# 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**

$$6 \quad variables \begin{cases} temperature \\ pressure \\ humidity \\ 3 - wind \quad velocity \end{cases}$$

- **1 Kilometer Cells**
- **10 slices** $\rightarrow 5 \times 10^9$ **cells**
- **each cell is 8 bytes,** $2 \times 10^{11}$ **Bytes** $= 200$ **GBytes**
- **at each cell will perform 100 ops/cell**
- **1 minute time step**
- $\frac{100 ops/cell \times 5 \times 10^9 cells}{1 min \times 60 sec/min} = 8 GFlop/s$



## 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?**

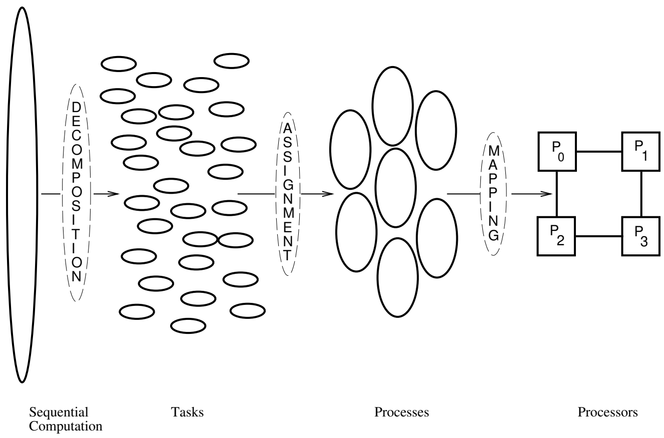
- **Must scale the local memory bandwidth linearly.**
- **Must scale the global interprocessor communication bandwidth.**
- **Scaling memory bandwidth cost-effectively requires separate, distributed memories.**
- **Cost-effectiveness also requires best price-performance in individual processors.**

**What we get**

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



**Creating a parallel program**




- Tasks : pieces of work that constitute the overall computation
- Processes (threads) : abstract entity that performs tasks
  - portion of address space private
  - portion shared
- 4 primary parts of the programming process
  - decomposition, assignment (partitioning), orchestration, and mapping



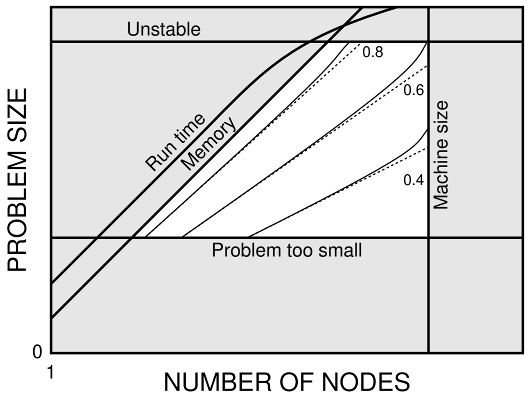
# What limits the performance of a parallel program?

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



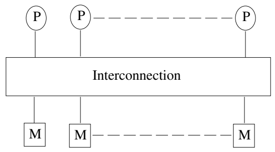
## Scalability

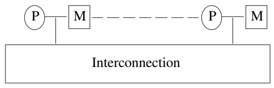
- A parallel algorithm is scalable if the concurrent efficiency can be held constant as the number of processors increases by increasing the problem size.
- The *isoefficiency function*, $\rho_\epsilon(N_p)$, **gives the problem size necessary to maintain some fixed efficiency,** $\epsilon$**, as** $N_p$ **increases.**



### Shared Memory Architectures

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



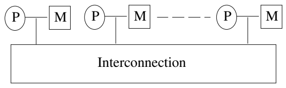

PVP - Parallel Vector Processors
SMP - Symmetic Multi-Processors

| | |
|---|---|
| Cray X,Y,C,T | Convex Exempler |
| NEC SX | (KSR) |
| Fujitsu CP 2600 | |
| Hitachi S-3800 | |
| SGI Power Chall | |
| SUN | |
| Sequent | |
| DEC Tubro Laser | |
| Chen | |



### Message Passing Architectures

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




MPP - Massively Parallel Processors

IBM SP-2
Intel Paragon
Cray T3D/E
NCube 3
(Thinking Machines CM-5)



# 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 C916, T32
- NEC SX–3, SX–4, Hitachi S820, Fujitsu VP2600

MIMD, distributed memory

- Cray T3D, T3E
- Fujitsu VPP500
- IBM SP–2
- Intel Paragon
- Meiko CS–2
- nCUBE 2s, nCUBE3s
- Parsytec GigaCube GC–2
- Thinking Machines CM–5

