Scientific Computing/Engineers – COSC 594 –

Lecture 1:
Overview of High-Performance Computing

Jack Dongarra
Electrical Engineering and Computer Science Department
University of Tennessee
Example of typical parallel machine
Example of typical parallel machine
Example of typical parallel machine

Shared memory programming between processes on a board and a combination of shared memory and distributed memory programming between nodes and cabinets
Example of typical parallel machine

Combination of shared memory and distributed memory programming
## November 2012: The TOP10

<table>
<thead>
<tr>
<th>Rank</th>
<th>Site</th>
<th>Computer</th>
<th>Country</th>
<th>Cores</th>
<th>Rmax [Pflops]</th>
<th>% of Peak</th>
<th>Power [MW]</th>
<th>MFlops/Watt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DOE / OS Oak Ridge Nat Lab</td>
<td>Titan, Cray XK7 (16C) + Nvidia Kepler GPU (14c) + custom</td>
<td>USA</td>
<td>560,640</td>
<td>17.6</td>
<td>66</td>
<td>8.3</td>
<td>2120</td>
</tr>
<tr>
<td>2</td>
<td>DOE / NNSA L Livermore Nat Lab</td>
<td>Sequoia, BlueGene/Q (16c) + custom</td>
<td>USA</td>
<td>1,572,864</td>
<td>16.3</td>
<td>81</td>
<td>7.9</td>
<td>2063</td>
</tr>
<tr>
<td>3</td>
<td>RIKEN Advanced Inst for Comp Sci</td>
<td>K computer Fujitsu SPARC64 VIIIfx (8c) + custom</td>
<td>Japan</td>
<td>705,024</td>
<td>10.5</td>
<td>93</td>
<td>12.7</td>
<td>827</td>
</tr>
<tr>
<td>4</td>
<td>DOE / OS Argonne Nat Lab</td>
<td>Mira, BlueGene/Q (16c) + custom</td>
<td>USA</td>
<td>786,432</td>
<td>8.16</td>
<td>81</td>
<td>3.95</td>
<td>2066</td>
</tr>
<tr>
<td>5</td>
<td>Forschungszentrum Juelich</td>
<td>JuQUEEN, BlueGene/Q (16c) + custom</td>
<td>Germany</td>
<td>393,216</td>
<td>4.14</td>
<td>82</td>
<td>1.97</td>
<td>2102</td>
</tr>
<tr>
<td>6</td>
<td>Leibniz Rechenzentrum</td>
<td>SuperMUC, Intel (8c) + IB</td>
<td>Germany</td>
<td>147,456</td>
<td>2.90</td>
<td>90*</td>
<td>3.42</td>
<td>848</td>
</tr>
<tr>
<td>7</td>
<td>Texas Advanced Computing Center</td>
<td>Stampede, Dell Intel (8) + Intel Xeon Phi (61) + IB</td>
<td>USA</td>
<td>204,900</td>
<td>2.66</td>
<td>67</td>
<td>3.3</td>
<td>806</td>
</tr>
<tr>
<td>8</td>
<td>Nat. SuperComputer Center in Tianjin</td>
<td>Tianhe-1A, NUDT Intel (6c) + Nvidia Fermi GPU (14c) + custom</td>
<td>China</td>
<td>186,368</td>
<td>2.57</td>
<td>55</td>
<td>4.04</td>
<td>636</td>
</tr>
<tr>
<td>9</td>
<td>CINECA</td>
<td>Fermi, BlueGene/Q (16c) + custom</td>
<td>Italy</td>
<td>163,840</td>
<td>1.73</td>
<td>82</td>
<td>0.822</td>
<td>2105</td>
</tr>
<tr>
<td>10</td>
<td>IBM</td>
<td>DARPA Trial System, Power7 (8C) + custom</td>
<td>USA</td>
<td>63,360</td>
<td>1.51</td>
<td>78</td>
<td>0.358</td>
<td>422</td>
</tr>
</tbody>
</table>
## November 2012: The TOP10

