Jack Dongarra

Computer Science Department
University of Tennessee

and

Mathematical Sciences Section
Oak Ridge National Laboratory

Work supported in part by ARPA, Department of Energy, and the National Science Foundation and in cooperation with the Center for Research on Parallel Computing.

I. Introduction to Parallel Computing

08:30	**Architecture of Parallel Computers**
	Shared Memory Architecture
	Local Memory Architecture
	Virtual Shared Memory Architecture
	Examples of Existing Parallel Computers
	Introduction to Parallel Programming
	Shared Memory Parallization
	Local Memory Parallization
	Message Passing
10:00	**Coffee and Communication Break**

II. Portable High Peformance Software

10:30	**Parallel Virtual Machine - PVM**
	Message Passing Interface - MPI
	Definition and Status of Standardization
	Usage and Programming
	Status of the Implementation for
	Existing Parallel Computers
12:00	**Lunch**

13:30 **Performance of Parallel Programs**
Linpack Bechmark
NASA Parllel Bechmark
ParkBench
Portability of High Performance Software
Quality Aspects of Parallel Software

15:00 **Coffee and Communication Break**

15:30 **Libraries**
LAPACK
ScalAPACK
BLACS
National High Performance Software Exchange

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,** 2×10^{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**

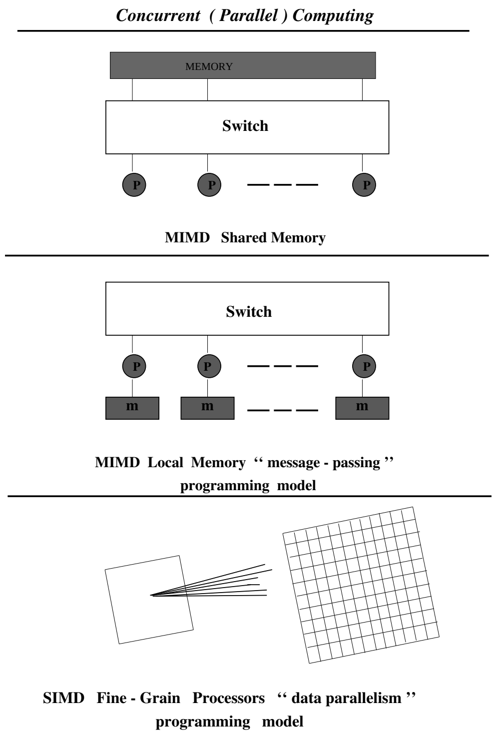
Scalable Multiprocessors

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**



Architecture Alternatives

- **Distribute, private memories using message-passing to communicate among processors.**

- **Single address space implemented with physically distributed memories.**

- **Both approaches require:**
  - **Attention to locality of access.**
  - **Scalable interconnection technology.**

Why Compute Distributedly?

Scalar O(1)

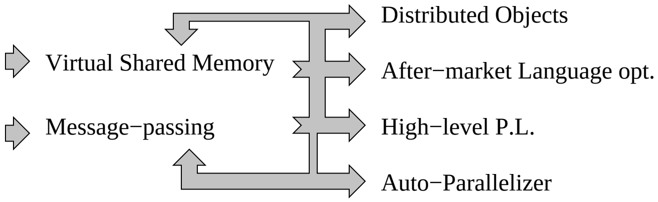
↓

Closely−Coupled Parallel O(10)

↓

Distributed O(100? 1000? 10000?)

Distributed Computing



Nevermid all that: Focus on Message−passing

Build a good message−passing system

Can write MP applications

Can build other abstraction on top

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

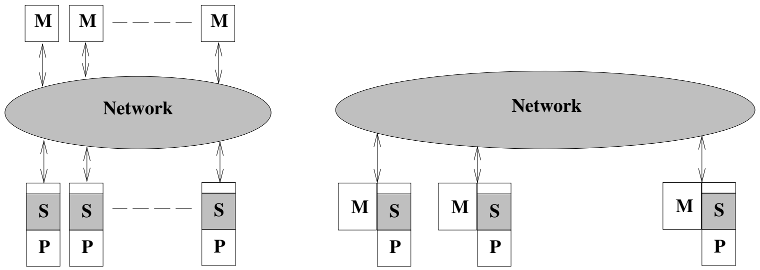
Comparison

- 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

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?

