CS 594 - 007
Scientific Computing for Engineers

Introduction to High Performance Computing

Jack Dongarra
University of Tennessee
Oak Ridge National Lab
University of Manchester

Simulation: The Third Pillar of Science

- Traditional scientific and engineering paradigm:
  1) Do theory or paper design.
  2) Perform experiments or build system.
- Limitations:
  - Too difficult -- build large wind tunnels.
  - Too expensive -- crash a passenger jet.
  - Too slow -- wait for climate or galactic evolution.
  - Too dangerous -- weapons, drug design, climate experimentation.
- Computational science paradigm:
  3) Use high performance computer systems to simulate the phenomenon
     » Base on known physical laws and efficient numerical methods.
Computational Science

Applications to Energy

Turbulence
Understanding the statistical geometry of turbulent dispersion of pollutants in the environment.

Energy Storage
Understanding the storage and flow of energy in next-generation nanostructured carbon tube supercapacitors.

Biofuels
A comprehensive simulation model of lignocellulosic biomass to understand the bottleneck to sustainable and economical ethanol production.

Nuclear Energy
High-fidelity predictive simulation tools for the design of next-generation nuclear reactors to safely increase operating margins.

Smart Truck
Aerodynamic forces account for ~53% of long haul truck fuel use. ORNL’s Jaguar predicted 12% drag reduction and yielded EPA-certified 6.9% increase in fuel efficiency.

Nano Science
Understanding the atomic and electronic properties of nanostructures in next-generation photovoltaic solar cell materials.

Source: Steven E. Koonin, DOE
Computational Science Fuses Three Distinct Elements:
Computational Science As An Emerging Academic Pursuit

- Many Programs in Computational Science
  - College for Computing
    - Georgia Tech; NJIT; CMU; ...
  - Degrees
    - Rice, Utah, UCSB; ...
  - Minor
    - Penn State, U Wisc, SUNY Brockport
  - Certificate
    - Old Dominion, U of Georgia, Boston U, ...
  - Concentration
    - Cornell, Northeastern, Colorado State, ...
  - Courses
At the University of Tennessee

- A few years ago there was a discussion to create a program in Computational Science
- This program evolved out of a set of meetings and discussions with faculty, students, and administration.
- Modeled on a similar minor degree program in the Statistics Department on campus.
- Appeared in the 2007-2008 Graduate Catalog
Graduate Minor in Computational Science

- Students in one of the three general areas of Computational Science:
  - Applied Mathematics,
  - Computer related, or
  - a Domain Science

  will become exposed to and better versed in the other two areas that are currently outside their “home” area.

- A pool of courses which deals with each of the three main areas has been put together by participating department for students to select from.

- Interdisciplinary Graduate Minor in Computational Science (IGMCS)
IGMCS: Requirements

- The Minor requires a combination of course work from three disciplines - Computer related, Mathematics/Stat, and a participating Science/Engineering domain (e.g., Chemical Engineering, Chemistry, Physics).

- At the Masters level a minor in Computational Science will require 9 hours (3 courses) from the pools.
  - At least 6 hours (2 courses) must be taken outside the student’s home area.
  - Students must take at least 3 hours (1 course) from each of the 2 non-home areas.

- At the Doctoral level a minor in computation science will require 15 hours (5 courses) from the pools.
  - At least 9 hours (3 courses) must be taken outside the student’s home area.
  - Students must take at least 3 hours (1 course) from each of the 2 non-home areas.
IGMCS Process for Students