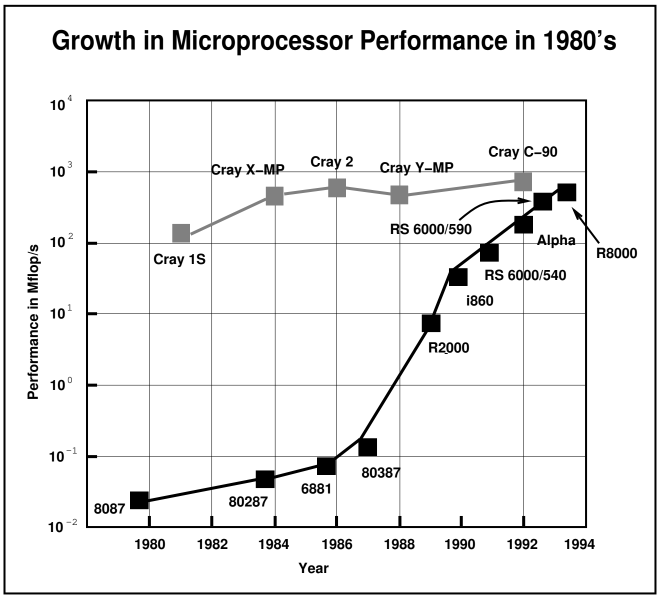
SIMD

- Thinking Machines CM–2
- Maspar MP–2

# MIMD, global: Comparison

| Computer | cycle time | single CPU peak Mflops/s | #CPUs | peak Mflops/s |
|---|---|---|---|---|
| Cray–1 | 12.5 ns | 160 | 1 | 160 |
| Cray X–MP | 8.5 ns | 233 | 4 | 932 |
| Cray 2 | 4.1 ns | 488 | 4 | 1,952 |
| Cray Y–MP | 6.0 ns | 333 | 8 | 2,666 |
| Cray C90 | 4.2 ns | 952 | 16 | 15,238 |
| Cray T90 | 2.2 ns | 1,800 | 32 | 57,600 |
| NEC SX–3 | 2.9 ns | 5,500 | 4 | 22,000 |
| NEC SX–4 | 4 ns | 2,000 | 32 | 64,000 |
| Fujitsu VP2600 | 4.0 ns | 4,000 | 1 | 4,000 |
| Hitachi S–3800/480 | | 8,000 | 4 | 32,000 |