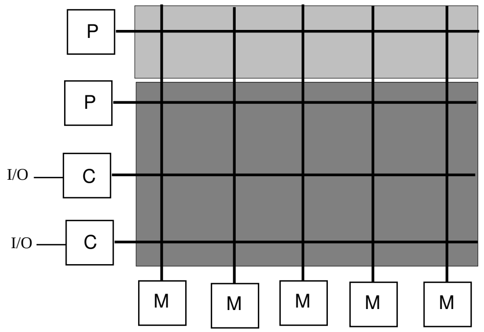
Scalability



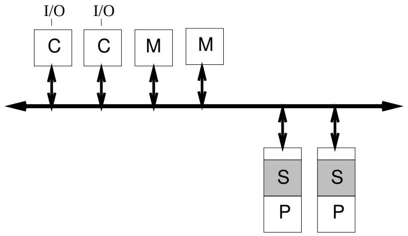
- Latency, Bandwidth, and Cost
- Construct shared memory out of simple message transactions across a general−purpose network.
  - read−request, read−response
- Caching?

Hardware Organizations

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

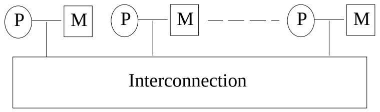


- "Minicomputer" approach
  - motivated by multiprogramming and transaction processing
  - used heavily for parallel comp.
  - latency
  - bandwidth
    >> caching is key
  - cost

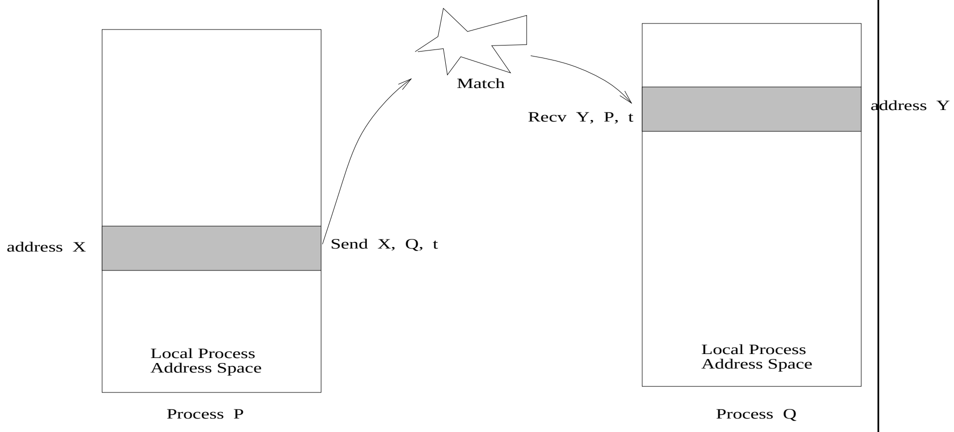


Message Passing Architectures

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



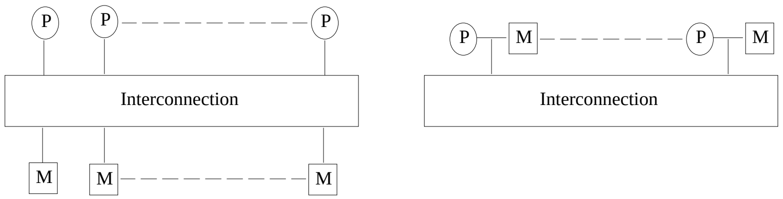
Message Passing Abstraction



- Send + Matching Recv provides a memory−memory copy plus a pairwise synchronization event.
- Substantial layer of software between user primitives and HW organization.
- Highly structured programs (SPMD) are common.
- Sending a message often incurs many overheads:
  - Building a header; copying data into network buffers; sending data; receiving data into buffers; copying data from kernel to user space.

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.