1. A student, with guidance from their faculty advisor, lays out a program of courses
2. Next, discussion with department’s IGMCS liaison
3. Form generated with courses to be taken
4. Form is submitted for approval by the IGMCS Program Committee
<table>
<thead>
<tr>
<th>Department</th>
<th>IGMCS Liaison</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthropology</td>
<td>Dr. Alex Bentley</td>
<td><a href="mailto:rabentley@utk.edu">rabentley@utk.edu</a></td>
</tr>
<tr>
<td>Biochem &amp; Cell and Mole Bio</td>
<td>Dr. Tongye Shen</td>
<td><a href="mailto:tshen@utk.edu">tshen@utk.edu</a></td>
</tr>
<tr>
<td>Chemical Engineering</td>
<td>Dr. Steven Abel</td>
<td><a href="mailto:abel@utk.edu">abel@utk.edu</a></td>
</tr>
<tr>
<td>Chemistry</td>
<td>Dr. Robert Hinde</td>
<td><a href="mailto:rhinde@utk.edu">rhinde@utk.edu</a></td>
</tr>
<tr>
<td>Civil &amp; Envir. Eng.</td>
<td>Dr. Joshua Fu</td>
<td><a href="mailto:jsfu@utk.edu">jsfu@utk.edu</a></td>
</tr>
<tr>
<td>Earth &amp; Planetary Sci</td>
<td>Dr. Edmund Perfect</td>
<td><a href="mailto:eperfect@utk.edu">eperfect@utk.edu</a></td>
</tr>
<tr>
<td>Ecology &amp; Evol. Biology</td>
<td>Dr. Paul Armworth</td>
<td><a href="mailto:p.armsworth@utk.edu">p.armsworth@utk.edu</a></td>
</tr>
<tr>
<td>EECS</td>
<td>Dr. Jack Dongarra</td>
<td><a href="mailto:dongarra@eecs.utk.edu">dongarra@eecs.utk.edu</a></td>
</tr>
<tr>
<td>Genome Sci &amp; Tech</td>
<td>Dr. Tongye Shen</td>
<td><a href="mailto:tshen@utk.edu">tshen@utk.edu</a></td>
</tr>
<tr>
<td>Geography</td>
<td>Dr. Nicholas Nagle</td>
<td><a href="mailto:nnagle@utk.edu">nnagle@utk.edu</a></td>
</tr>
<tr>
<td>Industrial and Systems Engineering</td>
<td>Dr. Jim Ostrowski</td>
<td><a href="mailto:jostrows@utk.edu">jostrows@utk.edu</a></td>
</tr>
<tr>
<td>Information Science</td>
<td>Dr. Peiling Wang</td>
<td><a href="mailto:pellingw@utk.edu">pellingw@utk.edu</a></td>
</tr>
<tr>
<td>Material Science &amp; Eng</td>
<td>Dr. David Keffer</td>
<td><a href="mailto:dkeffer@utk.edu">dkeffer@utk.edu</a></td>
</tr>
<tr>
<td>Mathematics</td>
<td>Dr. Vasilios Alexiades</td>
<td><a href="mailto:alexiades@utk.edu">alexiades@utk.edu</a></td>
</tr>
<tr>
<td>Mech, Aero &amp; Biomed Eng</td>
<td>Dr. Kivanc Ekici</td>
<td><a href="mailto:ekici@utk.edu">ekici@utk.edu</a></td>
</tr>
<tr>
<td>Microbiology</td>
<td>Dr. Igor Jouline</td>
<td><a href="mailto:ijouline@utk.edu">ijouline@utk.edu</a></td>
</tr>
<tr>
<td>Nuclear Engineering</td>
<td>Dr. Ivan Maldonado</td>
<td><a href="mailto:ivan.maldonado@utk.edu">ivan.maldonado@utk.edu</a></td>
</tr>
<tr>
<td>Physics</td>
<td>Dr. Thomas Papenbrock</td>
<td><a href="mailto:tpapenbr@utk.edu">tpapenbr@utk.edu</a></td>
</tr>
<tr>
<td>Statistics</td>
<td>Dr. Russell Zaretzki</td>
<td><a href="mailto:rzaretzk@utk.edu">rzaretzk@utk.edu</a></td>
</tr>
</tbody>
</table>
Students in Departments Not Participating in the IGMCS Program

- A student in such a situation can still participate.
  - Student and advisor should submit to the Chair of the IGMCS Program Committee the courses to be taken.
  - Requirement is still the same:
    - Minor requires a combination of course work from three disciplines - Computer Science related, Mathematics/Stat, and a participating Science/Engineering domain (e.g., Chemical Engineering, Chemistry, Physics).