<table>
<thead>
<tr>
<th>Rank</th>
<th>Site</th>
<th>Computer</th>
<th>Country</th>
<th>Cores</th>
<th>Rmax [Pflops]</th>
<th>% of Peak</th>
<th>Power [MW]</th>
<th>MFlops/Watt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DOE / OS Oak Ridge Nat Lab</td>
<td>Titan, Cray XK7 (16C) + Nvidia Kepler GPU (14c) + custom</td>
<td>USA</td>
<td>560,640</td>
<td>17.6</td>
<td>66</td>
<td>8.3</td>
<td>2120</td>
</tr>
<tr>
<td>2</td>
<td>DOE / NNSA L Livermore Nat Lab</td>
<td>Sequoia, BlueGene/Q (16c) + custom</td>
<td>USA</td>
<td>1,572,864</td>
<td>16.3</td>
<td>81</td>
<td>7.9</td>
<td>2063</td>
</tr>
<tr>
<td>3</td>
<td>RIKEN Advanced Inst for Comp Sci</td>
<td>K computer Fujitsu SPARC64 VIIIIfx (8c) + custom</td>
<td>Japan</td>
<td>705,024</td>
<td>10.5</td>
<td>93</td>
<td>12.7</td>
<td>827</td>
</tr>
<tr>
<td>4</td>
<td>DOE / OS Argonne Nat Lab</td>
<td>Mira, BlueGene/Q (16c) + custom</td>
<td>USA</td>
<td>786,432</td>
<td>8.16</td>
<td>81</td>
<td>3.95</td>
<td>2066</td>
</tr>
<tr>
<td>5</td>
<td>Forschungszentrum Juelich</td>
<td>JuQUEEN, BlueGene/Q (16c) + custom</td>
<td>Germany</td>
<td>393,216</td>
<td>4.14</td>
<td>82</td>
<td>1.97</td>
<td>2102</td>
</tr>
<tr>
<td>6</td>
<td>Leibniz Rechenzentrum</td>
<td>SuperMUC, Intel (8c) + IB</td>
<td>Germany</td>
<td>147,456</td>
<td>2.90</td>
<td>90*</td>
<td>3.42</td>
<td>848</td>
</tr>
<tr>
<td>7</td>
<td>Texas Advanced Computing Center</td>
<td>Stampede, Dell Intel (8) + Intel Xeon Phi (61) + IB</td>
<td>USA</td>
<td>204,900</td>
<td>2.66</td>
<td>67</td>
<td>3.3</td>
<td>806</td>
</tr>
<tr>
<td>8</td>
<td>Nat. SuperComputer Center in Tianjin</td>
<td>Tianhe-1A, NUDT Intel (6c) + Nvidia Fermi GPU (14c) + custom</td>
<td>China</td>
<td>186,368</td>
<td>2.57</td>
<td>55</td>
<td>4.04</td>
<td>636</td>
</tr>
<tr>
<td>9</td>
<td>CINECA</td>
<td>Fermi, BlueGene/Q (16c) + custom</td>
<td>Italy</td>
<td>163,840</td>
<td>1.73</td>
<td>82</td>
<td>.822</td>
<td>2105</td>
</tr>
<tr>
<td>10</td>
<td>IBM</td>
<td>DARPA Trial System, Power7 (8C) + custom</td>
<td>USA</td>
<td>63,360</td>
<td>1.51</td>
<td>78</td>
<td>.358</td>
<td>422</td>
</tr>
</tbody>
</table>

500 Slovak Academy Sci IBM Power 7 Slovak Rep 3,074 .077 81
ORNL’s “Titan” Hybrid System: Cray XK7 with AMD Opteron and NVIDIA Tesla processors

SYSTEM SPECIFICATIONS:
• Peak performance of 27 PF
  • 24.5 Pflop/s GPU + 2.6 Pflop/s AMD
• 18,688 Compute Nodes each with:
  • 16-Core AMD Opteron CPU
  • NVIDIA Tesla “K20x” GPU
  • 32 + 6 GB memory
• 512 Service and I/O nodes
• 200 Cabinets
• 710 TB total system memory
• Cray Gemini 3D Torus Interconnect
• 9 MW peak power