Performance Numbers Comparing Various RISC Processors

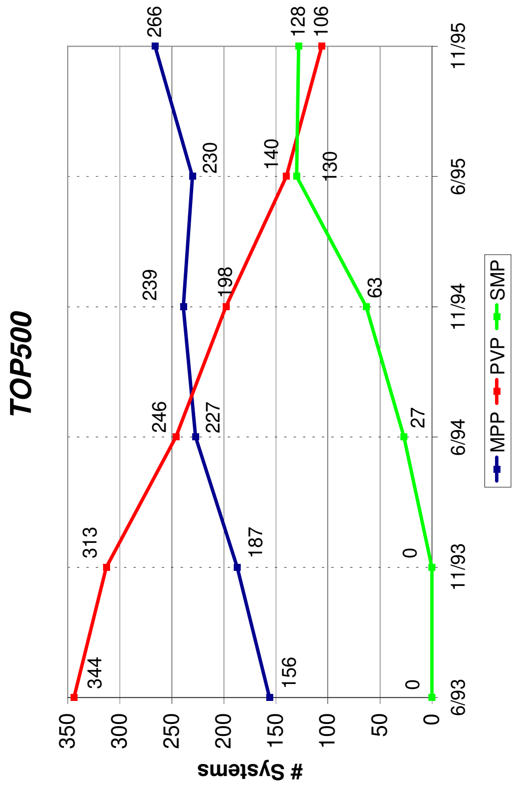
Using the Linpack Benchmark

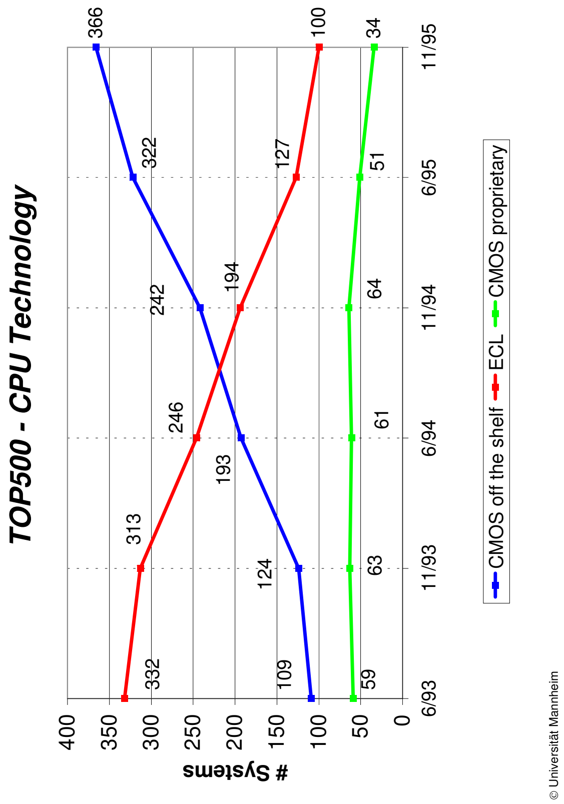
(All based on actual runs.)
Linpack n=100 based on Fortran code only.

Ax=b n=1000 is based on a blocked algorithm (LAPACK routine) using the Level 3 BLAS as provided by the vendor.

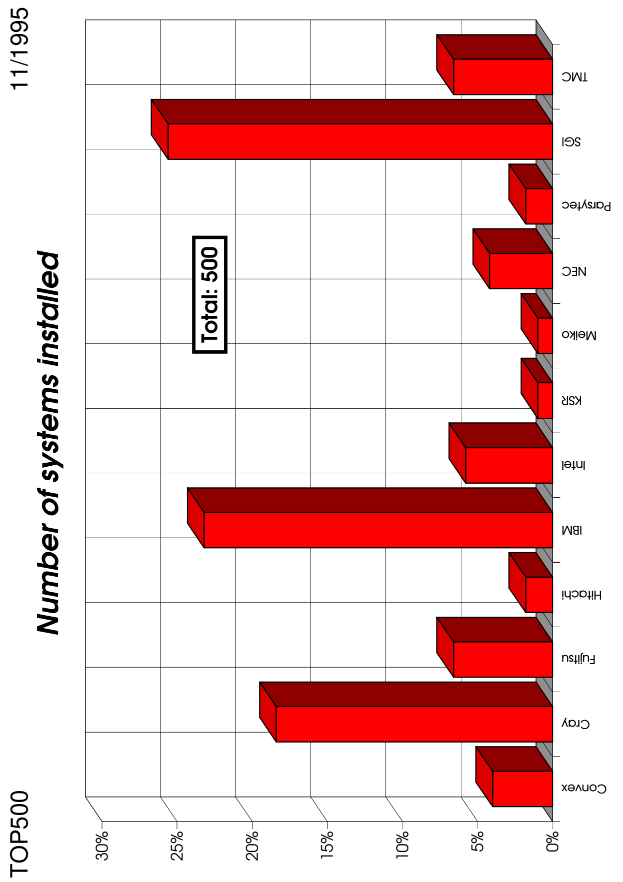
| Machine | MHz | Cycle time nsec | Linpack n=100 Mflops | % peak | Ax = b n=1000 Mflops | % peak | Theor Peak Mflops |
|---|---|---|---|---|---|---|---|
| DEC Alpha | 300 | 3.3 | 140 | 23% | 411 | 69% | 600 |
| IBM Power2 | 66 | 15 | 130 | 48% | 236 | 89% | 266 |
| SGI Power | 75 | 13.3 | 101 | 33% | 260 | 87% | 300 |
| HP 735 | 100 | 10 | 61 | 31% | 107 | 54% | 198 |
| DEC Alpha | 200 | 5 | 43 | 22% | 155 | 78% | 200 |
| DEC Alpha | 182 | 5 | 39 | 21% | 141 | 77% | 182 |
| IBM RS-580 | 66 | 15 | 38 | 30% | 104 | 83% | 125 |
| DEC Alpha | 160 | 6 | 36 | 23% | 114 | 71% | 160 |
| DEC Alpha | 150 | 7 | 30 | 20% | 107 | 71% | 150 |
| IBM RS-550 | 42 | 24 | 26 | 31% | 70 | 83% | 84 |
| HP 730/750 | 66 | 15 | 24 | 36% | 47 | 71% | 66 |
| SGI Indy2/Crim | 100 | 10 | 15 | 30% | 32 | 64% | 50 |
| IBM RS-530 | 25 | 40 | 15 | 30% | 42 | 84% | 50 |
| KSR (1 proc) | 40 | 25 | 15 | 38% | 31 | 78% | 40 |
| Intel i860 | 40 | 25 | 10 | 25% | 34 | 85% | 40 |
| CRAY T90 | 454 | 2.2 | 522 | 29% | 1576 | 88% | 1800 |
| CRAY C90 | 238 | 4.2 | 387 | 41% | 902 | 95% | 952 |
| CRAY J90 | 100 | 10 | 115 | 51% | 193 | 96% | 200 |
| CRAY Y-MP | 166 | 6 | 161 | 48% | 324 | 97% | 333 |
| CRAY X-MP | 118 | 8.5 | 121 | 51% | 218 | 93% | 235 |
| CRAY 3 | 481 | 2.1 | 241 | 25% | | | 962 |
| CRAY 2 | 244 | 4.1 | 120 | 25% | 384 | 79% | 488 |
| CRAY 1 | 80 | 12.5 | 27 | 17% | 110 | 69% | 160 |