- Student’s department should be encouraged to participate in the IGMCS program.
  - Easy to do, needs approved set of courses and a liaison
Internship

- This is optional but strongly encouraged.
- Students in the program can fulfill 3 hrs. of their requirement through an Internship with researchers outside the student’s major.
- The internship may be taken offsite, e.g. ORNL, or on campus by working with a faculty member in another department.
- Internships must have the approval of the IGMCS Program Committee.
COMPUTATIONAL SCIENCE: INHERENTLY INTERDISCIPLINARY

PARTICIPATING DEPARTMENTS
- Electrical Engineering and Computer Science
- Information Science
- Mathematics
- Statistics

DOMAIN SCIENCES
- Chemical Engineering
- Chemistry
- Geography
- Life Sciences
- Physics
...
Why Turn to Simulation?

◆ **When the problem is too . . .**
  - Complex
  - Large / small
  - Expensive
  - Dangerous
◆ **to do any other way.**

![Diagram of airplane and bird flying]
Why Turn to Simulation?

- Climate / Weather Modeling
- Data intensive problems (data-mining, oil reservoir simulation)
- Problems with large length and time scales (cosmology)
Look at the Fastest Computers

- **Strategic importance of supercomputing**
  - Essential for scientific discovery
  - Critical for national security
  - Fundamental contributor to the economy and competitiveness through use in engineering and manufacturing

- **Supercomputers are the tool for solving the most challenging problems through simulations**
Units of Measure

• High Performance Computing (HPC) units are:
  - Flop: floating point operation, usually double precision unless noted
  - Flop/s: floating point operations per second
  - Bytes: size of data (a double precision floating point number is 8)

• Typical sizes are millions, billions, trillions…

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mega</td>
<td>Mflop/s = 10^6 flop/sec</td>
<td>Mbyte = 2^{20} = 1048576 ~ 10^6 bytes</td>
<td></td>
</tr>
<tr>
<td>Giga</td>
<td>Gflop/s = 10^9 flop/sec</td>
<td>Gbyte = 2^{30} ~ 10^9 bytes</td>
<td></td>
</tr>
<tr>
<td>Tera</td>
<td>Tflop/s = 10^{12} flop/sec</td>
<td>Tbyte = 2^{40} ~ 10^{12} bytes</td>
<td></td>
</tr>
<tr>
<td>Peta</td>
<td>Pflop/s = 10^{15} flop/sec</td>
<td>Pbyte = 2^{50} ~ 10^{15} bytes</td>
<td></td>
</tr>
<tr>
<td>Exa</td>
<td>Eflop/s = 10^{18} flop/sec</td>
<td>Ebyte = 2^{60} ~ 10^{18} bytes</td>
<td></td>
</tr>
<tr>
<td>Zetta</td>
<td>Zflop/s = 10^{21} flop/sec</td>
<td>Zbyte = 2^{70} ~ 10^{21} bytes</td>
<td></td>
</tr>
<tr>
<td>Yotta</td>
<td>Yflop/s = 10^{24} flop/sec</td>
<td>Ybyte = 2^{80} ~ 10^{24} bytes</td>
<td></td>
</tr>
</tbody>
</table>

• Current fastest (public) machine ~ 200 Pflop/s Peak
  - Up-to-date list at www.top500.org
High-Performance Computing Today

- In the past decade, the world has experienced one of the most exciting periods in computer development.
- Microprocessors have become smaller, denser, and more powerful.
- The result is that microprocessor-based supercomputing is rapidly becoming the technology of preference in attacking some of the most important problems of science and engineering.
Technology Trends: Microprocessor Capacity

2X transistors/Chip Every 1.5 years
Called “Moore’s Law”

Gordon Moore (co-founder of Intel) Electronics Magazine, 19

Moore’s Law – The number of transistors on integrated circuit chips (1971-2016)

Moore’s law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore’s law.

2X memory and processor speed and ½ size, cost, & power every 18 months.

Gordon Moore (co-founder of Intel) Electronics Magazine, 19

Number of devices/chip doubles every 18 months

The IBM Model 350 disk file with a storage space of 5MB from 1976 and a Micro SD Card is the author Max Roser.