4,352 ft²
404 m²
### XK7 Compute Node Characteristics

<table>
<thead>
<tr>
<th>Feature</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMD Opteron 6274 Interlagos</td>
<td>16 core processor</td>
</tr>
<tr>
<td>Tesla K20x @ 1311 GF</td>
<td></td>
</tr>
<tr>
<td>Host Memory</td>
<td>32GB</td>
</tr>
<tr>
<td></td>
<td>1600 MHz DDR3</td>
</tr>
<tr>
<td>Tesla K20x Memory</td>
<td>6GB GDDR5</td>
</tr>
<tr>
<td>Gemini High Speed Interconnect</td>
<td></td>
</tr>
</tbody>
</table>

Slide courtesy of Cray, Inc.
Titan: Cray XK7 System

System:
- 200 Cabinets
- 18,688 Nodes
- 27 PF
- 710 TB

Cabinet:
- 24 Boards
- 96 Nodes
- 139 TF
- 3.6 TB

Board:
- 4 Compute Nodes
- 5.8 TF
- 152 GB

Compute Node:
- 1.45 TF
- 38 GB
New Linpack run with 705,024 cores at 10.51 Pflop/s (88,128 CPUs), 12.7 MW; 29.5 hours
Fujitsu to have a 100 Pflop/s system in 2014
Main configuration of TH-1A system

- 7,168 compute nodes (YH-X5670-FEP)
  - 2 six-core CPU and 1 GPU per node
  - CPU: Xeon X5670 (Westmere)
    - Processor speed - 2.93GHz
  - GPU: nVIDIA M2050
    - Connected with CPU by PCI-E
  - 32GB memory per node
  - 2U height

\[
\text{7168(n nodes)} \times 2(\text{CPU}) \times 2.93(\text{GHz}) \times 6(\text{Cores}) \times 4 = 1.008 \text{PFlops}
\]

\[
\text{7168(n nodes)} \times 1(\text{GPU}) \times 1.15(\text{GHz}) \times 448(\text{CUDA Cores}) = 3.692 \text{PFlops}
\]

Total: 4,701,061 GFlops
China

First Chinese Supercomputer to use a Chinese Processor
- Sunway BlueLight MPP
- ShenWei SW1600 processor, 16 core, 65 nm, fabbed in China
- 125 Gflop/s peak
- #14 with 139,364 cores, .796 Pflop/s & 1.07 Pflop/s Peak
- Power Efficiency 741 Mflops/W

Coming soon, Loongson (Godson) processor
- 8-core, 65nm Loongson 3B processor runs at 1.05 GHz, with a peak performance of 128 Gflop/s
10+ Pflop/s Systems in the States

- DOE Funded, Titan at ORNL, Based on Cray XK7 w/AMD & Nvidia accelerators,
  - 27 Pflop/s, 2012
- DOE Funded, Sequoia at Lawrence Livermore Nat. Lab, Based on IBM’s BG/Q,
  - 20 Pflop/s, 2012
- DOE Funded, BG/Q at Argonne National Lab, Based on IBM’s BG/Q,
  - 10 Pflop/s, 2012
- NSF Funded, Blue Waters at University of Illinois UC, Based Cray XE6/XK6 hybrid,
  - 11.5 Pflop/s, 2012
- NSF Funded, U of Texas, Austin, Based on Dell/Intel MIC,
  - 4 Pflop/s, 2012
Today’s Multicores
99% of Top500 Systems Are Based on Multicore

- Sun Niagara2 (8 cores)
- IBM Power 7 (8 cores)
- Fujitsu Venus (16 cores)
- IBM Cell (9 cores)
- AMD Interlagos (16 cores)
- IBM BG/Q (18 cores)
- Intel Westmere (10 cores)
- Intel Xeon Phi (60 cores)
Industrial Use of Supercomputers

- Of the 500 Fastest Supercomputer
  - Worldwide, Industrial Use is ~ 50%

- Aerospace
- Automotive
- Biology
- CFD
- Database
- Defense
- Digital Content Creation
- Digital Media
- Electronics
- Energy
- Environment
- Finance
- Gaming
- Geophysics
- Image Proc./Rendering
- Information Processing Service
- Information Service
- Life Science
- Media
- Medicine
- Pharmaceuticals
- Research
- Retail
- Semiconductor
- Telecom
- Weather and Climate Research
- Weather Forecasting
Power is an Industry Wide Problem

