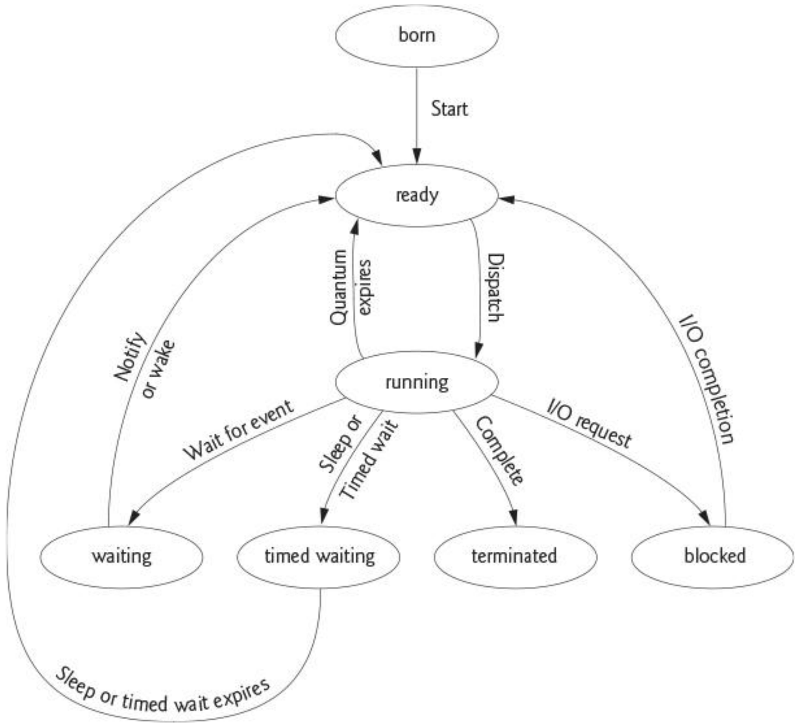
CSCI 325
Introduction to Parallel Systems and GPU Programming

Lecture 2
C++ Multithreading

Dr. Talgat Turanbekuly

Table of contents

C++ thread states



Born - thread begins its life cycle and remains until the program starts.

Ready - OS allocates processor time (quantum/timeslice) and sends the thread to **Running** state. Then the thread returns to the **Ready** state. Another thread is assigned to the processor.

Thread scheduling - allocating threads per time.

Waiting - thread A is paused until thread B completes and notifies thread A or specific time is passed.

Timed waiting - thread is in the state until time interval is passed or specific task is completed.

Blocked - the thread is waiting until I/O task is completed.

Terminated - when the thread completes its task.

Harvey Deitel, Paul Deitel and David Choffnes, "Chapter 4, Thread Concepts." Operating Systems, 3/e, p. 153. Upper Saddle River, NJ: Prentice Hall, 2004.

C++ multithreading

Lecture 2 examples

Thread Synchronization

It is possible to run parallel threads and execute several tasks.

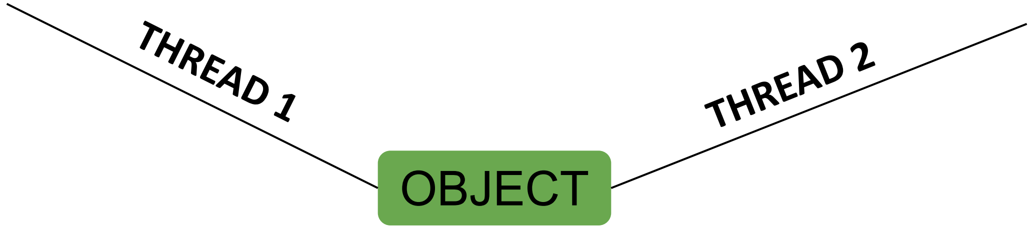
What if several threads try to change value of the same object?

Data Race

Thread-1 gets value of an object in order to change.

Thread-2 changes the value before thread-1 saves the object.

In this case thread-1 works with old value of the object.



Data Race

A ‘race condition’ may arise when two or more threads concurrently access the same memory location and at least one of the threads updates the location. It may lead to wrong results.

Thread Synchronization

problem solution:
Ensure only one thread at a time can access the object -
acquire the lock on the object.

One thread

1) gets value of the object
2) changes the value
3) saves new updated value to the object.

These three steps form **atomic operation** and cannot be divided.

Thread Synchronization

In order to manage multiple threads trying access one object:

thread synchronization

Only one thread is given access to the object

and

the rest threads are waiting.

mutual exclusion

Exercise

There are N text files in a directory. The program allows user to enter a word or a phrase and searches it in all of the files of the directory. The results should be written to another text file as the word is found. For example, there are 4 files: A.txt, B.txt, C.txt and D.txt. User enters “capital”. The program finds it in B.txt and then in C.txt after that in A.txt and D.txt. So the results.txt may have something like:

```
capital
B.txt line 2
C.txt line 7
A.txt line 30
D.txt line 29
```

Here in file A.txt the word was found a little later, but it took the queue to write to the results.txt earlier. It is ok if this happens.
Use multithreading for searching and mutex for writing the result.

Summary

There are several ways to declare and operate threads in C++

Thread synchronization allows only one thread to work with data

Thread synchronization prevents unpredictable results

Resources

Paul J. Deitel. C++20 Fundamentals, 3rd Edition. 2024;

Harvey Deitel, and Paul J. Deitel. C++20 for Programmers: An Object's-Natural Approach, 3rd Edition, 2022;

Anthony Williams. C++ Concurrency in Action, Second Edition, 2019;

HPC, CUDA internships, jobs

https://scholarshipdb.net/scholarships?q=cuda&listed=Last-24-hours

https://scholarshipdb.net/scholarships?q=gpu&listed=Last-24-hours

sign up to learn.nvidia.com