Overview of HPC

Jack Dongarra

Computer Science Department
University of Tennessee

and

Mathematical Sciences Section
Oak Ridge National Laboratory

(http://www.netlib.org/utk/people/JackDongarra.html)

Three Basic Ways of Improving Computer Productivity

- Increasing component operating speed
- Parallelizing the process of calculation
- Making computers that are specialized for a certain class of problem

Why Parallel Computers?

- Desire to solve bigger, more realistic applications problems.
- Fundamental limits are being approached.
- More cost effective solution

Example: Weather Prediction (Navier-Stokes)
with 3D Grid around the Earth

6 variables

\[
\begin{array}{c}
temperature \\
pressure \\
humidity \\
wind velocity
\end{array}
\]

- 1 Kilometer Cells
- 10 slices $\rightarrow 5 \times 10^9$ cells
- each cell is 8 bytes, $2 \times 10^{10}$ Bytes $= 200$ GBytes
- at each cell will perform 100 ops/cell
- 1 minute time step

\[
\frac{100 \text{ ops/cell} \times 5 \times 10^9 \text{ cells}}{\text{time/minute/day}} = 8 \text{ GFlop/s}
\]

Goals of a Parallel Architecture

- High Performance:
  - Performance $>$ fastest uniprocessors
- Competitive cost-performance:
  - Cost-performance is competitive with workstations
- Scalable performance and cost-performance
- General-purpose: cost-effective for a wide range of applications
Scalable Multiprocessors

What is Required?

- Must scale the local memory bandwidth linearly.
- Must scale the global interprocessor communication bandwidth.
- Scaling memory bandwidth cost-effectively requires separate, distributed memories.
- Cost-effectiveness also requires best price-performance in individual processors.

What we get

- Compelling Price/Performance
- Tremendous scalability
- Tolerable entry price
- Tackle intractable problems

What limits the performance of a parallel program?

- Available Parallelism
- Load Balance
  - some processors work while others wait
- Extra work
  - management of parallelism
  - redundant computation
- Communication

Scalability

- A parallel algorithm is scalable if the concurrent efficiency can be held constant as the number of processors increases by increasing the problem size.
- The iso-efficiency function, $\rho_s(N_s)$, gives the problem size necessary to maintain some fixed efficiency, $\epsilon$, as $N_s$ increases.
Shared Memory Architectures

- **Key Feature**: All processors in the system can directly access memory locations in the system, thus providing a convenient and fast mechanism for processors to communicate.
  - Convenient: (i) location transparency; (ii) abstraction supported is same as that on current day uniprocessors.
- Memory can be centrally placed or distributed in such machines.

![Interconnection Diagram]

Message Passing Architectures

- Processors can directly access only local memory and all communication and synchronization happens via messages.

![Interconnection Diagram]

Comparison

- **Shared Memory**
  - Explicit global data structure
  - Decomposition of work is independent of data layout
  - Communication is implicit
  - Explicit synchronization
    - need to avoid race condition and over writing

- **Message Passing**
  - Implicit global data structure
  - Decomposition of data determines assignment of work
  - Communication is explicit
  - Synchronization is implicit
    - data buffered till received

Survey of Current Machines

- **MIMD, global memory**
  - Cray Y-MP C90, T32
  - NEC SX-3, SX-4, Hitachi S820, Fujitsu VP2000

- **MIMD, distributed memory**
  - Cray T3D, T3E
  - Fujitsu VPP500
  - IBM SP-2
  - Intel Paragon
  - Mekko CS-2
  - nCUBE 2s, nCUBE3s
  - Parsytec GigaCube GC-2
  - Thinking Machines CM-5