## Divide

- **Important on Tridiagonal Solver for ADI Schemes.**
- **$9n$ floating point operations in the inner loop, $n$ are floating-point divide instructions.**
- **Also important on block tridiagonal and pentadiagonal solvers.**
- **Newton's iteration usually used**
- **Not typically pieplined**

| Processor | Clock (Mhz) | Divide (CP) | Cache BW (MW/s) | Memory BW (MW/s) | Peak Perf. (MF/s) | Linpack (MF/s) |
|---|---|---|---|---|---|---|
| Cray C-90/1 | 240 | 4 | - | 1440 | 960 | 387 |
| Cray Y-MP/1 | 166 | 4 | - | 500 | 333 | 161 |
| RS/6000-590 | 66 | 19 | 266 | 266 | 264 | 130 |
| DEC Alpha | 150 | 63 | 150 | 37 | 150 | 30 |
| HP PA-RISC | 99 | 20 | 99 | 33 | 198 | 41 |
| Intel i860 | 40 | 190 | 80 | 16 | 60 | 10 |
| SGI (R4400) | 100 | 36 | 100 | 50 | 50 | 17 |
| Super SPARC | 40 | 7 | 40 | 40 | 40 | 7 |



## Fundamental Architectural Issues

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

Table 1: Multiprocessor Latency and Bandwidth.

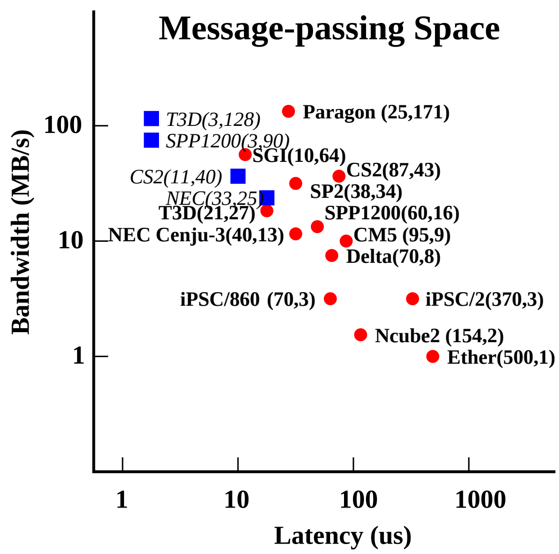
| Machine | OS | Latency $n = 0$ ($\mu$s) | Bandwidth $n = 10^6$ (MB/s) | $n_{1/2}$ bytes | Theoretical Bandwidth (MB/s) |
|---|---|---|---|---|---|
| Convex SPP1200 (PVM) | SPP-UX 3.0.4.1 | 63 | 15 | 1000 | 250 |
| Convex SPP1200 (sm m-n) | SPP-UX 3.0.4.1 | 11 | 71 | 1000 | 250 |
| Cray T3D (sm) | MAX 1.2.0.2 | 3 | 128 | 363 | 300 |
| Cray T3D (PVM) | MAX 1.2.0.2 | 21 | 27 | 1502 | 300 |
| Intel Paragon | OSF 1.0.4 | 29 | 154 | 7236 | 175 |
| Intel Paragon | SUNMOS 1.6.2 | 25 | 171 | 5856 | 175 |
| Intel Delta | NX 3.3.10 | 77 | 8 | 900 | 22 |
| Intel iPSC/860 | NX 3.3.2 | 65 | 3 | 340 | 3 |
| Intel iPSC/2 | NX 3.3.2 | 370 | 2.8 | 1742 | 3 |
| IBM SP-1 | MPL | 270 | 7 | 1904 | 40 |
| IBM SP-2 | MPI | 35 | 35 | 3263 | 40 |
| KSR-1 | OSF R1.2.2 | 73 | 8 | 635 | 32 |
| Meiko CS2 (sm) | Solaris 2.3 | 11 | 40 | 285 | 50 |
| Meiko CS2 | Solaris 2.3 | 83 | 43 | 3559 | 50 |
| nCUBE 2 | Vertex 2.0 | 154 | 1.7 | 333 | 2.5 |
| nCUBE 1 | Vertex 2.3 | 384 | 0.4 | 148 | 1 |
| NEC Cenju-3 | Env. Rel 1.5d | 40 | 13 | 900 | 40 |
| NEC Cenju-3 (sm) | Env. Rel 1.5d | 34 | 25 | 400 | 40 |
| SGI | IRIX 6.1 | 10 | 64 | 799 | 1200 |
| TMC CM-5 | CMMD 2.0 | 95 | 9 | 962 | 10 |
| Ethernet | TCP/IP | 500 | 0.9 | | 1.2 |
| FDDI | TCP/IP | 900 | 9.7 | | 12 |
| ATM-100 | TCP/IP | 900 | 3.5 | | 12 |