The data visualization is available at Our World in Data.
Moore’s Secret Sauce: Dennard Scaling

Moore’s Law put lots more transistors on a chip…but it’s Dennard’s Law that made them useful

Dennard Scaling:
- Decrease feature size by a factor of λ and decrease voltage by a factor of λ; then
- # transistors increase by λ²
- Clock speed increases by λ
- Energy consumption does not change

2x transistor count
40% faster
50% more efficient

Dennard observed that voltage and current should be proportional to the linear dimensions of a transistor

Design of Ion-Implanted MOSFET’s with Very Small Physical Dimensions

Robert H. Dennard, Katsuo Iga, Kiyosega, Siemens AG, Munich, Germany

Abstract—This paper presents the design, fabrication, and characterization of ion-implanted MOSFET’s with channel lengths down to 3 μm. The transistors, for which the oxide thickness is 90 Å, are operated at a supply voltage of 5 V. The threshold voltage is 0.5 V. The minimum gate oxide thickness (when the channel length is 0.5 μm) is 20 Å.

[Denard, Gaensslen, Yu, Rideout, Bassous, Leblanc, IEEE JSSC, 1974]
Unfortunately Dennard Scaling is Over: What is the Catch?

Breakdown is the result of small feature sizes, current leakage poses greater challenges, and also causes the chip to heat up.

Powering the transistors without melting the chip
Dennard Scaling Over
Evolution of processors

The primary reason cited for the breakdown is that at small sizes, current leakage poses greater challenges, and also causes the chip to heat up, which creates a threat of thermal runaway and therefore further increases energy costs.
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
Transistors into Cores
Today’s Multicores
99% of Top500 Systems Are Based on Multicore

Sun Niagara2 (8 cores)
IBM Power 9 (24 cores)
Fujitsu Venus (16 cores)
Nvidia Volta (5120 Cuda cores)
AMD Ryzen (16 cores)
Intel Skylake (18 cores)
Intel Xeon Phi (60 cores)
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>&quot;New&quot; 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 \( \propto \text{Voltage}^2 \times \text{Frequency} \) \((V^2F)\)**

- **Frequency \( \propto \text{Voltage} \)**

- **Power \( \propto \text{Frequency}^3 \)**

<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><strong>Superscalar</strong></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>“New” Superscalar</strong></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><strong>Multicore</strong></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.
Look at the Fastest Computers

- **Strategic importance of supercomputing**
  - Essential for scientific discovery
  - Critical for national security
  - Fundamental contributor to the economy and competitiveness through use in engineering and manufacturing

- Supercomputers are the tool for solving the most challenging problems through simulations
Example of typical parallel machine

[Diagram of a typical parallel machine showing chip/socket, cores, memory controller, shared L3 cache, and PCIe.]
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
What do you mean by performance?

- **What is a xflop/s?**
  - xflop/s is a rate of execution, some number of floating point operations per second.
  - Whenever this term is used it will refer to 64 bit floating point operations and the operations will be either addition or multiplication.

- **What is the theoretical peak performance?**
  - The theoretical peak is based not on an actual performance from a benchmark run, but on a paper computation to determine the theoretical peak rate of execution of floating point operations for the machine.
  - The theoretical peak performance is determined by counting the number of floating-point additions and multiplications (in full precision) that can be completed during a period of time, usually the cycle time of the machine.
  - For example, an Intel Xeon 5570 quad core at 2.93 GHz can complete 4 floating point operations per cycle or a theoretical peak performance of 11.72 GFlop/s per core or 46.88 Gflop/s for the socket.
- Listing of the 500 most powerful Computers in the World
- Yardstick: Rmax from LINPACK MPP
  \[ Ax = b, \quad \text{dense problem} \]
- Updated twice a year
  SC‘xy in the States in November
  Meeting in Germany in June
- All data available from www.top500.org
PERFORMANCE DEVELOPMENT OF HPC OVER THE LAST 25 YEARS FROM THE TOP500

My Laptop: 166 Gflop/s

SUM

N=1

N=500

143 Pflop/s

1.40 Eflop/s

875 Tflop/s

6-8 years

1.17 Tflop/s

59.7 GFlop/s

0.1

1

10

100

1000

10000

100000

1000000

10000000

1E+09

1E+10

1994

1996

1998

2000

2002

2004

2006

2008

2010

2012

2014

2016

2018
PERFORMANCE DEVELOPMENT

- **SUM**
- \(N=1\)
- \(N=10\)
- \(N=100\)