“Hiding in Plain Sight, Google Seeks More Power”,
by John Markoff, June 14, 2006

Google facilities
- leveraging hydroelectric power
- old aluminum plants

Microsoft and Yahoo are building big data centers upstream in Wenatchee and Quincy, Wash.
- To keep up with Google, which means they need cheap electricity and readily accessible data networking

Microsoft Quincy, Wash. 470,000 Sq Ft, 47MW!
Commercial Data Centers

Facebook  
300,000 sq ft  
1.5 cents per kW hour  
Prineville OR

Microsoft 700,000 sq ft in Chicago

Apple 500,000 sq ft in Rural NC  
4 cents kW/h
**COOLING:** High-efficiency water-based cooling systems—less energy-intensive than traditional chillers—circulate cold water through the containers to remove heat, eliminating the need for air-conditioned rooms.

**STRUCTURE:** A 24,000-square-meter facility houses 400 containers. Delivered by trucks, the containers attach to a spine infrastructure that feeds network connectivity, power, and water. The data center has no conventional raised floors.

**POWER:** Two power substations feed a total of 300 megawatts to the data center, with 200 MW used for computing equipment and 100 MW for cooling and electrical losses. Batteries and generators provide backup power.

**CONTAINER:** Each 67.5-cubic-meter container houses 2500 servers, about 10 times as many as conventional data centers pack in the same space. Each container integrates computing, networking, power, and cooling systems.
Future Computer Systems

- Most likely be a hybrid design
- Think standard multicore chips and accelerator (GPUs)
- Today accelerators are attached
- Next generation more integrated
- Intel’s Xeon Phi
  - 244 “cores”
- AMD’s Fusion
  - Multicore with embedded graphics ATI
- Nvidia’s Kepler with 2688 Cuda cores
  - Project Denver plans to develop an integrated chip using ARM architecture in 2013.
Commodity plus Accelerator Today

**Commodity**
- Intel Xeon
- 8 cores
- 3 GHz
- 8*4 ops/cycle
- 96 Gflop/s (DP)

**Accelerator (GPU)**
- Nvidia K20X “Kepler”
- 2688 “Cuda cores”
- 0.732 GHz
- 2688*2/3 ops/cycle
- 1.31 Tflop/s (DP)

**Interconnect**
- PCI-X 16 lane
- 64 Gb/s (8 GB/s)
- 1 GW/s

192 Cuda cores/SMX
2688 “Cuda cores”
We Have Seen This Before

- Floating Point Systems FPS-164/MAX Supercomputer (1976)
- Intel Math Co-processor (1980)
- Weitek Math Co-processor (1981)
Challenges of using GPUs

- **High levels of parallelism**
  *Many GPU cores, serial kernel execution*
  [e.g. 240 in the Nvidia Tesla; up to 512 in Fermi - to have concurrent kernel execution]

- **Hybrid/heterogeneous architectures**
  *Match algorithmic requirements to architectural strengths*
  [e.g. small, non-parallelizable tasks to run on CPU, large and parallelizable on GPU]

- **Compute vs communication gap**
  *Exponentially growing gap; persistent challenge*
  [Processor speed improves 59%, memory bandwidth 23%, latency 5.5%]
  [on all levels, e.g. a GPU Tesla C1070 (4 x C1060) has compute power of O(1,000) Gflop/s but GPUs communicate through the CPU using O(1) GB/s connection]
Moore’s Law is Alive and Well

Data from Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten, and Krste Asanović
Slide from Kathy Yelick
But Clock Frequency Scaling Replaced by Scaling Cores / Chip

15 Years of exponential growth ~2x year has ended

Transistors (in Thousands)
Frequency (MHz)
Cores

Data from Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten, and Krste Asanović
Slide from Kathy Yelick
Performance Has Also Slowed, Along with Power

Power is the root cause of all this

A hardware issue just became a software problem

Data from Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten, and Krste Asanović
Slide from Kathy Yelick
Power Cost of Frequency

- Power $\propto$ Voltage$^2$ x Frequency \( (V^2F) \)
- Frequency $\propto$ Voltage
- Power $\propto$ Frequency$^3$