Table 2: Computation Performance.

| Machine | OS | Clock cycle MHz | (nsec) | Linpack 100 Mfls | (ops/cl) | Linpack 1000 Mfls | (ops/cl) | Latency us | (cl) |
|---|---|---|---|---|---|---|---|---|---|
| Convex SPP1200 (PVM) | SPP-UX 3.0.4.1 | 100 | (8.33) | 65 | (.54) | 123 | (1.02) | 63 | (7560) |
| Convex SPP1200 (sm m-n) | | | | | | | | 11 | (1260) |
| Cray T3D (sm) | MAX 1.2.0.2 | 150 | (6.67) | 38 | (.25) | 94 | (.62) | 3 | (450) |
| Cray T3D (PVM) | | | | | | | | 21 | (3150) |
| Intel Paragon | OSF 1.0.4 | 50 | (20) | 10 | (.20) | 34 | (.68) | 29 | (1450) |
| Intel Paragon | SUNMOS 1.6.2 | | | | | | | 25 | (1250) |
| Intel Delta | NX 3.3.10 | 40 | (25) | 9.8 | (.25) | 34 | (.85) | 77 | (3080) |
| Intel iPSC/860 | NX 3.3.2 | 40 | (25) | 9.8 | (.25) | 34 | (.85) | 65 | (2600) |
| Intel iPSC/2 | NX 3.3.2 | 16 | (63) | .37 | (.01) | – | (–) | 370 | (5920) |
| IBM SP-1 | MPL | 62.5 | (16) | 38 | (.61) | 104 | (1.66) | 270 | (16875) |
| IBM SP-2 | MPI | 66 | (15.15) | 130 | (1.97) | 236 | (3.58) | 35 | (2310) |
| KSR-1 | OSF R1.2.2 | 40 | (25) | 15 | (.38) | 31 | (.78) | 73 | (2920) |
| Meiko CS2 (MPI) | Solaris 2.3 | 90 | (11.11) | 24 | (.27) | 97 | (1.08) | 83 | (7470) |
| Meiko CS2 (sm) | | | | | | | | 11 | (990) |
| nCUBE 2 | Vertex 2.0 | 20 | (50) | .78 | (.04) | 2 | (.10) | 154 | (3080) |
| nCUBE 1 | Vertex 2.3 | 8 | (125) | .10 | (.01) | – | (–) | 384 | (3072) |
| NEC Cenju-3 | Env Rev 1.5d | 75 | (13.3) | 23 | (.31) | 39 | (.52) | 40 | (3000) |
| NEC Cenju-3(sm) | Env Rev 1.5d | 75 | (13.3) | 23 | (.31) | 39 | (.52) | 34 | (2550) |
| SGI Power Challenge | IRIX 6.1 | 90 | (11.11) | 126 | (1.4) | 308 | (3.42) | 10 | (900) |
| TMC CM-5 | CMMD 2.0 | 32 | (31.25) | – | (–) | – | (–) | 95 | (3040) |








# MIMD, multicomputer: networked workstations

Enabling software technology: PVM (Parallel Virtual Machine) available from netlib@ornl.gov

Enabling software technology: MPI (Message Passing Interface) available from netlib@ornl.gov