- **SIMD**
  - Thinking Machines CM-2
  - Maspar MP-2
MIMD, global: Comparison

<table>
<thead>
<tr>
<th>Computer</th>
<th>cycle time</th>
<th>single CPU peak Mflop/s</th>
<th># CPUs peak Mflop/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cray-1</td>
<td>12.5 ns</td>
<td>160</td>
<td>1</td>
</tr>
<tr>
<td>Cray X-MP</td>
<td>8.3 ns</td>
<td>233</td>
<td>4</td>
</tr>
<tr>
<td>Cray 2</td>
<td>4.1 ns</td>
<td>488</td>
<td>4</td>
</tr>
<tr>
<td>Cray Y-MP</td>
<td>6.0 ns</td>
<td>333</td>
<td>8</td>
</tr>
<tr>
<td>Cray C90</td>
<td>4.2 ns</td>
<td>952</td>
<td>16</td>
</tr>
<tr>
<td>Cray T90</td>
<td>2.2 ns</td>
<td>1,800</td>
<td>32</td>
</tr>
<tr>
<td>NEC SX-4i</td>
<td>2.9 ns</td>
<td>5,500</td>
<td>4</td>
</tr>
<tr>
<td>NEC SX-4</td>
<td>4 ns</td>
<td>2,000</td>
<td>32</td>
</tr>
<tr>
<td>Fujitsu VP2600</td>
<td>4.0 ns</td>
<td>4,000</td>
<td>1</td>
</tr>
<tr>
<td>Hitachi S880i</td>
<td>8.00 ns</td>
<td>8,000</td>
<td>4</td>
</tr>
</tbody>
</table>

Growth in Microprocessor Performance in 1980's

Performance Numbers Comparing Various RISC Processors

Using the Linpack Benchmark

(All based on actual runs.)

Linpack Mflop/s based on Fortran code only.

Ax = b Mflop/s is based on a blocked algorithm (LAPACK routine) using the Level 3 BLAS as provided by the vendor.

<table>
<thead>
<tr>
<th>Machine</th>
<th>Mflop/s cycle time</th>
<th>Linpack n=100</th>
<th>% peak</th>
<th>Ax = b n=1000</th>
<th>% peak</th>
<th>Thess Peak</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEC Alpha</td>
<td>300 3.2</td>
<td>100 23%</td>
<td>411 69%</td>
<td>600</td>
<td>600</td>
<td></td>
</tr>
<tr>
<td>HRS Power</td>
<td>66 15</td>
<td>110 18%</td>
<td>250 89%</td>
<td>356</td>
<td>356</td>
<td></td>
</tr>
<tr>
<td>NEC Power</td>
<td>75 13.3</td>
<td>301 113%</td>
<td>268 87%</td>
<td>380</td>
<td>380</td>
<td></td>
</tr>
<tr>
<td>BP 735</td>
<td>45 10.5</td>
<td>61 38%</td>
<td>307 5%</td>
<td>395</td>
<td>395</td>
<td></td>
</tr>
<tr>
<td>DEC Alpha</td>
<td>290 5</td>
<td>43 22%</td>
<td>355 78%</td>
<td>280</td>
<td>280</td>
<td></td>
</tr>
<tr>
<td>DEC Alpha</td>
<td>192 5</td>
<td>39 2%</td>
<td>161 77%</td>
<td>182</td>
<td>182</td>
<td></td>
</tr>
<tr>
<td>DEC Alpha</td>
<td>100 6</td>
<td>36 2%</td>
<td>114 7%</td>
<td>90</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>DEC Alpha</td>
<td>150 7</td>
<td>30 2%</td>
<td>107 7%</td>
<td>56</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>DEC Alpha</td>
<td>42 2.4</td>
<td>26 1%</td>
<td>76 88%</td>
<td>84</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>HP 710/750</td>
<td>66 15</td>
<td>24 16%</td>
<td>47 7%</td>
<td>66</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>NEC i486/486</td>
<td>140 18</td>
<td>15 13%</td>
<td>32 6%</td>
<td>50</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>DEC Alpha</td>
<td>25 10</td>
<td>15 10%</td>
<td>42 8%</td>
<td>51</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>NEC SX4i</td>
<td>40 25</td>
<td>15 18%</td>
<td>31 7%</td>
<td>40</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Intel 2860</td>
<td>40 25</td>
<td>10 23%</td>
<td>34 8%</td>
<td>40</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>CRAY T400</td>
<td>474 2.2</td>
<td>523 29%</td>
<td>156 8%</td>
<td>1800</td>
<td>1800</td>
<td></td>
</tr>
<tr>
<td>CRAY C90</td>
<td>236 4.2</td>
<td>36 1%</td>
<td>982 97%</td>
<td>952</td>
<td>952</td>
<td></td>
</tr>
<tr>
<td>CRAY 300</td>
<td>100 18</td>
<td>115 7%</td>
<td>581 96%</td>
<td>280</td>
<td>280</td>
<td></td>
</tr>
<tr>
<td>CRAY Y-MP</td>
<td>166 6</td>
<td>30 18%</td>
<td>321 97%</td>
<td>313</td>
<td>313</td>
<td></td>
</tr>
<tr>
<td>CRAY M-MP</td>
<td>128 8.2</td>
<td>221 7%</td>
<td>226 9%</td>
<td>235</td>
<td>235</td>
<td></td>
</tr>
<tr>
<td>CRAY 1</td>
<td>481 2.1</td>
<td>241 27%</td>
<td>962</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRAY 2</td>
<td>264 4.3</td>
<td>230 22%</td>
<td>184 7%</td>
<td>988</td>
<td>988</td>
<td></td>
</tr>
<tr>
<td>CRAY 1</td>
<td>80 22.2</td>
<td>27 17%</td>
<td>108 6%</td>
<td>340</td>
<td>340</td>
<td></td>
</tr>
</tbody>
</table>
**Fundamental Architectural Issues**