Achieved:
- ASCI Red Sandia NL
- RoadRunner Los Alamos NL
- TaihuLight

Today:
- SW 26010 & Intel KNL ~ 3 Tflop/s
- 1 cabinet of TaihuLight ~ 3 Pflop/s

 Eflops \((10^{18})\) Achieved?
- China says 2020
- U.S. says 2021
## November 2018: The TOP 10 Systems

<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>GFlops/Watt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DOE / OS Oak Ridge Nat Lab</td>
<td>Summit, IBM Power 9 (22C, 3.0GHz), Nvidia GV100, Mellonox EDR</td>
<td>USA</td>
<td>2,397,824</td>
<td>144.</td>
<td>72</td>
<td>11.1</td>
<td>14.7</td>
</tr>
<tr>
<td>2</td>
<td>DOE / NNSA L Livermore Nat Lab</td>
<td>Sierra, IBM Power 9 (22C, 3.1GHz), Nvidia GV100, Mellonox EDR</td>
<td>USA</td>
<td>1,572,480</td>
<td>94.6</td>
<td>75</td>
<td>7.44</td>
<td>12.7</td>
</tr>
<tr>
<td>3</td>
<td>National Super Computer Center in Wuxi</td>
<td>Sunway TaihuLight, SW26010 (260C) + Custom</td>
<td>China</td>
<td>10,649,000</td>
<td>93.0</td>
<td>74</td>
<td>15.4</td>
<td>6.05</td>
</tr>
<tr>
<td>4</td>
<td>National Super Computer Center in Guangzhou</td>
<td>Tianhe-2A NUDT, Xeon (12C) + MATRIX-2000 + Custom</td>
<td>China</td>
<td>4,981,760</td>
<td>61.4</td>
<td>61</td>
<td>18.5</td>
<td>3.32</td>
</tr>
<tr>
<td>5</td>
<td>Swiss CSCS</td>
<td>Piz Daint, Cray XC50, Xeon (12C) + Nvidia P100(56C) + Custom</td>
<td>Switzerland</td>
<td>387,872</td>
<td>21.2</td>
<td>78</td>
<td>2.38</td>
<td>8.90</td>
</tr>
<tr>
<td>6</td>
<td>DOE / NNSA Los Alamos &amp; Sandia</td>
<td>Trinity, Cray XC40, Xeon Phi (68C) + Custom</td>
<td>USA</td>
<td>979,968</td>
<td>20.2</td>
<td>49</td>
<td>7.58</td>
<td>2.66</td>
</tr>
<tr>
<td>7</td>
<td>Nat Inst of Advanced Indus Sci &amp; Tech</td>
<td>AI Bridging Cloud Infrast (ABC) Fujitsu Xeon (20C, 22.4GHz) Nvidia V1000 IB-EDR</td>
<td>Japan</td>
<td>391,680</td>
<td>16.9</td>
<td>61</td>
<td>1.65</td>
<td>12.05</td>
</tr>
<tr>
<td>8</td>
<td>Leibniz Rechenzentrum</td>
<td>SuperMUC-NG, Lenovo, ThinkSystem SD530, Xeon Platinum 8174 24C 3.1GHz, Intel Omni-Path</td>
<td>Germany</td>
<td>311,040</td>
<td>19.5</td>
<td>63</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>DOE / OS Oak Ridge Nat Lab</td>
<td>Titan, Cray XK7, AMD (16C) + Nvidia Kepler GPU (14C) + Custom</td>
<td>USA</td>
<td>560,640</td>
<td>17.6</td>
<td>65</td>
<td>8.21</td>
<td>2.14</td>
</tr>
<tr>
<td>10</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>17.2</td>
<td>85</td>
<td>7.89</td>
<td>2.18</td>
</tr>
<tr>
<td>500</td>
<td>Internet Services</td>
<td>Inspur TS1000 Xeon</td>
<td>44,800</td>
<td>0.874</td>
<td>49</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
US has fallen to the lowest point since the TOP500 list was created. China has 45% of the systems. US has 22% of the systems.
State of Supercomputing in 2018

