VNU–HCMUS
FACULTY OF INFORMATION TECHNOLOGY

Chapter 4A

Thread

Instructor: **Vu Thi My Hang, Ph.D.**
TA/Lab Instructor: **Le Quoc Hoa, M.Sc.**

Plan

- Thread Concepts
- Thread Models
- Thread Management

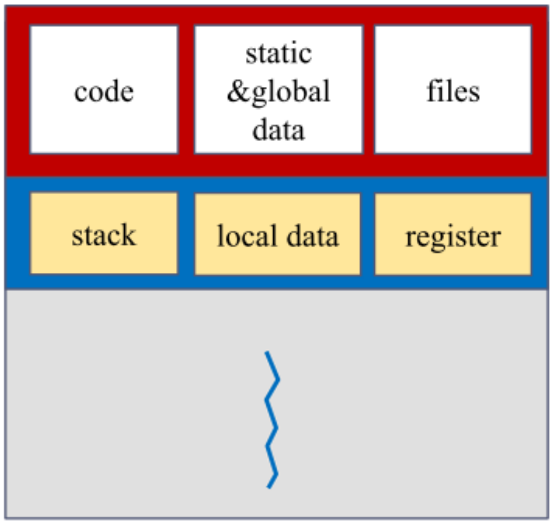
Plan

- **Thread Concept**
- Thread Models
- Thread Management

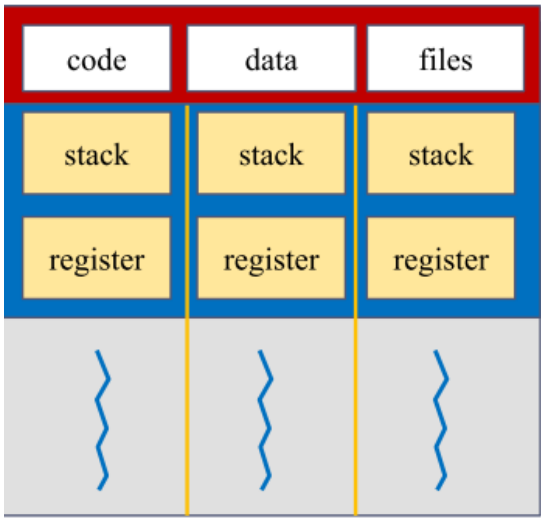
What is a Thread? (Tiểu trình, Luồng)

Thread: an execution line within a process (i.e., a lightweight process)

- A process is divided into different parts (threads)
- Threads belonging to a process can execute concurrently