- **How is communicated data and/or partner referenced?**
- **How can producers and consumers of data coordinate their activities?**
- **How long does it take to communicate data in a protected fashion?**
- **How much data can be communicated per second? How many operations per second?**

---

**Number of systems installed**

- **Total: 500**

---

**TOP500 - CPU Technology**

- **# Systems**
  - 0 to 500
  - 501 to 1000
  - 1001 to 1500
  - 1501 to 2000
  - 2001 to 2500
  - 2501 to 3000
  - 3001 to 3500
  - 3501 to 4000

- **Processor**
  - CMOS off the shelf
  - ECL
  - CMOS proprietary

---

**Cray**

- Clock: 28MHz
- Processor: Sparc II

---

**Intel**

- Clock: 12MHz
- Processor: Pentium Pro

---

**SGI**

- Clock: 50MHz
- Processor: MIPS R2000

---

**Hitachi**

- Clock: 15MHz
- Processor: PowerPC 601

---

**HP PA-RISC**

- Clock: 30MHz
- Processor: PA-7000

---

**Fujitsu**

- Clock: 40MHz
- Processor: SuperSPARC II

---

**IBM**

- Clock: 25MHz
- Processor: PowerPC 603

---

**Intel**

- Clock: 15MHz
- Processor: Pentium II

---

**TMC**

- Clock: 15MHz
- Processor: MIPS R10000

---

**Convex**

- Clock: 15MHz
- Processor: PowerPC 601

---

**Cray**

- Clock: 15MHz
- Processor: PowerPC 601

---

**Fujitsu**

- Clock: 15MHz
- Processor: PowerPC 601

---

**Hitachi**

- Clock: 15MHz
- Processor: PowerPC 601

---

**IBM**

- Clock: 15MHz
- Processor: PowerPC 601

---

**Intel**

- Clock: 15MHz
- Processor: PowerPC 601

---

**TMC**

- Clock: 15MHz
- Processor: PowerPC 601
Table 1: Multiprocessor Latency and Bandwidth.