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

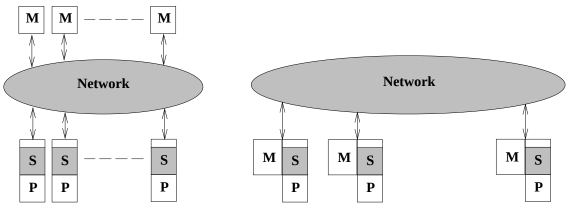


# Message-Passing Implementations

- 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 <u>full</u> cost array
    - one <u>owned</u> region, plus potentially <u>stale</u> copy of others
    - send frequent updates so that copies not too stale
  - Consequences:
    - waste of memory in replication
    - stale data $\Rightarrow$ poorer quality results or more iterations
- In either case, lots of thinking needed on programmers part



**Scalability**

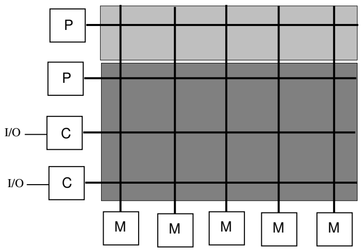



- Latency, Bandwidth, and Cost
- Construct shared memory out of simple message transactions across a general–purpose network.
  - read–request, read–response
- Caching?

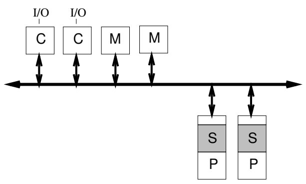


**Hardware Organizations**

- "Mainframe" approach
  - motivated by multiprogramming
  - extends xbar
  - latency
  - bandwidth
  - cost




- "Minicomputer" approach
  - motivated by multiprogramming and transaction processing
  - used heavily for parallel comp.
  - latency
  - bandwidth
    >> caching is key
  - cost



## 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 DGEFA 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.

Interested in ...

| | |
|---|---|
| $R_{max}$ | Gflop/s for largest problem |
| $N_{max}$ | size of largest problem |
| $N_{1/2}$ | size where half the $R_{max}$ achieved |
| $R_{peak}$ | theoretical peak performance |

Must get the "correct results".

The results were checked for accuracy by calculating a residual for the problem $||Ax - b||/(||A||||x||)$.

The executation rate is computed using $2/3n^3 + 2n^2$ operations.





