = 1 + rand() % NUM_THREADS;
    printf("Thread %d: Sleeping for %d seconds.\n", index, sleep_time);
    sleep(sleep_time);
    printf("Thread %d: Ended.\n", index);
    return NULL;
}
```

gerar um tempo de espera
aleatório para cada thread



(continua)

Exemplo

(continuação)

7/9

```
int main(void) {  
    pthread_t threads[NUM_THREADS];  
    int thread_args[NUM_THREADS];  
    for (int i = 0; i < NUM_THREADS; i++) {  
        thread_args[i] = i;  
        pthread_create(&threads[i], NULL, perform_work, &thread_args[i]);  
    }  
    for (int i = 0; i < NUM_THREADS; i++) {  
        pthread_join(threads[i], NULL);  
    }  
    printf("Main program ended.\n");  
    return 0;  
}
```

tipo de dado usado para armazenar identificadores de threads POSIX.

faz o programa aguardar a finalização da thread especificada.

(continua)

- Compila com
 - `gcc thread.c -pthread -o thread.out`
- Saída
 - `$./thread.out`
Thread 0: Sleeping for 4 seconds.
Thread 1: Sleeping for 2 seconds.
Thread 2: Sleeping for 3 seconds.
Thread 3: Sleeping for 1 seconds.
Thread 4: Sleeping for 4 seconds.
Thread 3: Ended.
Thread 1: Ended.
Thread 2: Ended.
Thread 0: Ended.
Thread 4: Ended.
Main program ended.

Threads POSIX para Windows

- O projeto **Pthreads4w** busca fornecer uma implementação wrapper portátil e de código aberto
- Ele também pode ser usado para portar software Unix (que usa pthreads) com pouca ou nenhuma modificação na plataforma Windows
- A versão 3.0.0 ou posterior do Pthreads4w, lançada sob a Apache Public License v2.0, é compatível com sistemas Windows de 64 ou 32 bits
- O projeto **Mingw-w64** possui o **winpthreads**, sua biblioteca para "traduzir" pthreads (threads do Linux) para o Windows. Esta versão tenta ser mais otimizada que a Pthreads4w ao utilizar preferencialmente as funções originais do próprio Windows
- O subsistema de ambiente Interix disponível no pacote Windows Services for UNIX/Subsystem for UNIX-based Applications fornece uma porta nativa da API pthreads, não mapeada na API Win32, construída diretamente na interface syscall do sistema operacional