- Pflops (> $10^{15}$ Flop/s) computing fully established with 429 systems.
- Three technology architecture possibilities or “swim lanes” are thriving.
  - Commodity (e.g. Intel)
  - Commodity + accelerator (e.g. GPUs) (137 systems)
  - Special purpose lightweight cores (e.g. IBM BG, Knights Landing, TaihuLight, PEZY-SC2, ARM (1 system))
- Interest in supercomputing is now worldwide, and growing in many new markets (~50% of Top500 computers are in industry).
- Exascale ($10^{18}$ Flop/s) projects exist in many countries and regions.
- Intel processors largest share, 95% followed by AMD, .6%. 
PRODUCERS
COUNTRIES / SYSTEM SHARE

- United States: 22%
- China: 46%
- Japan: 6%
- United Kingdom: 4%
- Germany: 3%
- France: 4%
- Ireland: 2%
- Canada: 2%
- Others: 11%
VENDORS / SYSTEM SHARE

- **Lenovo**: 140 systems, 28% of 500
- **Inspur**: 86 systems, 17% of 500
- **Sugon**: 57 systems, 12% of 500
- **Cray Inc.**: 49 systems, 10% of 500
- **HPE**: 45 systems, 9% of 500
- **Bull**: 22 systems, 4% of 500
- **Fujitsu**: 15 systems, 3% of 500
- **Huawei**: 14 systems, 3% of 500
- **IBM**: 12 systems, 2% of 500
- **Dell EMC**: 12 systems, 2% of 500
- **Others**: 48 systems, 10% of 500

# of systems, % of 500
Comparison

1 Gflop/s  
70 Gflop/s  
200,000,000 Gflop/s
Systems in the US DOE under ECP

- **Oak Ridge Lab and Lawrence Livermore Lab received IBM and Nvidia based systems**
  - IBM Power 9 + Nvidia Volta V100
  - 5-10 times last generation Titan supercomputer on apps
  - 4,600 nodes, each containing six (ORNL) 7.5-teraflop NVIDIA V100 GPUs, and two IBM Power9 CPUs, its aggregate peak performance > 200 petaflops.

- **In 2021 Argonne Lab to receive Intel based system**
  - Exascale systems, based on Future Intel parts
  - Aurora 21
    - Balanced architecture to support three pillars
    - Large-scale Simulation (PDEs, traditional HPC)
    - Data Intensive Applications (science pipelines)
    - Deep Learning and Emerging Science AI
    - Integrated computing, acceleration, storage
    - Towards a common software stack
### System Performance
- Peak performance of 200 Pflop/s for modeling & simulation
- Peak performance of 3.3 Eflop/s for 16 bit floating point used in for data analytics and artificial intelligence

### Each node has
- 2 IBM POWER9 processors
  - Each w/22 cores
- 6 NVIDIA Tesla V100 GPUs
  - Each w/80 SMs
- 608 GB of fast memory
- 1.6 TB of NVMe memory

### The system includes
- 4608 nodes
- Dual-rail Mellanox EDR InfiniBand network
- 250 PB IBM Spectrum Scale file system transferring data at 2.5 TB/s
US Department of Energy Exascale Computing Program has formulated a holistic approach that uses co-design and integration to achieve capable exascale.

- **Application Development**: Science and mission applications
- **Software Technology**: Scalable and productive software stack
- **Hardware Technology**: Hardware technology elements
- **Exascale Systems**: Integrated exascale supercomputers

ECP’s work encompasses applications, system software, hardware technologies and architectures, and workforce development.
Toward Exascale

- **China plans for Exascale 2020**
  - Three separate developments in HPC; “Anything but from the US”
    - **Wuxi**
      - Upgrade the ShenWei O(100) Pflops
    - **National University for Defense Technology**
      - Tianhe-2A O(100) Pflops will be Chinese ARM processor + accelerator
    - **Sugon – CAS ICT**
      - X86 + accelerator based; collaboration with AMD
  - **US DOE - Exascale Computing Program - 7 Year Program (currently in year 2)**
    - Initial exascale system based on advanced architecture and delivered in 2021
    - Enable capable exascale systems, based on ECP R&D, delivered in 2022 and deployed in 2023

Exascale
- 50X the performance of today’s 20 PF systems
- 20-30 MW power
- ≤ 1 perceived fault /week
- SW to support broad range of apps
Secretary of Energy Rick Perry Announces $1.8 Billion Initiative for New Supercomputers

APRIL 9, 2018

Systems Will Solidify U.S. Leadership in the “Exascale” Computing Era

WASHINGTON, D.C. – U.S. Secretary of Energy Rick Perry today announced a Request for Proposals (RFP), potentially worth up to $1.8 billion, for the development of at least two new exascale supercomputers to be deployed at U.S. Department of Energy (DOE) National Laboratories in the 2021-2023 timeframe.