Table 1: **Performance in Solving a System of Linear Equations**

| Computer | "LINPACK Benchmark" n = 100 OS/Compiler | Mflop/s | "TPP" Best Effort n=1000, Mflop/s | "Theoretical Peak" Mflop/s |
|---|---|---|---|---|
| Cray T916 (16 proc. 2.2 ns) | cf77 (6.0) -Zp -Wd-68 | | 19490 | 28800 |
| Cray T916 (8 proc. 2.2 ns) | cf77 (6.0) -Zp -Wd-68 | | 10800 | 14400 |
| Cray T916 (4 proc. 2.2 ns) | cf77 (6.0) -Zp -Wd-68 | | 5711 | 7200 |
| Cray T916 (2 proc. 2.2 ns) | cf77 (6.0) -Zp -Wd-68 | | 2943 | 3600 |
| Cray T916 (1 proc. 2.2 ns) | cf77 (6.0) -Zp -Wd-68 | 522 | 1576 | 1800 |
| Cray C90 (16 proc. 4.2 ns) | CF77 5.0 -Zp -Wd-e68 | 479 | 10780 | 15238 |
| Cray C90 (8 proc. 4.2 ns) | CF77 5.0 -Zp -Wd-e68 | 468 | 6175 | 7619 |
| Cray 3-128 (4 proc. 2.11 ns) | CSOS 1.0 level 129 | 421 | 2862 | 3792 |
| Hitachi S-3800/480(4 proc 2 ns) | | | 20640 | 32000 |
| Hitachi S-3800/380(3 proc 2 ns) | | | 16880 | 24000 |
| Hitachi S-3800/280(2 proc 2 ns) | | | 12190 | 16000 |
| Hitachi S-3800/180(1 proc 2 ns) | OSF/1 MJ FORTRAN:V03-00 | 408 | 6431 | 8000 |
| Cray 3-128 (2 proc. 2.11 ns) | CSOS 1.0 level 129 | 393 | 1622 | 1896 |
| Cray C90 (4 proc. 4.2 ns) | CF77 5.0 -Zp -Wd-e68 | 388 | 3275 | 3810 |
| Cray C90 (2 proc. 4.2 ns) | CF77 5.0 -Zp -Wd-e68 | 387 | 1703 | 1905 |
| Cray C90 (1 proc. 4.2 ns) | CF77 5.0 -Zp -Wd-e68 | 387 | 902 | 952 |
| NEC SX-3/44R (4 proc. 2.5 ns) | | | 15120 | 25600 |
| NEC SX-3/42R (4 proc. 2.5 ns) | | | 8950 | 12800 |
| NEC SX-3/41R (4 proc. 2.5 ns) | | | 4815 | 6400 |
| NEC SX-3/34R (3 proc. 2.5 ns) | | | 12730 | 19200 |
| NEC SX-3/32R (3 proc. 2.5 ns) | | | 6718 | 9600 |
| NEC SX-3/31R (3 proc. 2.5 ns) | | | 3638 | 4800 |
| NEC SX-3/24R (2 proc. 2.5 ns) | | | 9454 | 12800 |
| NEC SX-3/22R (2 proc. 2.5 ns) | | | 5116 | 6400 |
| NEC SX-3/21R (2 proc. 2.5 ns) | | | 2627 | 3200 |
| NEC SX-3/14R (1 proc. 2.5 ns) | f77sx 040 R2.2 -pi*:* | 368 | 5199 | 6400 |
| NEC SX-3/12R (1 proc. 2.5 ns) | f77sx 040 R2.2 -pi*:* | 368 | 2757 | 3200 |
| Cray 3-128 (1 proc. 2.11 ns) | CSOS 1.0 level 129 | 327 | 876 | 948 |
| NEC SX-3/44 (4 proc. 2.9 ns) | | | 13420 | 22000 |
| NEC SX-3/24 (2 proc. 2.9 ns) | | | 8149 | 11000 |
| NEC SX-3/42 (4 proc. 2.9 ns) | | | 7752 | 11000 |
| NEC SX-3/22 (2 proc. 2.9 ns) | | | 4404 | 5500 |
| NEC SX-3/14 (1 proc. 2.9 ns) | f77sx 020 R1.13 -pi*:* | 314 | 4511 | 5500 |
| NEC SX-3/12 (1 proc. 2.9 ns) | f77sx 020 R1.13 -pi*:* | 313 | 2283 | 2750 |
| Cray Y-MP/832 (8 proc. 6 ns) | CF77 4.0 -Zp -Wd-e68 | 275 | 2144 | 2667 |
| Fujitsu VP2600/10 (3.2 ns) | FORTRAN77 EX/VP V11L10 | 249 | 4009 | 5000 |
| Cray Y-MP/832 (4 proc. 6 ns) | CF77 4.0 -Zp -Wd-e68 | 226 | 1159 | 1333 |
| Fujitsu VPP500/1(1 proc. 10 ns) | FORTRAN77EX/VP V12L20 | 206 | 1490 | 1600 |
| Cray Y-MP M98 (8 proc. 6 ns) | CF77 5.0 -Zp -Wd-e68 | 204 | 1733 | 2666 |
| Fujitsu VP2200/10 (3.2 ns) | FORTRAN77 EX/VP V12L10 | 203 | 1048 | 1250 |
| Cray 2S/4-128 (4 proc. 4.1 ns) | CSOS 1.0 level 129 | 202 | 1406 | 1951 |
| NEC SX-3/11R (1 proc. 2.5 ns) | f77sx 040 R2.2 -pi*:* | 202 | 1418 | 1600 |
| NEC SX-3/1LR (1 proc. 2.5 ns) | f77sx 040 R2.2 -pi*:* | 201 | 767 | 800 |
| Cray Y-MP/832 (2 proc. 6 ns) | CF77 5.0 -Zp -Wd-e68 | 181 | 604 | 667 |