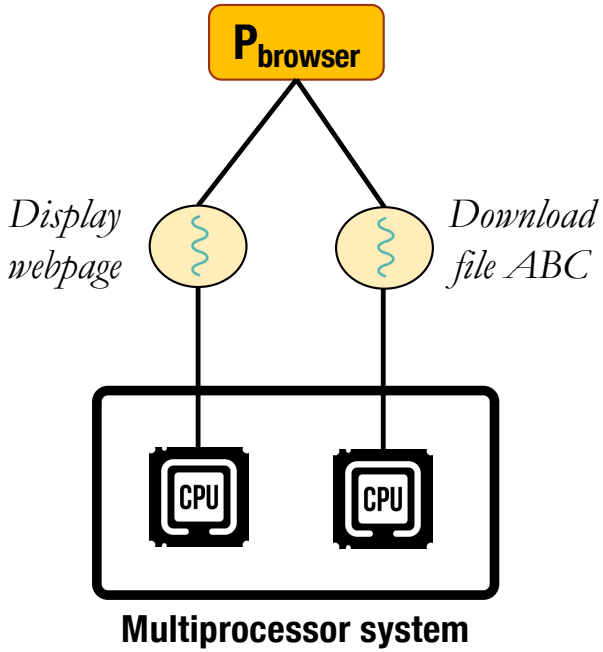
Single-threaded process

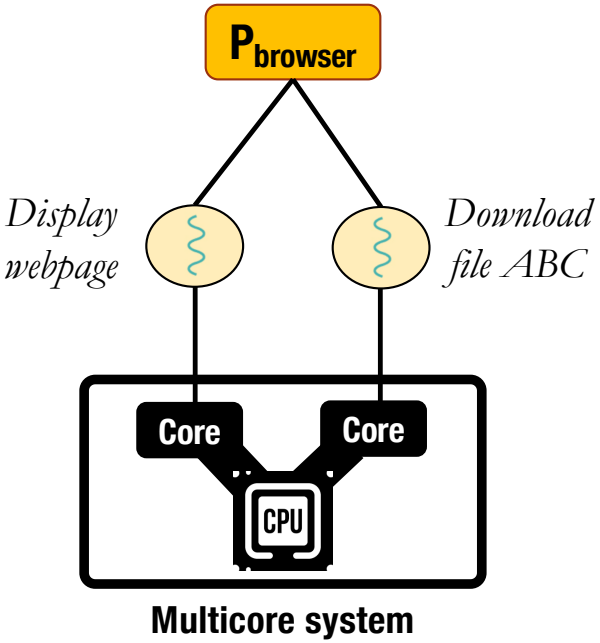


Multi-threaded process

Purpose

- Process perform various tasks *(threads of execution)* simultaneously
  - ✓ Especially efficient on multiprocessor or multicore systems





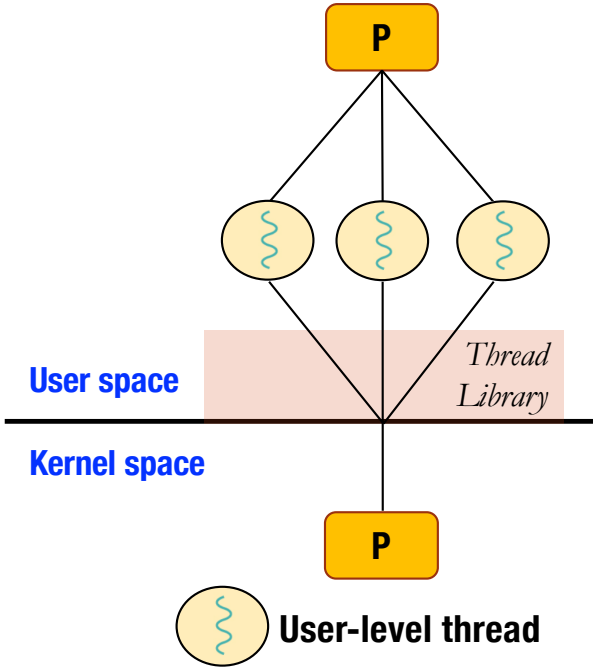
Benefits

- Responsiveness (Tính phản hồi nhanh)
  - ✓ If a thread of a process is blocked or time-consuming, other parts of the process still can continue their jobs
- Resources sharing (Chia sẻ tài nguyên)
  - ✓ Threads belonging to the same process share the same address space
- Economy (Giảm chi phí)
  - ✓ Less cost of thread creation and management
- Scalability (Khả năng mở rộng)
  - ✓ Threads can be executed on different cores/processors

Plan

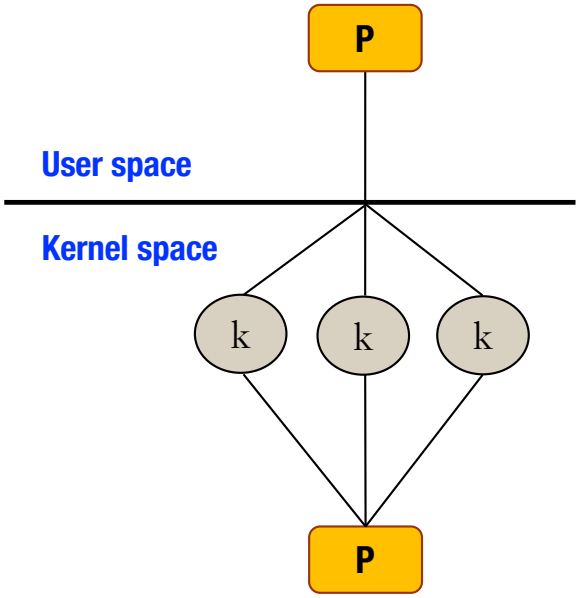
- Thread Concept
- **Thread Models**
- Thread Management

