CpSc 418: Parallel Computation
Topics and Concepts

1 Performance

Measuring Performance:

- Throughput vs. latency
- $speedUp = \frac{T_{sequential}}{T_{parallel}}$
- Superlinear speed-up: what it is, typical causes.

Performance Loss

- Overhead: resources needed by the parallel algorithm that the sequential algorithm doesn't have.
- Other causes of performance loss: non-parallelizable code, contention, idle processors. Sometimes, the parallel algorithm is fundamentally different than the sequential one.
- Amdahl's Law
- Embarrassingly parallel applications.

2 Architecture

Superscalar architectures

- Instruction-level parallelism
- Tracking instruction dependencies with register renaming
- Executing during cache-misses.
- Examples of high instructions per cycle (IPC), and low IPC programs.
- The poor scalability of superscalar architectures.

Message passing machines

- Network topologies: ring, mesh, torus, hypercube.
- Cross-section bandwidth.
- Why high-dimensional machines become "all wire".
- Advantages of communication with nearby processors.
- Examples of large-scale message passing machines: supercomputers such as Jaguar and Tianhe 2.

Shared Memory Machines

- Cache coherence protocols: MESI
- Sequential consistency: the parallel execution appears *as if* all reads and writes were done one-at-a-time using a (very fast) global memory.
- Real computers make memory guarantees that aren't as strong as sequential conistenty – to get better performance using write buffers.
- Examples of shared-memory machines: multicore x86 CPUs, large commercial database servers, Intel's Xeon Phi.

3 Algorithms

Parallel models of computation

- PRAM: once popular with theoreticians, not very realistic.
  Valiant's find the max in $\log \log N$ parallel time.
- CTA: communication has cost λ.
  $\lambda \gg 1$ – models high communication cost for typical parallel software.
  Even shared memory machines have communication costs: cache miss penalties, fences, etc.
  For very large number of processors, need to consider network bandwidth constraints (see note about cross-section bandwidth above.

Simple algorithms for illustrating parallel computation

- Count 3's
- matrix multiply
- prime seive, Collatz length

Reduce and Scan

- Parallelism for associative operations
- Reduce computes a single result
- Scan computes a result for each prefix of the input array or list.
- Should be able to draw the tree depicting the computation.
- Should be able to write code (or pseudocode) for usinng reduce and scan.

Sorting

- The 0-1 principle
- The monotonicity lemma
- Some parallel sorting algorithms: odd-even exchange, shear sort, bitonic sort, the HW4.Q3 algorithm

Algorithms for coordinating parallel computation

- Producer-consumer: what it does, how it works, the role of mutexes and condition variables.
- Mutual exclusion: Dekker, Peterson, Lamport.
- Invariants

4 Applications

Erlang

GPUs

Map-Reduce

PReach