<table>
<thead>
<tr>
<th></th>
<th>Cores</th>
<th>V</th>
<th>Freq</th>
<th>Perf</th>
<th>Power</th>
<th>PE (Bops/watt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Superscalar</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>“New” Superscalar</td>
<td>1X</td>
<td>1.5X</td>
<td>1.5X</td>
<td>1.5X</td>
<td>3.3X</td>
<td>0.45X</td>
</tr>
</tbody>
</table>
### Power Cost of Frequency

- **Power** $\propto$ **Voltage**$^2$ x **Frequency** ($V^2F$)
- **Frequency** $\propto$ **Voltage**
- **Power** $\propto$ **Frequency**$^3$

<table>
<thead>
<tr>
<th></th>
<th>Cores</th>
<th>V</th>
<th>Freq</th>
<th>Perf</th>
<th>Power</th>
<th>PE (Bops/watt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Superscalar</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>“New” Superscalar</td>
<td>1X</td>
<td>1.5X</td>
<td>1.5X</td>
<td>1.5X</td>
<td>3.3X</td>
<td>0.45X</td>
</tr>
<tr>
<td>Multicore</td>
<td>2X</td>
<td>0.75X</td>
<td>0.75X</td>
<td>1.5X</td>
<td>0.8X</td>
<td>1.88X</td>
</tr>
</tbody>
</table>

(Bigger # is better)

50% more performance with 20% less power

Preferable to use multiple slower devices, than one superfast device
Problem with Multicore

- As we put more processing power on the multicore chip, one of the problems is getting the data to the cores.

- Next generation will be more integrated, 3D design with a photonic network.
Moore’s Law Reinterpreted

- Number of cores per chip doubles every 2 year, while clock speed decreases (not increases).
  - Need to deal with systems with millions of concurrent threads
    - Future generation will have billions of threads!
  - Need to be able to easily replace inter-chip parallelism with intro-chip parallelism
- Number of threads of execution doubles every 2 year
The High Cost of Data Movement

- Flop/s or percentage of peak flop/s become much less relevant

<table>
<thead>
<tr>
<th>Approximate power costs (in picoJoules)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DP FMADD flop</td>
</tr>
<tr>
<td>100 pJ</td>
</tr>
<tr>
<td>DP DRAM read</td>
</tr>
<tr>
<td>4800 pJ</td>
</tr>
<tr>
<td>Local Interconnect</td>
</tr>
<tr>
<td>7500 pJ</td>
</tr>
<tr>
<td>Cross System</td>
</tr>
<tr>
<td>9000 pJ</td>
</tr>
</tbody>
</table>

Source: John Shalf, LBNL

- Algorithms & Software: minimize data movement; perform more work per unit data movement.
Energy Cost Challenge

- At ~$1M per MW energy costs are substantial
  - 10 Pflop/s in 2011 uses ~10 MWs
  - 1 Eflop/s in 2018 > 100 MWs

- DOE Target: 1 Eflop/s in 2018 at 20 MWs
Major Changes to Software

• Must rethink the design of our software
  ▪ Another disruptive technology
    • Similar to what happened with cluster computing and message passing
  ▪ Rethink and rewrite the applications, algorithms, and software
Exascale Computing

- Exascale systems are likely feasible by 2017±2
- 10-100 Million processing elements (cores or mini-cores) with chips perhaps as dense as 1,000 cores per socket, clock rates will grow more slowly
- 3D packaging likely
- Large-scale optics based interconnects
- 10-100 PB of aggregate memory
- Hardware and software based fault management
- Heterogeneous cores
- Performance per watt — stretch goal 100 GF/watt of sustained performance \( \Rightarrow >> 10 - 100 \text{ MW Exascale system} \)
- Power, area and capital costs will be significantly higher than for today’s fastest systems

Google: exascale computing study
Factors that Necessitate Redesign of Our Software

- **Steepness of the ascent from terascale to petascale to exascale**
- **Extreme parallelism and hybrid design**
  - Preparing for million/billion way parallelism
- **Tightening memory/bandwidth bottleneck**
  - Limits on power/clock speed implication on multicore
  - Reducing communication will become much more intense
  - Memory per core changes, byte-to-flop ratio will change
- **Necessary Fault Tolerance**
  - MTTF will drop
  - Checkpoint/restart has limitations