<table>
<thead>
<tr>
<th>Machine</th>
<th>OS</th>
<th>Latency (us)</th>
<th>Bandwidth (MB/s)</th>
<th>n/2</th>
<th>Theoretical Bandwidth (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convex SPP (PVM)</td>
<td>PVM-5(25,128)</td>
<td>55</td>
<td>71</td>
<td>498</td>
<td>298</td>
</tr>
<tr>
<td>Convex SPP (m)</td>
<td>SP2(3,90)</td>
<td>11</td>
<td>71</td>
<td>498</td>
<td>298</td>
</tr>
<tr>
<td>Cray T3D (m)</td>
<td>MAX-120.2</td>
<td>3</td>
<td>120</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>Cray T3D (PVM)</td>
<td>MAX-120.2</td>
<td>31</td>
<td>120</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>Intel Paragon</td>
<td>OSF-96.2</td>
<td>25</td>
<td>172</td>
<td>275</td>
<td>275</td>
</tr>
<tr>
<td>Intel Delta</td>
<td>NX-3,23,30</td>
<td>77</td>
<td>9</td>
<td>900</td>
<td>900</td>
</tr>
<tr>
<td>Intel IPC/860</td>
<td>NX-3,23,30</td>
<td>65</td>
<td>3</td>
<td>340</td>
<td>340</td>
</tr>
<tr>
<td>Intel IPC/2</td>
<td>NX-3,23,30</td>
<td>730</td>
<td>2.5</td>
<td>1742</td>
<td>1742</td>
</tr>
<tr>
<td>IBM S/390</td>
<td>MPE</td>
<td>7</td>
<td>7</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>IBM S/2</td>
<td>MIP</td>
<td>35</td>
<td>3</td>
<td>3303</td>
<td>3303</td>
</tr>
<tr>
<td>KNLx2</td>
<td>OSF-13.2</td>
<td>73</td>
<td>8</td>
<td>923</td>
<td>923</td>
</tr>
<tr>
<td>Meiko CS2 (m)</td>
<td>Solanas-2</td>
<td>11</td>
<td>40</td>
<td>208</td>
<td>208</td>
</tr>
<tr>
<td>Meiko CS2</td>
<td>Solanas-2</td>
<td>83</td>
<td>46</td>
<td>3109</td>
<td>3109</td>
</tr>
<tr>
<td>netCUBE 2</td>
<td>Vortex-2</td>
<td>174</td>
<td>1.2</td>
<td>188</td>
<td>188</td>
</tr>
<tr>
<td>netCUBE 1</td>
<td>Vortex-2</td>
<td>354</td>
<td>0.4</td>
<td>189</td>
<td>189</td>
</tr>
<tr>
<td>NEC Conext</td>
<td>Enc. Rel.</td>
<td>34</td>
<td>25</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td>NEC Centerio</td>
<td>Enc. Rel.</td>
<td>34</td>
<td>25</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td>SGI</td>
<td>IRX-6</td>
<td>30</td>
<td>65</td>
<td>799</td>
<td>799</td>
</tr>
<tr>
<td>TMC CS3x5</td>
<td>CUBEM-20</td>
<td>95</td>
<td>0</td>
<td>962</td>
<td>962</td>
</tr>
<tr>
<td>Internet</td>
<td>TCP/IP</td>
<td>500</td>
<td>0.0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>FIBRE</td>
<td>TCP/IP</td>
<td>1000</td>
<td>9.7</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>ATM-10</td>
<td>TCP/IP</td>
<td>1000</td>
<td>3.5</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 2: Computation Performance.

<table>
<thead>
<tr>
<th>Machine</th>
<th>OS</th>
<th>Clock cycle</th>
<th>Throughput (Gflop/s)</th>
<th>Latency (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convex SPP (PVM)</td>
<td>PVM-5(25,128)</td>
<td>43</td>
<td>55</td>
<td>298</td>
</tr>
<tr>
<td>Convex SPP (m)</td>
<td>SP2(3,90)</td>
<td>31</td>
<td>71</td>
<td>498</td>
</tr>
<tr>
<td>Cray T3D (m)</td>
<td>MAX-120.2</td>
<td>3</td>
<td>120</td>
<td>300</td>
</tr>
<tr>
<td>Cray T3D (PVM)</td>
<td>MAX-120.2</td>
<td>31</td>
<td>120</td>
<td>300</td>
</tr>
<tr>
<td>Intel Paragon</td>
<td>OSF-96.2</td>
<td>25</td>
<td>172</td>
<td>275</td>
</tr>
<tr>
<td>Intel Delta</td>
<td>NX-3,23,30</td>
<td>77</td>
<td>9</td>
<td>900</td>
</tr>
<tr>
<td>Intel IPC/860</td>
<td>NX-3,23,30</td>
<td>65</td>
<td>3</td>
<td>340</td>
</tr>
<tr>
<td>Intel IPC/2</td>
<td>NX-3,23,30</td>
<td>730</td>
<td>2.5</td>
<td>1742</td>
</tr>
<tr>
<td>IBM S/390</td>
<td>MPE</td>
<td>7</td>
<td>7</td>
<td>1000</td>
</tr>
<tr>
<td>IBM S/2</td>
<td>MIP</td>
<td>35</td>
<td>3</td>
<td>3303</td>
</tr>
<tr>
<td>KNLx2</td>
<td>OSF-13.2</td>
<td>73</td>
<td>8</td>
<td>923</td>
</tr>
<tr>
<td>Meiko CS2 (m)</td>
<td>Solanas-2</td>
<td>11</td>
<td>40</td>
<td>208</td>
</tr>
<tr>
<td>Meiko CS2</td>
<td>Solanas-2</td>
<td>83</td>
<td>46</td>
<td>3109</td>
</tr>
<tr>
<td>netCUBE 2</td>
<td>Vortex-2</td>
<td>174</td>
<td>1.2</td>
<td>188</td>
</tr>
<tr>
<td>netCUBE 1</td>
<td>Vortex-2</td>
<td>354</td>
<td>0.4</td>
<td>189</td>
</tr>
<tr>
<td>NEC Conext</td>
<td>Enc. Rel.</td>
<td>34</td>
<td>25</td>
<td>400</td>
</tr>
<tr>
<td>NEC Centerio</td>
<td>Enc. Rel.</td>
<td>34</td>
<td>25</td>
<td>400</td>
</tr>
<tr>
<td>SGI</td>
<td>IRX-6</td>
<td>30</td>
<td>65</td>
<td>799</td>
</tr>
<tr>
<td>TMC CS3x5</td>
<td>CUBEM-20</td>
<td>95</td>
<td>0</td>
<td>962</td>
</tr>
<tr>
<td>Internet</td>
<td>TCP/IP</td>
<td>500</td>
<td>0.0</td>
<td>12</td>
</tr>
<tr>
<td>FIBRE</td>
<td>TCP/IP</td>
<td>1000</td>
<td>9.7</td>
<td>12</td>
</tr>
<tr>
<td>ATM-10</td>
<td>TCP/IP</td>
<td>1000</td>
<td>3.5</td>
<td>12</td>
</tr>
</tbody>
</table>