Thread-level Implementation



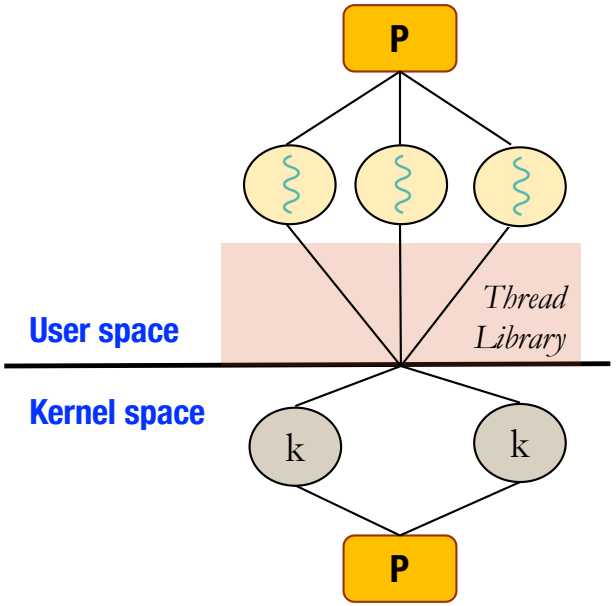
User-level thread

Supported in user space



k Kernel-level thread

Supported in kernel space



Hybrid thread

Supported in both user and kernel space

User-level Threads

- Implemented by user thread libraries (i.e., APIs supported by programming languages) in user space with no kernel support

☺ Be efficient

✓ Thread management, context switching, and synchronization done without any OS intervention

☺ Do not need system support

☹ The kernel knows nothing about user threads

✓ Entire process will be blocked if one of its threads is blocked

✓ Timesharing system: giving the same amount of time to each process for execution ➔ Fairness when P1 having 10 threads and P2 having 1000 threads?

Kernel-level Threads

- Supported directly by OS and implemented in kernel space

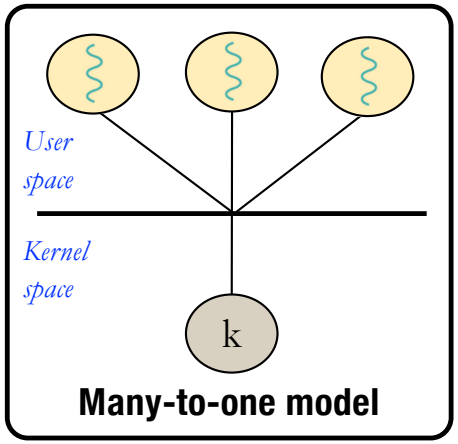
☺ OS manages thread

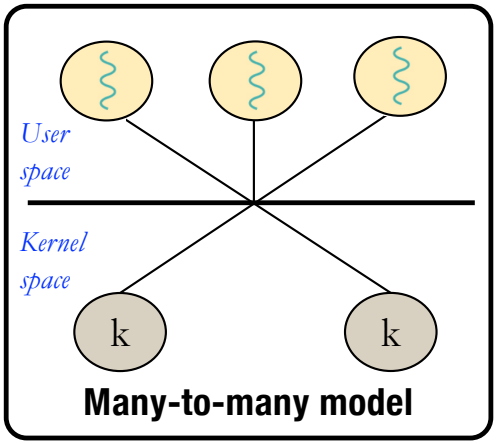
✓ When a thread of a process is blocked, other parts of the process can continue to execute independently