The new supercomputers funded through this RFP will be follow-on systems to the first U.S. exascale system authorized by Secretary Perry this past June, named Aurora, which is currently under development at Argonne National Laboratory (ANL) and scheduled to come online in 2021. The RFP announced today also envisions the possibility of upgrades or even a follow-on system to Aurora in 2022-2023, depending on an assessment of needs and opportunities at that time.

This RFP calls for a system to be deployed at Oak Ridge National Laboratory in Oak Ridge, Tennessee, and for another to be sited at Lawrence Livermore National Laboratory in Livermore, California.
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
- Telecomm
- Weather and Climate Research
- Weather Forecasting
Many fields are beginning to adopt machine learning to augment modeling and simulation methods

- Climate
- Biology
- Drug Design
- Epidemiology
- Materials
- Cosmology
- High-Energy Physics
Deep Learning Needs Batched Operations

Matrix Multiply is the time consuming part.

Convolution Layers and Fully Connected Layers require matrix multiply

There are many GEMM’s of small matrices, perfectly parallel, can get by with 16-bit floating point

Convolution Step
In this case 3x3 GEMM

Fully Connected Classification
Mixed Precision

• Today many precisions to deal with

<table>
<thead>
<tr>
<th>Type</th>
<th>Size</th>
<th>Range</th>
<th>$u = 2^{-t}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>half</td>
<td>16 bits</td>
<td>$10^{\pm5}$</td>
<td>$2^{-11} \approx 4.9 \times 10^{-4}$</td>
</tr>
<tr>
<td>single</td>
<td>32 bits</td>
<td>$10^{\pm38}$</td>
<td>$2^{-24} \approx 6.0 \times 10^{-8}$</td>
</tr>
<tr>
<td>double</td>
<td>64 bits</td>
<td>$10^{\pm308}$</td>
<td>$2^{-53} \approx 1.1 \times 10^{-16}$</td>
</tr>
<tr>
<td>quadruple</td>
<td>128 bits</td>
<td>$10^{\pm4932}$</td>
<td>$2^{-113} \approx 9.6 \times 10^{-35}$</td>
</tr>
</tbody>
</table>

• Note the number range with half precision (16 bit fl.pt.)
IEEE 754 Half Precision (16-bit) Floating Pt Standard

A lot of interest driven by “machine learning”
Power is an Industry Wide Problem

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

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

Google facilities

- leveraging hydroelectric power
- old aluminum plants

Google Plant in The Dalles, Oregon, from NYT, June 14, 2006

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 over slow links
- Next generation more integrated
- Intel’s Xeon Phi
  - 244 “threads” 61 cores
- AMD’s Fusion
  - Multicore with embedded graphics ATI
- Nvidia’s Volta with 5120 “Cuda cores”, 80 cores, 640 Tensor cores (4x4 32 bit matrix multiply)
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”
- .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

**Host Memory**

**Device Memory**
- 6 GB

**192 Cuda cores/SMX**
ACCELERATORS
Multi- to Many-Core

- **Complex cores**: huge, complex, lots of internal concurrency latency hiding
- **Simple cores**: small, simpler core little internal concurrency latency-sensitive

All Complex Cores  
*e.g. Intel Xeon*

Mixed Big & Small Cores

All Small Cores  
*e.g. Intel MIC*
Challenges of using Accelerators

- **High levels of parallelism**
  Many GPU/Accelerator 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 ]
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 intra-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

Approximate power costs (in picoJoules)

<table>
<thead>
<tr>
<th></th>
<th>2011</th>
</tr>
</thead>
<tbody>
<tr>
<td>DP FMADD flop</td>
<td>100 pJ</td>
</tr>
<tr>
<td>DP DRAM read</td>
<td>4800 pJ</td>
</tr>
<tr>
<td>Local Interconnect</td>
<td>7500 pJ</td>
</tr>
<tr>
<td>Cross System</td>
<td>9000 pJ</td>
</tr>
</tbody>
</table>