Software infrastructure does not exist today
Why Fast Machines Run Slow

- **Latency**
  - Waiting for access to memory or other parts of the system

- **Overhead**
  - Extra work that has to be done to manage program concurrency and parallel resources the real work you want to perform

- **Starvation**
  - Not enough work to do due to insufficient parallelism or poor load balancing among distributed resources

- **Contention**
  - Delays due to fighting over what task gets to use a shared resource next. Network bandwidth is a major constraint.
Latency in a Single System

THE WALL
Processor-DRAM Memory Gap

\[ \text{µProc} \]
- 60%/yr.
  - (2X/1.5yr)

\[ \text{DRAM} \]
- 9%/yr.
  - (2X/10 yrs)

"Moore’s Law"

Processor-Memory Performance Gap:
- (grows 50% / year)
## Memory hierarchy

- **Typical latencies for today’s technology**

<table>
<thead>
<tr>
<th>Hierarchy</th>
<th>Processor clocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register</td>
<td>1</td>
</tr>
<tr>
<td>L1 cache</td>
<td>2-3</td>
</tr>
<tr>
<td>L2 cache</td>
<td>6-12</td>
</tr>
<tr>
<td>L3 cache</td>
<td>14-40</td>
</tr>
<tr>
<td>Near memory</td>
<td>100-300</td>
</tr>
<tr>
<td>Far memory</td>
<td>300-900</td>
</tr>
<tr>
<td>Remote memory</td>
<td>$O(10^3)$</td>
</tr>
<tr>
<td>Message-passing</td>
<td>$O(10^3)-O(10^4)$</td>
</tr>
</tbody>
</table>
Memory Hierarchy

- Most programs have a high degree of **locality** in their accesses
  - spatial locality: accessing things nearby previous accesses
  - temporal locality: reusing an item that was previously accessed
- Memory hierarchy tries to exploit locality

<table>
<thead>
<tr>
<th>Speed</th>
<th>1ns</th>
<th>10ns</th>
<th>100ns</th>
<th>10ms</th>
<th>10sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>B</td>
<td>KB</td>
<td>MB</td>
<td>GB</td>
<td>TB</td>
</tr>
</tbody>
</table>
Percentage of peak

- A rule of thumb that often applies
  - A contemporary processor, for a spectrum of applications, delivers (i.e., sustains) 10% or less of peak performance
- There are exceptions to this rule, in both directions
- Why such low efficiency?
- There are two primary reasons behind the disappointing percentage of peak
  - IPC (in)efficiency
  - Memory (in)efficiency
Today the theoretical IPC (instructions per cycle) is 4 in most contemporary processors.

Detailed analysis for a spectrum of applications indicates that the average IPC is 1.2–1.4.

We are leaving ~75% of the possible performance on the table.
Finding Enough Parallelism

- Suppose only part of an application seems parallel

- Amdahl’s law
  - Let $f_s$ be the fraction of work done sequentially, $(1-f_s)$ is fraction parallelizable
  - $N$ = number of processors

- Even if the parallel part speeds up perfectly may be limited by the sequential part
Amdahl’s Law

Amdahl’s Law places a strict limit on the speedup that can be realized by using multiple processors. Two equivalent expressions for Amdahl’s Law are given below:

\[ t_N = \left( \frac{f_p}{N} + f_s \right) t_1 \]  Effect of multiple processors on run time

\[ S = \frac{1}{(f_s + f_p/N)} \]  Effect of multiple processors on speedup

Where:

\[ f_s = \text{serial fraction of code} \]
\[ f_p = \text{parallel fraction of code} = 1 - f_s \]
\[ N = \text{number of processors} \]
Illustration of Amdahl’s Law

It takes only a small fraction of serial content in a code to degrade the parallel performance. It is essential to determine the scaling behavior of your code before doing production runs using large numbers of processors.
Overhead of Parallelism

- Given enough parallel work, this is the biggest barrier to getting desired speedup
- Parallelism overheads include:
  - cost of starting a thread or process
  - cost of communicating shared data
  - cost of synchronizing
  - extra (redundant) computation
- Each of these can be in the range of milliseconds (=millions of flops) on some systems
- Tradeoff: Algorithm needs sufficiently large units of work to run fast in parallel (I.e. large granularity), but not so large that there is not enough parallel work