

Operating Systems: Practice: Lesson 4

Sevak Amirkhanian

A dark blue diagonal gradient bar that starts from the bottom left and extends towards the top right, covering the lower half of the slide.

Change your .gitignore

CMakeLists.txt.user

CMakeCache.txt

CMakeFiles

CMakeScripts

Testing

Makefile

cmake_install.cmake

install_manifest.txt

compile_commands.json

CTestTestfile.cmake

_deps

build/

Simple example for reusing thread

We will create a thread and execute multiple functions using the same thread.

Semaphores

Semaphore is a another synchronization primitive which have similarities with conditional variables.

Semaphore is basically an integer counter which has 2 operations:

`P()` // increments the counter

`V()` // decrements the counter

The value of the semaphore is the number of units of the resource that are currently available

POSIX semaphores

POSIX semaphores have 4 main operations:

sem_init(...)

sem_post(..)

sem_wait(..)

sem_destroy(...)

https://man7.org/linux/man-pages/man7/sem_overview.7.html

Binary semaphores

If the semaphore count is 1, semaphore is essentially a mutex which is why in literature mutex is sometimes called *binary semaphore*.

`sem_wait()` -> `pthread_mutex_lock()`

`sem_post()` -> `pthread_mutex_unlock()`

Real-life examples of semaphores

Semaphores may be used in networking operating systems in routers for connection throttling.

Semaphores may also be used database drivers to restrict the amount of parallel connections, hence read-write operations.

Producer-Consumer Problem

The simple of description of the problem is:

We have a storage with limited storage with size N .

We have M producers who periodically produce a new item and insert into the storage if and only if there is a free space in the storage.

We have K consumers who periodically consume a new item from the storage if and only if there the storage is not empty.

Solution with semaphores

We will use 3 semaphores:

1 binary semaphore for lock

1 counting semaphore for full state

1 counting semaphore for empty state

Object Pool Pattern

The object pool pattern is a creational design pattern which uses previously allocated and initialized set of objects instead of initializing them on demand.

Real-life examples:

1. C# runtime called CLR pools all strings
2. ODBC drives pool database connections

Homework 3: Thread Pool

Threads are kernel objects and working with them has obvious performance penalties. Therefore pooling them seems a good idea.

You need to provide an interface executing multiple tasks on a finite set of kernel threads.

The interface can be seen in `threadpool.h` file.

Extra point:
Increase threadpool size if all threads are busy.

Thank you.