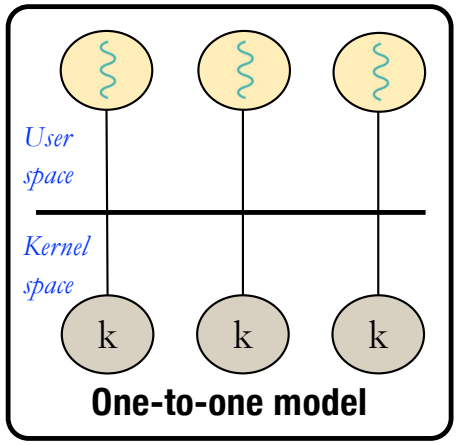
☹ Slower and less efficient

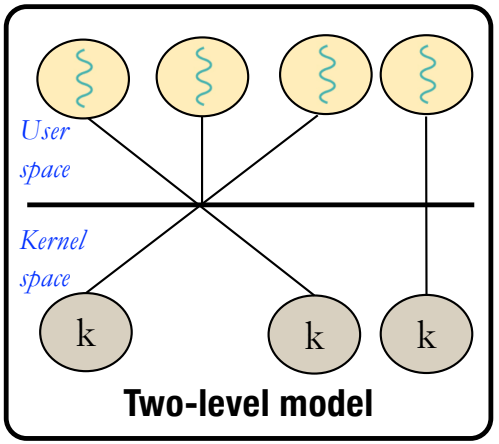
✓ Thread management, context switching, and synchronization require OS intervention