Source: John Shalf, LBNL

- Algorithms & Software: minimize data movement; perform more work per unit data movement.
nVIDIA's 28nm chips.
For a floating point operation costs 20pJ.
Getting the operands from local memory (situated 1mm away) consumes 26pJ.
If the operands need to be obtained from the other end of the die, it requires 1nJ
while if the operands need to be read from DRAM, the cost is 16nJ.
Energy Cost Challenge

- At ~$1M per MW energy costs are substantial
  - 10 Pflop/s in 2011 uses ~10 MWs
  - 1 Eflop/s in 2020 > 100 MWs

- DOE Target: 1 Eflop/s around 2020-2022 at 20 MWs
We Can Build an Exascale System Today?

Connect together 5 ORNL Summit systems

 Require 60 MW of power, programming for 100 M threads, and $2.0B price tag
Future Systems May Be Composed of Different Kinds of Cores

- DRAM chips (cells)
  - Latency
- Memory controller
- Address
- Data
- 3D DRAM (cells)
  - Memory controller
  - Address
  - Data

Lower latency
Higher bandwidth
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
Critical Issues at Peta & Exascale for Algorithm and Software Design

• **Synchronization-reducing algorithms**
  - Break Fork-Join model

• **Communication-reducing algorithms**
  - Use methods which have lower bound on communication

• **Mixed precision methods**
  - 2x speed of ops and 2x speed for data movement

• **Autotuning**
  - Today’s machines are too complicated, build “smarts” into software to adapt to the hardware

• **Fault resilient algorithms**
  - Implement algorithms that can recover from failures

• **Reproducibility of results**
  - Today we can’t guarantee this. We understand the issues, but some of our “colleagues” have a hard time with this.
Conclusions

• For the last decade or more, the research investment strategy has been overwhelmingly biased in favor of hardware.

• This strategy needs to be rebalanced - barriers to progress are increasingly on the software side.

• Moreover, the return on investment is more favorable to software.
  - Hardware has a half-life measured in years, while software has a half-life measured in decades.

• High Performance Ecosystem out of balance
  - Hardware, OS, Compilers, Software, Algorithms, Applications
    - No Moore’s Law for software, algorithms and applications
CS 594 – Scientific Computing for Engineers
Assignment #6
February 21st, 2018
Due: March 7th, 2018
Simple Operations

I would like you to implement a version of the following mathematical operations:

- the 2-norm of a vector,
  \[ \| x \|_2 = \sqrt{x^T x} = \sqrt{\sum x_i^2} \]

- matrix - vector multiplication,
  \[ y = y + A \cdot x \]
  \[ y_i = y_i + \sum_{j=1}^{n} A_{ij} \cdot x_j \text{ for } i = 1, \ldots, n \]

- matrix multiplication
  \[ C = C + A \cdot B \]
  \[ C_{ij} = C_{ij} + \sum_{k=1}^{n} A_{ik} \cdot B_{kj} \text{ for } i, j = 1, \ldots, n \]

The point of this assignment is not to write software, but to look at the performance for each of your implementations and try to explain why you are getting the performance you see and what you could do to increase the performance. You should produce a software implementation for each and run some experiments on various computers with different processors. I would like to see a report and analysis of your results; perhaps some plots of your performance data for n between say 10 and 1000. Please verify and convince me that you are computing the correct results in each case. Let me know what computers you used and how you are getting the performance results as well.

You can find out information on various processors at:


CS 594
Notes on Assignment #1

The programming that is part of this homework is not the important aspect of the assignment; it is the analysis of what your program is doing. In particular I’m interested in the following: first you should convince me that your program is doing the right thing, that is you should verify that you are computing the “correct” solution. You can do this by using data that when used with your program produces a known result or solution. (Don’t use zeros or ones in your data as that may give an incorrect timing behavior.) Or you can compare your results with a routine from a standard numerical library (that you assume is correct) and compute a “residual”, say something like:

\[ || \text{your solution} - \text{known solution} || \]

I would like you to analyze your timing results by graphing the rate of execution (ops/sec) as you vary the size of the problem.

I would also like an analysis of the rate you achieve to the theoretical peak performance rate of the processor. You should also describe why the performance you are achieving is so different than the peak.