Table 3: **Highly Parallel Computing**

| Computer (Full Precision) | Number of Processors | $R_{max}$ Gflop/s | $N_{max}$ order | $N_{1/2}$ order | $R_{peak}$ Gflop/s |
|---|---|---|---|---|---|
| Intel Paragon XP/S MP (50 MHz OS=SUNMOS) | 6768 | 281.1 | 128600 | 25700 | 338 |
| Intel Paragon XP/S MP (50 MHz OS=SUNMOS) | 6144 | 256.2 | 122500 | 24300 | 307 |
| Intel Paragon XP/S MP (50 MHz OS=SUNMOS) | 5376 | 223.6 | 114500 | 22900 | 269 |
| Intel Paragon XP/S MP (50 MHz OS=SUNMOS) | 4608 | 191.5 | 106000 | 21000 | 230 |
| Numerical Wind Tunnel* (9.5 ns) | 140 | 170.4 | 42000 | 13800 | 236 |
| Numerical Wind Tunnel* (9.5 ns) | 128 | 157.9 | 40960 | 13120 | 216 |
| Intel Paragon XP/S MP (50 MHz OS=SUNMOS) | 3648 | 151.7 | 95000 | 18100 | 182 |
| Fujitsu VPP500/128(10nsec) | 128 | 149.7 | 40960 | 13120 | 205 |
| Intel Paragon XPS-140 (50 MHz OS=SUNMOS) | 3680 | 143.4 | 55700 | 20500 | 184 |
| Paragon XP/S MP(1024 Nodes, OS=SUNMOS S1.6) | 3072 | 127.1 | 86000 | 17800 | 154 |
| Fujitsu VPP500/100(10nsec) | 100 | 120.4 | 36000 | 11450 | 160 |
| Fujitsu VPP500/96 (10nsec) | 96 | 116.2 | 35712 | 11200 | 154 |
| Cray T3D 1024 (150 MHz) | 1024 | 100.5 | 81920 | 10224 | 152 |
| Fujitsu VPP500/80 (10nsec) | 80 | 98.9 | 32640 | 10050 | 128 |
| IBM SP2-T2 (66 MHz) | 512 | 88.4 | 73500 | 20150 | 136 |
| Fujitsu VPP500/64 (10nsec) | 64 | 81.1 | 29440 | 8950 | 102 |
| Hitachi S-3000 cluster/412 (3x4) (2 ns) | 12 | 78.2 | 31120 | 4880 | 96 |
| Intel Paragon XPS-140 (50 MHz) | 1872 | 72.9 | 55000 | 17500 | 94 |
| Paragon XP/S MP(512 Nodes, OS=SUNMOS S1.6) | 1516 | 64.0 | 61000 | 12200 | 77 |
| Thinking Machines CM-5 | 1024 | 59.7 | 52224 | 24064 | 131 |
| Hitachi S-3000 cluster/309 (3x3) (2 ns) | 9 | 59.0 | 26940 | 3180 | 72 |
| Hitachi S-3000 cluster/408 (2x4) (2 ns) | 8 | 54.1 | 31200 | 3760 | 64 |
| Cray T3D 512 (150 MHz) | 512 | 50.8 | 57856 | 7136 | 76 |
| IBM SP2-T2 (66 MHz) | 256 | 44.2 | 53000 | 13500 | 68 |
| Fujitsu VPP500/32 (10nsec) | 32 | 42.4 | 20736 | 4940 | 51 |
| Hitachi S-3000 cluster/306 (2x3) (2 ns) | 6 | 40.9 | 27000 | 2400 | 48 |
| Hitachi S-3000 cluster/206 (3x2) (2 ns) | 6 | 40.6 | 21600 | 2160 | 48 |
| Thinking Machines CM-5 | 512 | 30.4 | 36864 | 16384 | 66 |
| Paragon XP/S MP(256 Nodes, OS=SUNMOS S1.6) | 768 | 31.7 | 43500 | 8400 | 38 |
| IBM SP2-T2 (66 MHz) | 160 | 28.7 | 42200 | 10300 | 42 |
| Hitachi S-3800/480 (2 ns) | 4 | 28.4 | 15500 | 830 | 32 |
| Hitachi S-3000 cluster/204 (2x2) (2 ns) | 4 | 27.9 | 21600 | 1640 | 32 |
| Hitachi S-3000 cluster/404 (1x4) (2 ns) | 4 | 27.2 | 31200 | 2680 | 32 |
| SGI POWER CHALLENGE (90 MHz) | 128 | 26.7 | 53000 | 20000 | 46 |
| Cray T3D 256 (150 MHz) | 256 | 25.3 | 40960 | 4918 | 38 |
| NEC SX-3/44R (2.5 ns) | 4 | 23.2 | 6400 | 830 | 26 |
| IBM SP2-T2 (66 MHz) | 128 | 22.9 | 37000 | 9200 | 34 |
| Fujitsu VPP500/16 (10nsec) | 16 | 21.7 | 14592 | 3090 | 26 |
| Hitachi S-3800/380 (2 ns) | 3 | 21.6 | 15680 | 760 | 24 |
| Hitachi S-3000 cluster/303 (1x3) (2 ns) | 3 | 21.5 | 27000 | 1560 | 24 |
| NEC SX-3/44 (2.9 ns) | 4 | 20.0 | 6144 | 832 | 22 |
| SGI POWER CHALLENGE (75 MHz) | 96 | 18.5 | 53000 | 20000 | 29 |
| NEC SX-3/34R (2.5 ns) | 3 | 17.4 | 6144 | 691 | 19 |
| Paragon XP/S MP(128 Nodes, OS=SUNMOS S1.6) | 384 | 16.0 | 30700 | 5700 | 19 |
| SGI POWER CHALLENGE (90 MHz) | 64 | 15.6 | 37000 | 8500 | 37 |