Intro to Parallel Computing

* Sequential Computing
  + The computer executes sequence of instructions of the program in order, one at a time
  + There is a single thread of execution
* Parallel Computing
  + Breaks programs into smaller sequential operations and performs them in parallel
    - Two or more processes (threads of execution)
  + Communication via shared memory or message passing
  + Synchronization of lines of execution
* Hardware
  + Single processor
  + Multiprocessors (shared memory)
  + Multicomputer/Cluster (distributed memory)
* File Downloading
  + Consider a client-server system for file downloads
    - Without parallelism
      * It is impossible to interact with the client while download is in progress
    - With parallelism
      * The user can interact with the client while a download is in progress
        + The server can handle multiple clients at the same time
* Examples in Algorithms
  + Algorithms can be broken in parallel parts
    - Merge sort
    - Computing the sum of the list by computing fragments of it in parallel
    - Work on independent parts of an array in parallel
* Parallel on a single processor?
  + When several processes are executed on a single processor they are not running in true parallel
    - Instead, they are executed in interleaved fashion
  + Parallel designs of a program can still be effective even if you only have a single processor
    - Many sequential programs spend considerable time blocked
    - This time can be used by another thread in your program (rather than being given by the OS to someone else’s program
* Compare Sequential vs Parallel Execution
  + We can measure performance achieved by parallelizing a given application over a sequential implementation
    - Speedup: ratio between the sequential time and parallel time
* Amdahl’s Law
  + Small number of sequential operations can significantly limit speedup achievable by a parallel computer
  + In other words, there are a lot of little processes that are not benefitting from having more processors
  + Equation:
    - Speedup(p) = 1 / [f + (1 – f)/p]
* Efficiency and Scalability
  + Efficiency
    - A measure of processor utilization, defined as the ratio of the speedup and the number of processors used
  + Scalability
    - Measure of its ability to increase performance as number if processors increase
    - A scalable system maintains efficiency as processors are added
* Many Examples on Week13: Thursday: Slides 4-10
* Parallelism with Threads
  + A thread is a light-weight process
    - Has its own stack and execution state but shares the address space with its parent
* Shared Memory Model
  + All threads have access to the same global, shared memory
  + Threads also have their own private data
  + Programmers are responsible for synchronizing access globally shared data
* Notes on Threads
  + Because threads within the same process share resources:
    - Changes made by one thread to shared system resources will be seen by all other threads
    - Reading and writing to the same memory locations is possible, and therefore requires explicit synchronization by the programmer
* Multithreaading
  + The process of executing multiple threads simultaneously
    - A multithreaded program contains two or more process that can run concurrently
    - Each process can handle a different task at the same time making optimal use of the available resources specially when your computer has multiple cpu’s