-

Conclusions on Interconnects

- High bandwidth interconnects remain the major challenge for MPP systems
- We continue to have the "Tflops" hang-up, and focus attention on the wrong issue
- However, clusted workstations will provide an efficient through put system for small and medium range jobs
- Latency is more critical than bandwidth for system performance

Naming

- Single Global Linear-Address-Space (shared memory)
- Single Global Segmented-Address-Space (global objects)
- Multiple Local Address/Name Space (message passing)
- Naming strategy has implications on:
  - Programmer/Software
  - Performance
  - Design Complexity

Shared-Memory Implementation

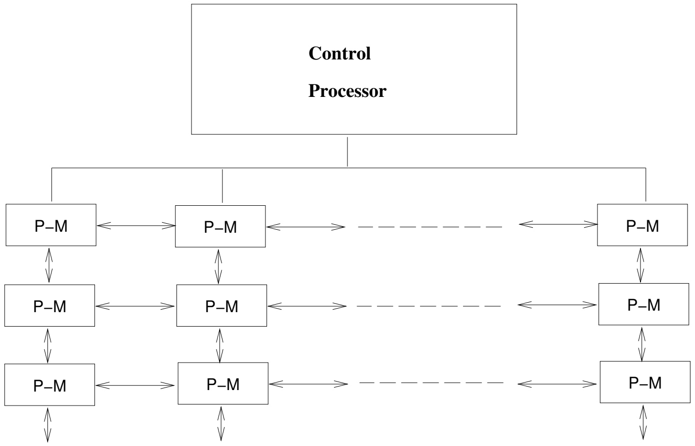
- Shared memory algorithm:
  - Divide cost-array into regions
    - logically assign regions to PEs
  - Assign wires to PEs based on the region in which center lies
  - Do load balancing using stealing when local queue empty
- Good points:
  - Good load balancing
  - Mostly local accesses
  - High cache-hit 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 ⇒ poorer quality results or more iterations
- In either case, lots of thinking needed on programmers part

Data Parallel Architectures

- Programming model
  - operations are performed on each element of a large (regular) data stucture in a single step
  - arithmetic, global data transfer
- Processor is logically associated with each data element, general communication, and cheap global synchronization.
  - driven originally by simple O.D.E.



Survey of Current Machines

MIMD, global memory

- Cray Y–MP C916, T90, J90
- NEC SX–3 and 4, Hitachi S820, Fujitsu VP2600

MIMD, distributed memory

- Cray T3D
- Fujitsu VPP500

MIMD, multicomputer

- IBM SP–2
- Intel iPSC/860, Delta, Paragon
- Meiko CS–2
- nCUBE 2s, nCUBE3s
- Parsytec GigaCube GC–2

SIMD/MIMD

- Thinking Machines CM–2, CM–5

SIMD

- 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	4.0 ns	1,000	16	16,000
Cray T90	2.0 ns	2,000	32	64,000
NEX SX–3	2.9 ns	5,500	4	22,000
NEX SX–4	8.0 ns	2,000	32	64,000
	8.0 ns	2,000	512	1,024,000
Fujitsu VP2600	4.0 ns	4,000	1	4,000
Hitachi S–3800/480		8,000	4	32,000

MIMD, global: Comparison

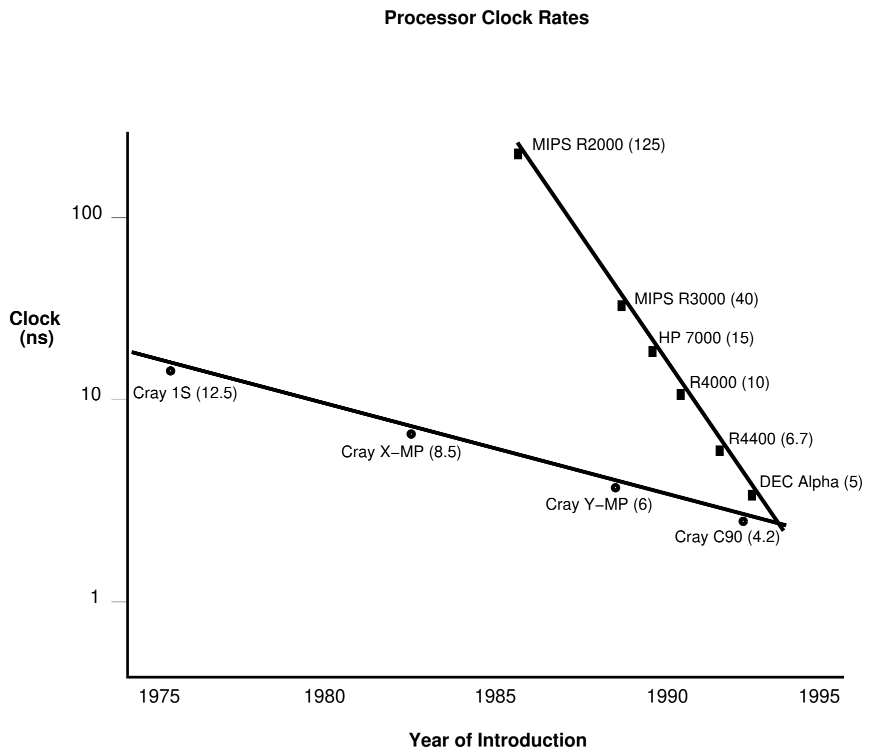
Computer	Number of pipes/proc
C90	4
VP–2600	8
S–820	8
SX–3	16
SX–4	16

