Threads and Multithread Model

Hui Chen [a]

[a]CUNY Brooklyn College

March 4, 2020

Outline

1. Overview and Motivation

2. Multithread Achitecture

3. Parallelism and Multicore Programming

4. Thread Model

Outline

1. Overview and Motivation

2. Multithread Achitecture

3. Parallelism and Multicore Programming

4. Thread Model

Process and Threads

Recall our discusion that multiple processes run concurrently ...

- P_1 on CPU, context switch, P_2 on CPU, context switch, P_3 on CPU
- The OS save the context of the current process, and load the context of the next process ...
- The OS maintains Process Control Blocks (PCB) for the processes

where a process consists of,

- an execution context, and
- an address space (program text, data, stack, and heap).

How about we let a process have

- *multiple execution contexts*, and
- an address sapce (program text, data, stack, and heap)?

which leads to Thread Control Block (TCB).

Benefits of Threads

- Improving responsiveness
- Easing resource sharing
- Can be made more economic (less overhead)
- Can be more scalable (to multicore architecture)

Outline

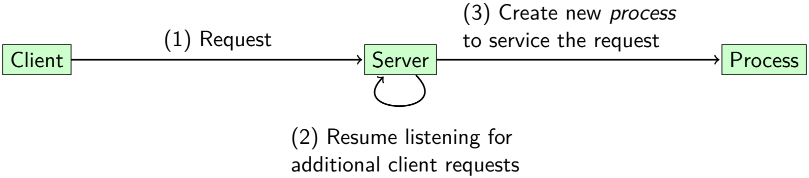
1 Overview and Motivation

2 Multithread Achitecture

3 Parallelism and Multicore Programming

4 Thread Model

Multithread vs Multiprocess Architecture



Client

(1) Request

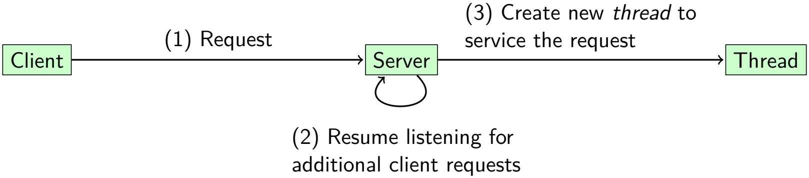
Server

(3) Create new *process* to service the request

Process

(2) Resume listening for additional client requests

Figure: Multiprocess architecture server



Client

(1) Request

Server

(3) Create new *thread* to service the request

Thread

(2) Resume listening for additional client requests

Figure: Multithread architecture server

Discussion Question

1. What benefits can we obtain from multithread architecture, but not from multiprocess architecture?
2. What benefits can we obtain from multiprocess architecture, but not from multithread architecture?

Outline

1. Overview and Motivation
2. Multithread Achitecture
3. Parallelism and Multicore Programming
4. Thread Model

Concurrency and Parallelism

Discuss,

- what concurrency is?
- what parallelism is?

Data and Task Paralleism

- Data parallelism. Distributes subsets of the same data across multiple cores, same operation on each
- Task parallelism. Distributing threads across cores, each thread performing unique operation

Amdahl's Law

$$\text{speedup} = \frac{1}{S + \frac{1-S}{N}} \tag{1}$$

where S is serial portion and N processing cores

- It identifies performance gains from adding additional cores to an application that has both serial and parallel components

Multicore Programming

Multicore or multiprocessor systems putting pressure on programmers, challenges include:

- Dividing activities
- Balance
- Data splitting
- Data dependency
- Testing and debugging

Outline

1 Overview and Motivation

2 Multithread Achitecture

3 Parallelism and Multicore Programming

4 Thread Model

Kernel and User Threads

- User threads.
    - Management done by user-level threads library.
    - TCBs in user process
    - Kernel threads are more expensive to create, and can support multiple processors
- Kernel threads
    - Management done by the kernel
    - TCBs in the kernel
    - User threads can be blocked by the process, less concurrency, in particular, on multiprocessor/multicore systems

Multithreading Models

- Many-to-One
- One-to-One
- Many-to-Many