COMP 3430

Operating Systems

May 22nd, 2019

Goals

By the end of today's lecture (and readings), you should be able to:

Pixabay License

- Compare and contrast *processes* and *threads*
- Write a *simple* program that uses threads
- Describe problems that come up with concurrent code
- Select appropriate strategies for dealing with concurrency
- Write a program that employs concurrency

Why use threads?

Pixabay License

Cost?

- Threads are *free*, right? Right???
  - … RIGHT?!
- Let's take a measurement .

Threads are *not* free

- Threads are *cheap* – compared to processes
- Think about it: When/what/where is the cost(s) with threads? pthread_create
  - Think a bit more: Can we *reduce* these costs?

Pools

- Idea: **Amortize** the costs
  - Try to minimize cost of creation over work done.
- Create many threads *when the program starts*.
  - Hand out *already created* threads when work needs to be done.
- COMP 3350: Thread pools are a **design pattern**.
- Let's look at some code.

Threads

- Processes → coarse-grained concurrency
  - Easier to use (`fork` off and forget)
  - **Everything**** is copied (see: `man 2 fork`)
- Threads → *fine*-grained concurrency
  - Hard*er* to use (we have to manage threads)
  - Cheap*er* to use (memory is shared) (?)

Pixabay License

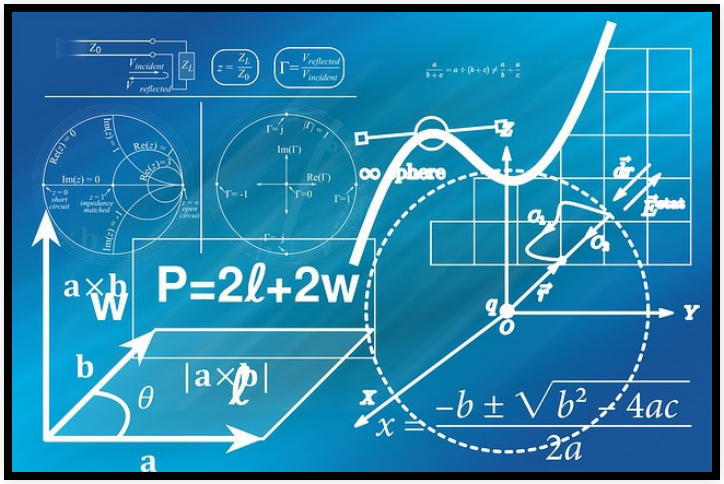
Problems with threads

- Threads are great! Right? … right?!
- With, uh, someone around you: Come up with some reasons why threads **are not** great.

Thread-world problems. (Pixabay License)

Solutions?

- There might be… *many* problems.
- Let's make some observations about problematic code.



Pixabay License

Summary

- Threads are great!
  - More flexible than coarse-grained processes.
  - More performant than processes (… in theory).

Summary

- Threads are **harder** to implement in practice.
  - Shared memory … `Segmentation fault`/the total is *what*???
  - Memory sharing.
  - Common address space.
  - Shared memory.
  - Mutual exclusion (… of shared memory).
  - Memory is shared among threads.
  - You have to manage shared memory.
  - All threads in a process share memory.
  - Memory. Shared. Memory. … shared.

Pixabay License

Next week

Let's take a look at the schedule.

Me: Mom, can we get $f(x)$?
Mom: No, we have $f(x)$ at home.
$f(x)$ at home: