

Introduction to High Performance Computing

Welcome

1. Explore big ideas about how to extract parallelism and data locality from algorithms and data structures

Readiness Survey

Philosophy and Logistics

1. Lectures: Develop intuition
2. Readings: Formalize intuition mathematically
3. Projects: Writing real code
 - Don't understand an algorithm or data structure until actually implemented

What is HPC?

1. High performance is an ambiguous term because it's relative
 - Supercomputing is also used to refer to the same concepts

Supercomputing in a Nutshell

1. Given a computational problem and a machine, how do you compute at the absolute limits of scale and speed?
 - Limits come from physics (speed of light or amount of energy, power, or heat dissipated by a system)

Topics

1. Sequential or serial RAM model: Single serial processor connected to memory
 - Issues instructions that operate on data
 - Operands always live in memory
 - Assume all instructions have a cost that is bounded by some constant
 - $O(n \log(n))$
2. Parallel RAM model: Multiple processors connected to shared memory
 - Still assume constant cost per operation
 - Processors can communicate via shared variables
 - Reduce total cost by up to P, where P is the number of processors
 - $O(n * \log(n) / P)$
3. Distributed memory network model: Interconnected network of RAMs
 - Each processor has separate memory, no processor can read or write to any other processor's memory
 - Instead, must communicate over a network
 - $\text{Cost}(n, P) = a * f(n, P) + B * g(n, p)$
 - Need to account for number of messages and total volume of communication
4. Two-level I/O Memory: Multiple processors share memory, but there's at least one additional level of faster memory sitting between the processor and the slower main memory
 - How much data needs to move between the memory and processor using the intermediate "scratch pad" memory
 - Cache, virtual memory