Kernel-level and User-level Thread Mapping





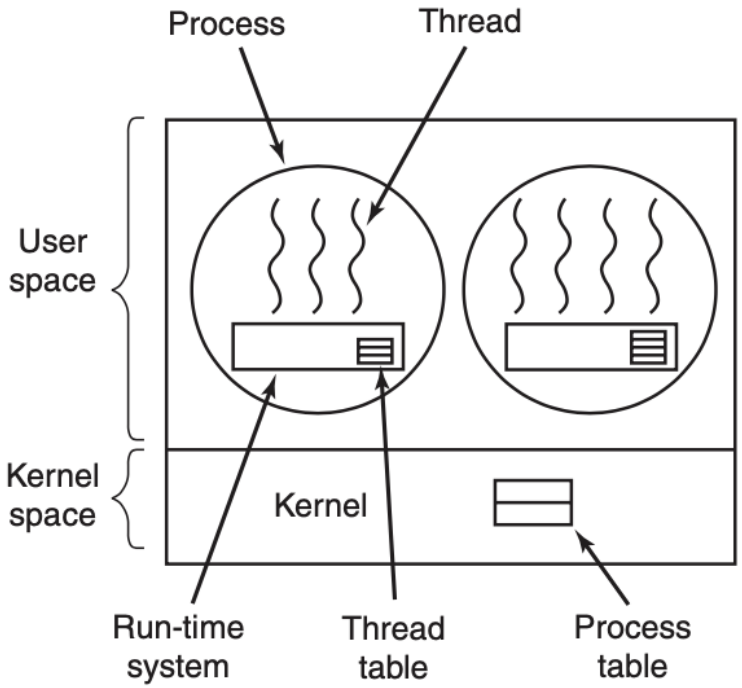




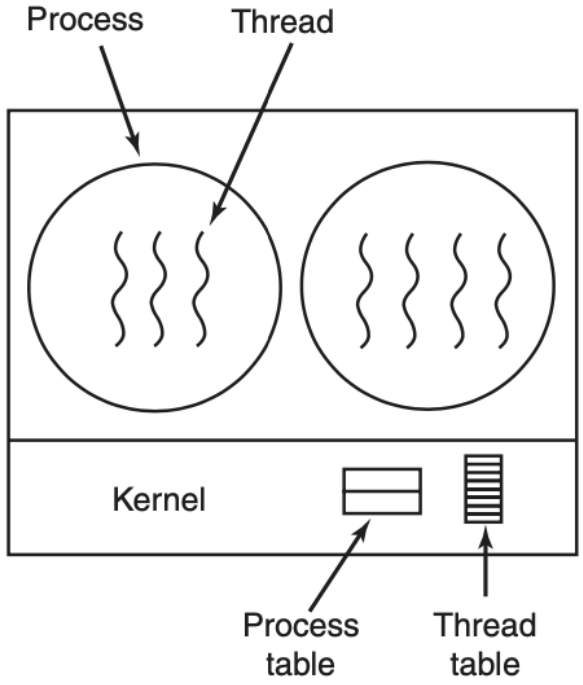
Plan

- Thread Concept
- Thread Models
- **Thread Management**

User-level vs. Kernel-level Management

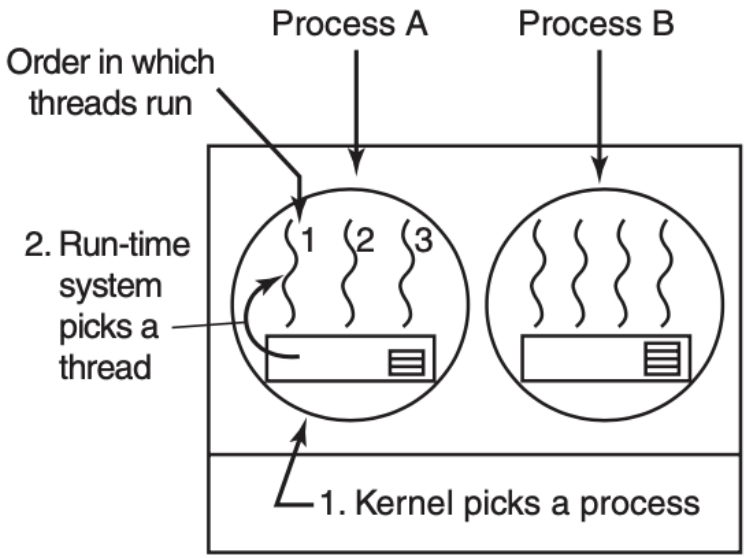


User-level thread Management

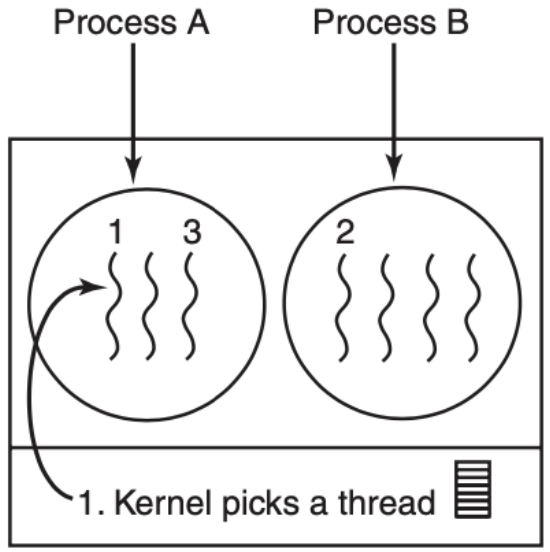


Kernel-level thread Management

Thread Scheduling



User-level thread Scheduling



Kernel-level thread Scheduling

Thread Libraries

- Provide APIs for thread creation and management
  - ✓ POSIX thread (Pthread): support both user and kernel thread libraries
  - ✓ Windows thread: support kernel threads managed directly by Windows
  - ✓ Java thread: support Java thread programming

Thread call	Description
Pthread_create	Create a new thread
Pthread_exit	Terminate the calling thread
Pthread_join	Wait for a specific thread to exit
Pthread_yield	Release the CPU to let another thread run
Pthread_attr_init	Create and initialize a thread's attribute structure
Pthread_attr_destroy	Remove a thread's attribute structure

Some of Pthread functions