Message-passing Space

MIMD, multicompiler: networked workstations

Enabling software technology: PVM (Parallel Virtual Machine) available from netlib.org

Enabling software technology: MPI (Message Passing Interface) available from netlib.org

very active research area; about 150 software products; catalog available NHSE

Enabling hardware technology: high bandwidth interconnect is not here yet;

Ethernet: msec latencies and 100’s of Kbyte/sec bandwidth insufficient

Other technology is on the verge of becoming available: HIPPI products, Fibre Channel, ATM.
Shared-Memory Implementation

- Shared memory algorithm:
  - Divide cost-array into regions
  - Logically assign regions to PEs
  - Assign wires to PEs based on the region in which center lies
  - Do load balancing using stealing when local queue empty

- Good points:
  - Good load balancing
  - Mostly local accesses
  - High cache-hit ratio

- Not so good:
  - Non-deterministic
  - Potential hot spots
  - Amount of parallelism

Message-Passing Implementations

- Solution-1:
  - Distribute wires and cost-array regions as in sh-mem implementation
  - Handle when wire-path crosses to remote region
    - Send computation to remote PE, or
    - Send messages to access remote data

- Solution-2:
  - Wires distributed as in sh-mem implementation
  - Each PE has copy of full cost array
    - One owned region, plus potentially stale copy of others
    - Send frequent updates so that copies not too stale

- Consequences:
  - Waste of memory in replication
  - Stale data => poorer quality results or more iterations

In either case, lots of thinking needed on programmers part

Scalability

- Latency, Bandwidth, and Cost
- Construct shared memory out of simple message transactions across a general-purpose network.
  - Read-request, read-response
- Caching?
LINPACK BENCHMARK

- Linpack Benchmark is really three benchmarks.
  - Run the Benchmark code, all Fortran, no changes allowed to the software. User only to supply timing function. Benchmark runs the LINPACK routines DGESV and DGESL which calls the BLAS (DAXPY). Time to factor and solve a system of equations of order 100.
  - Run the Benchmark code, solve a system of equations of order 1000. User can supply the software to solve the problem.
  - Run the largest system of equations problem on the machine. User can supply the software to solve the problem.

\[ R_{\text{max}} \text{ Gflop/s for largest problem} \]
\[ N_{\text{max}} \text{ size of largest problem} \]
\[ N_{\text{size}} \text{ size where half the } R_{\text{max}} \text{ achieved} \]
\[ R_{\text{peak}} \text{ theoretical peak performance} \]

Must get the "correct results".

The results were checked for accuracy by calculating a residual for the problem \[ \|Ax - b\|/\|b\| \].

The execution rate is computed using \[ 2/3n^3 + 2n^2 \text{ operations.} \]