Parallel Programming: Advanced Topics

30 January 2007
CMPT370
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Review last time

- OpenMP
  - #pragma omp parallel: begin parallel section
    - #pragma omp for
    - #pragma omp sections
  - Shared vs. private variables
    - private(), reduction()
  - #pragma omp critical/single/barrier
  - OMP_NUM_THREADS, omp_get_num_threads()
  - Timing: omp_get_wtime()
  - schedule(static/dynamic/guided/runtime)

Addendum: timing

- <time.h> clock(): CPU time with tick resolution
  - Usually 100 or 1000 ticks/sec
- <time.h> time(): wall-clock time with second res
- <sys/time.h> gettimeofday():
  - Wall-clock time in seconds with tick resolution
- double omp_get_wtime():
  - Wall-clock time in seconds with tick resolution
  - Platform independent
  - Thread dependent

Communication Issues

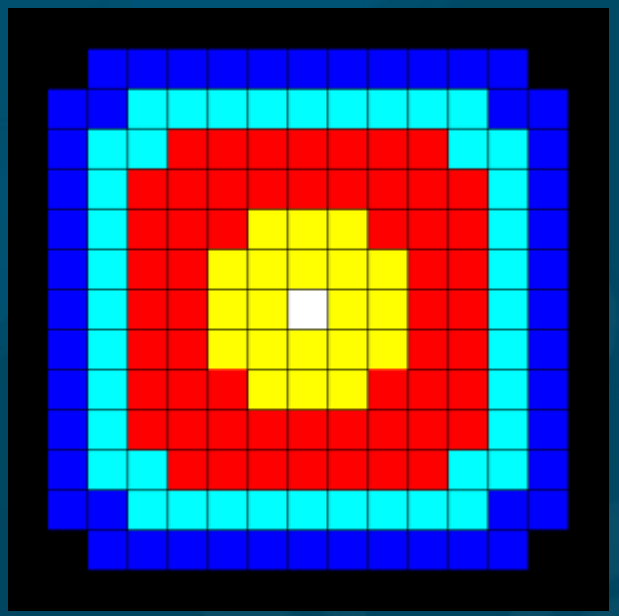
- I/O often the bottleneck: minimize communication
- Latency vs. bandwidth
  - Coarse-grain parallelism: e.g., FoldingAtHome
- Unicast (point-to-point) vs. multicast
- Synchronous (blocking) vs. asynchronous (non-blocking)
- Ease of programming
  - OpenMP abstracts away from programmer
  - MPI makes communication more explicit

Synchronization

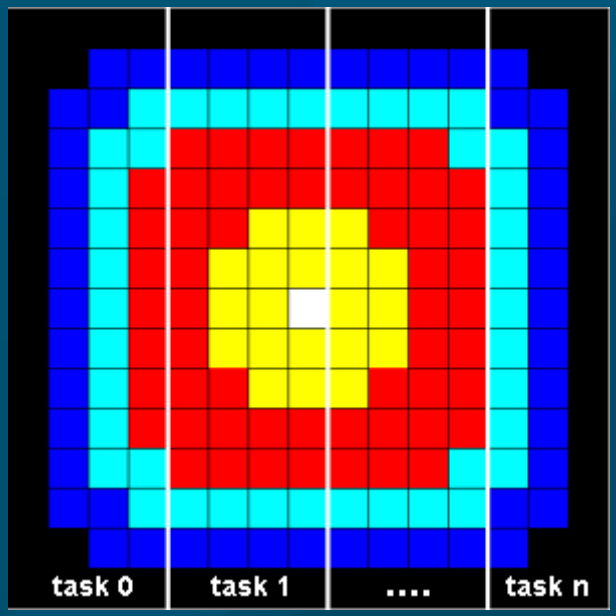
- Barrier
  - Wait for all tasks to catch up
    - Slowest task becomes weakest link
  - Implicit barrier at end of each parallel section
- Lock/semaphore/mutex
  - Only one thread can hold the lock at a time
  - Wait (block) for lock to free before moving on
  - e.g., #pragma omp critical
- Synchronous communications
  - Both parties must synchronize

Data Parallelism

- Some programs are not well parallelizable:
  - e.g., Fibonacci(n): dependencies
- Some can still be parallelized, but with some communication required:
  - Heat equation on a 2D grid
  - Each pixel $U_{x,y}$ +=
    - $C_x (U_{x-1,y} + U_{x+1,y} - 2*U_{x,y}) +$
    - $C_y (U_{x,y-1} + U_{x,y+1} - 2*U_{x,y})$
  - Used for blurring images



Heat equation: boundaries



- Divide work by region of image:
  - Data parallelism
- Interior of region can be done independently
- Boundaries need information from neighbouring threads
- Use non-blocking communication to send/receive boundary pixels from neighbours while processing interior

TODO

- Lab2 due next Tue 6Feb
  - Design + implement your own OpenMP program
  - Lab write-up