Shared memory vector supercomputers use paallelism in three ways
– multiple floating point pipes
– small number of powerful vector processors
– vector pipelines, which can be considered as form of SIMD parallelism

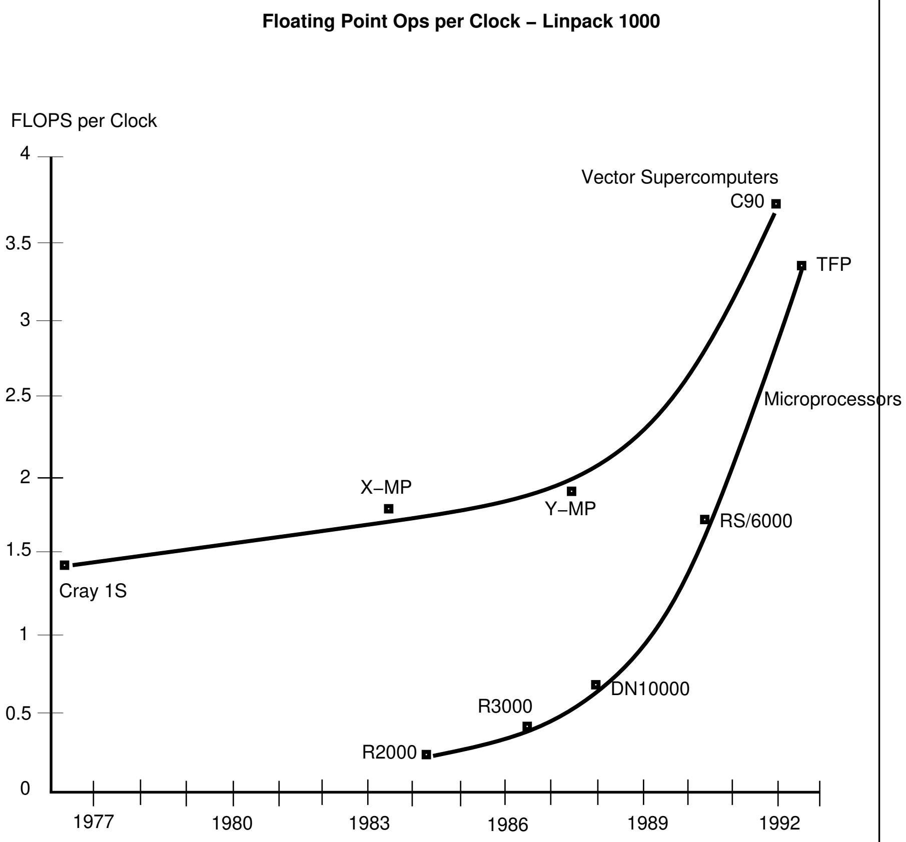
Comparison RISC Micros and Y-MP

	Y-MP 167Mhz	i860 40 Mhz	RS6000 62.5 Mhz	Alpha 150 Mhz
divide clocks	4	38	20	63
cache size (KB)	none	8	64	8
memory bandwidth (Mwords/sec)	500	17	125	37.5
peak Mflops	330	60	125	150
ratio	1.5	0.28	1.0	0.25

Scientific applications need at least a ratio of 1.0
Alpha repeats the mistakes of the i860



Processor Clock Rates
Clock (ns)
100
10
1
MIPS R2000 (125)
MIPS R3000 (40)
HP 7000 (15)
R4000 (10)
R4400 (6.7)
DEC Alpha (5)
Cray 1S (12.5)
Cray X−MP (8.5)
Cray Y−MP (6)
Cray C90 (4.2)
1975
1980
1985
1990
1995
Year of Introduction



Floating Point Ops per Clock − Linpack 1000
FLOPS per Clock
4
3.5
3
2.5
2
1.5
1
0.5
0
Vector Supercomputers
C90
TFP
Microprocessors
X−MP
Y−MP
RS/6000
Cray 1S
DN10000
R3000
R2000
1977
1980
1983
1986
1989
1992

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.

