Threads

Systems II

Overview

A thread (of control) is a basic unit of CPU utilisation.

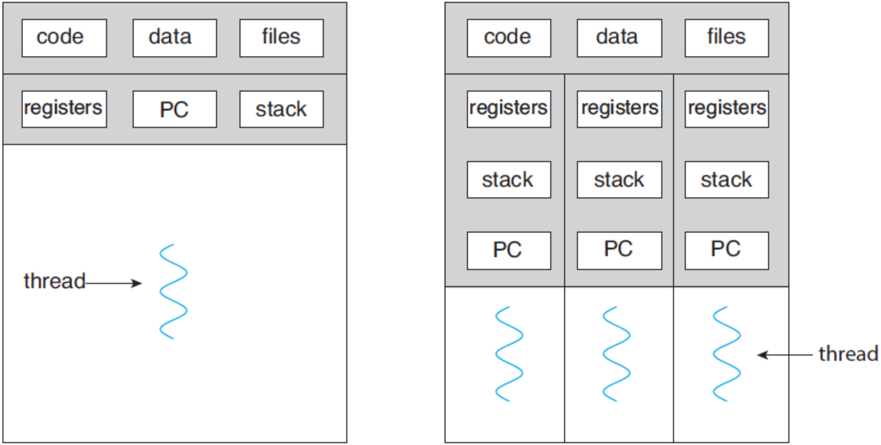
Each thread has its own:

- thread ID
- register set, program counter (PC)
- stack

All threads of the same process share:

- code section
- data section
- other OS resources (open files and signals)

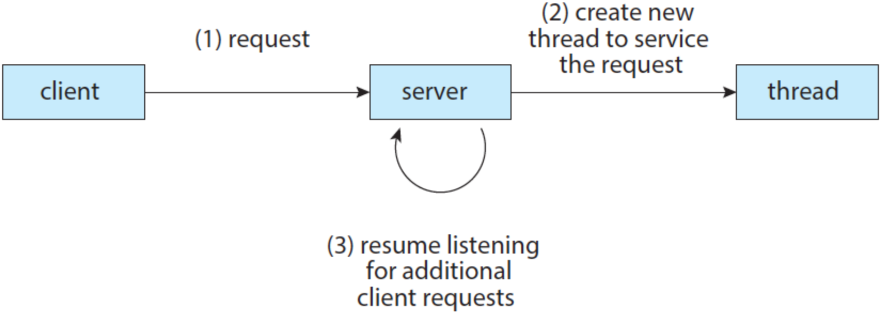
Single-threaded vs. multi-threaded processes



single-threaded process

multithreaded process

An example


(1) request
(2) create new thread to service the request
client
server
thread
(3) resume listening for additional client requests

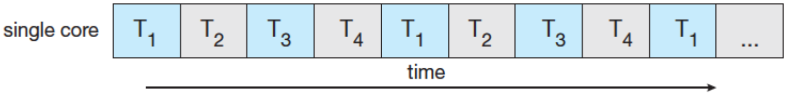
Benefits of multithreading

- responsiveness (e.g. user interfaces)
- resource sharing (e.g. same address space)
- economy (process creating is costly, switching between threads faster)
- scalability (multiprocessor architecture)

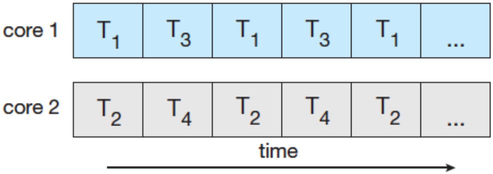
Multicore vs. single-core systems

concurrency vs. parallelism

Concurrent execution on a single-core system:



Parallel execution on a multicore system:

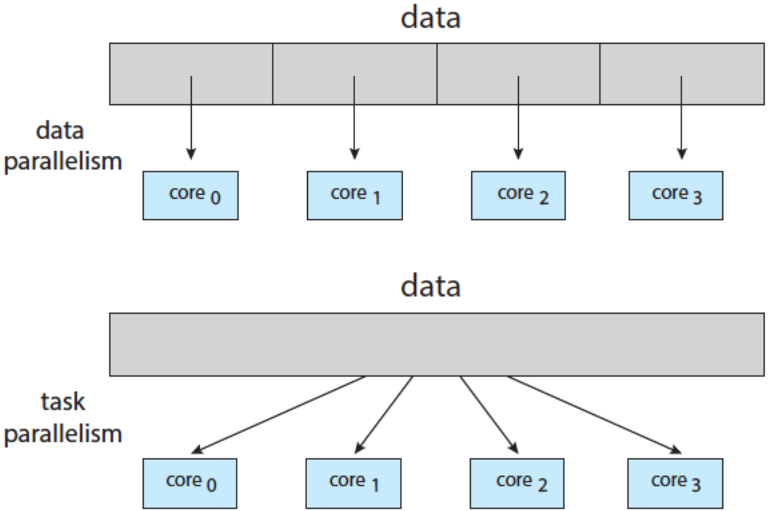


Challenges

- identifying tasks: find areas that can be divided into separate, concurrent tasks
- balance: ensure that the tasks perform equal work of equal value
- data splitting: data accessed and manipulated by the tasks must be divided to run on separate cores
- data dependency: the data accessed by the tasks must be examined for dependencies between two or more task
- testing and debugging: when a program is running in parallel on multiple cores, many different execution paths are possible

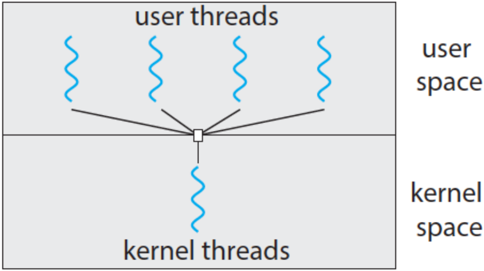
Example: Can we parallelise Game of Life?

Types of Parallelism



Multithreading models

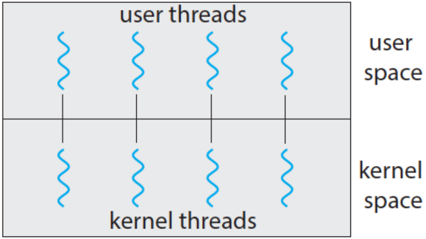
Many-to-one model



- management is done by the thread library in user space
- the entire process will block if a thread makes a blocking system call
- multiple threads are unable to run in parallel on multicore systems
- example: Green threads (thread library for Solaris systems)

Multithreading models

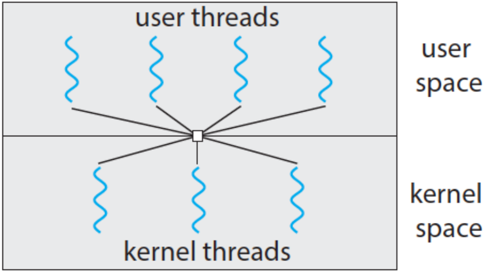
One-to-one model



- user thread ↔ kernel thread
- allows another thread to run when one makes a blocking system call
- allows multiple threads to run in parallel on multiprocessors
- possible drawback: a large number of threads may burden the performance of OS
- examples: Linux, Windows (contemporary version)

Multithreading models

Many-to-many model



Thread Libraries

A thread library provides the programmer with an API for creating and managing threads.

Well known libraries:

- POSIX Pthreads
- Windows thread library
- Java thread API