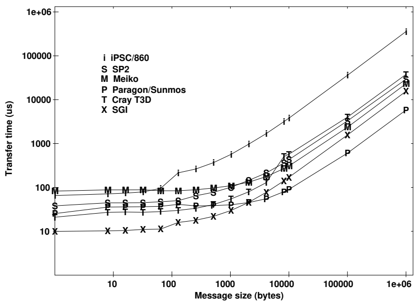
Machine	MHz	Cycle time nsec	Linpack n=100 Mflops	% peak	Ax = b n=1000 Mflops	% peak	Theor Peak Mflops
DEC Alpha	300	3.3	140	23%	411	69%	600
IBM Power2	66	15	130	48%	236	89%	266
SGI Power	75	13.3	101	33%	260	87%	300
HP 735	100	10	61	31%	107	54%	198
DEC Alpha	200	5	43	22%	155	78%	200
DEC Alpha	182	5	39	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	500	2	???	??%	???	??%	2000
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

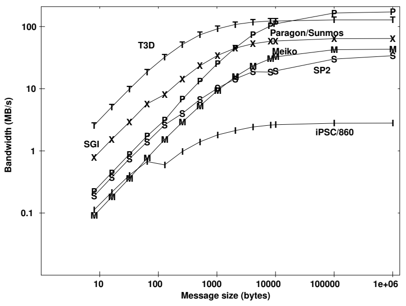
Table 1: Multiprocessor Latency and Bandwidth.

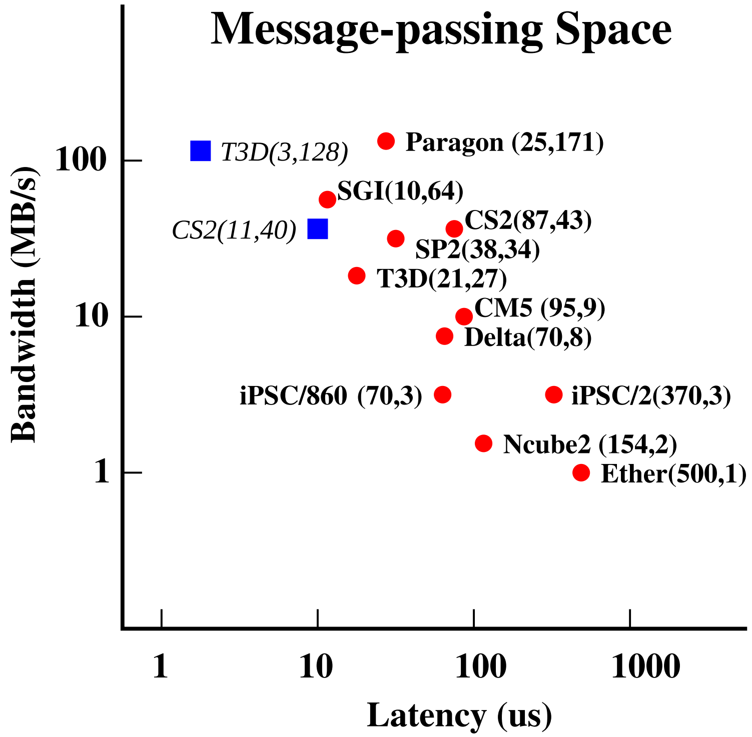
Machine	OS	Latency $n = 0$ (μs)	Bandwidth $n = 10^6$ (MB/s)	$n_{1/2}$ bytes	Theoretical Bandwidth (MB/s)
Cray T3D (sm)	MAX 1.2.0.2	3	128	363	300
Cray T3D (PVM)	MAX 1.2.0.2	21	27	1502	300
Intel Paragon	OSF 1.0.4	29	154	7236	175
Intel Paragon	SUNMOS 1.6.2	25	171	5856	175
Intel Delta	NX 3.3.10	77	8	900	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
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)
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)
TMC CM-5	CMMD 2.0	32	(31.25)	–	(–)	–	(–)	95	(3040)



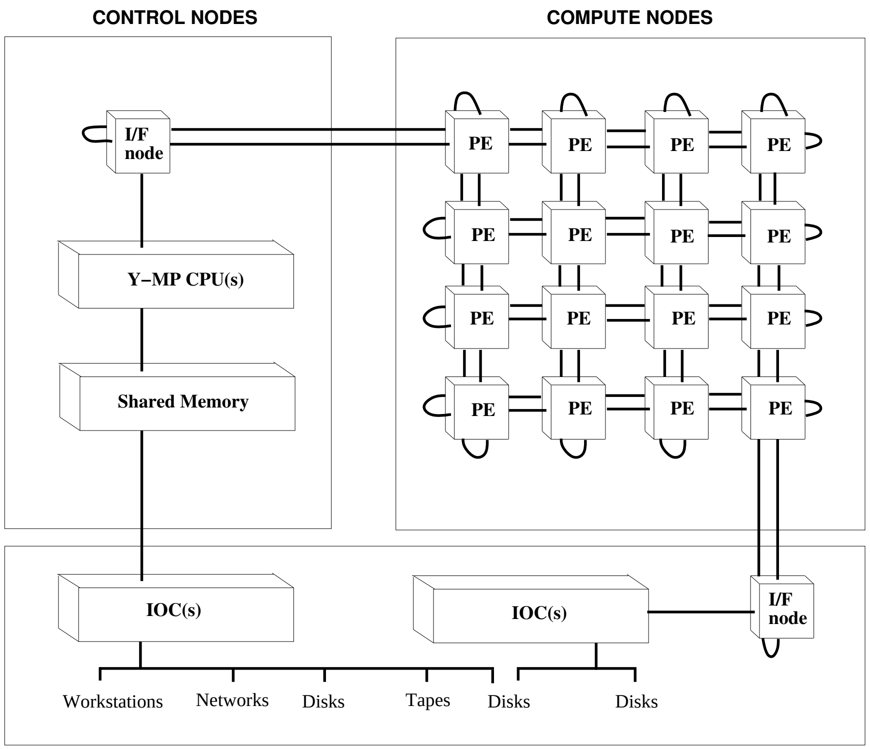

T3D
Paragon/Sunmos
Meiko
SP2
SGI
iPSC/860
Bandwidth (MB/s)
Message size (bytes)


Message-passing Space
T3D(3,128)
Paragon (25,171)
SGI(10,64)
CS2(11,40)
CS2(87,43)
SP2(38,34)
T3D(21,27)
CM5 (95,9)
Delta(70,8)
iPSC/860 (70,3)
iPSC/2(370,3)
Ncube2 (154,2)
Ether(500,1)
Bandwidth (MB/s)
Latency (us)

MIMD distributed – Cray T3D

- Three year development plan with new generations of machines projected to be available in 1993 (150 GFLOPS Peak), 1995 (1 TFLOPS Peak), 1997 (1 TFLOPS sustained).
- The MPP component will be closely coupled with a vector/scalar front end Y-MP.
- The first generation MPP PE is based on the DEC Alpha microprocessor providing a peak of 150 MFLOPS per PE.
- Future generations may use a different micro-architecture.
- First MPP scales up to 100s of GFLOPS peak.
- Physically distributed memory with 8 - 32 Mbytes of memory per processor.

MIMD, distributed : Cray T3D



MIMD distributed – Cray MPP

Technical Strengths

- experience in scientific computing (hardware, software, applications, customer support)
- existing user base to build on

Technical Challenges

- integration of outside technology (DEC Alpha)
- use of Y-MP as front end

MIMD Multicomputer – Intel Paragon

- Uses i860XP microprocessor at 75 Mflops
- Up to 4000 processors 2D mesh
- Aggregate performance over 300 Gflops
- 16 - 64 Mbyte memory per processor
- New message routing chip at 200 Mbytes/sec

MIMD Multicomputer – Intel Paragon

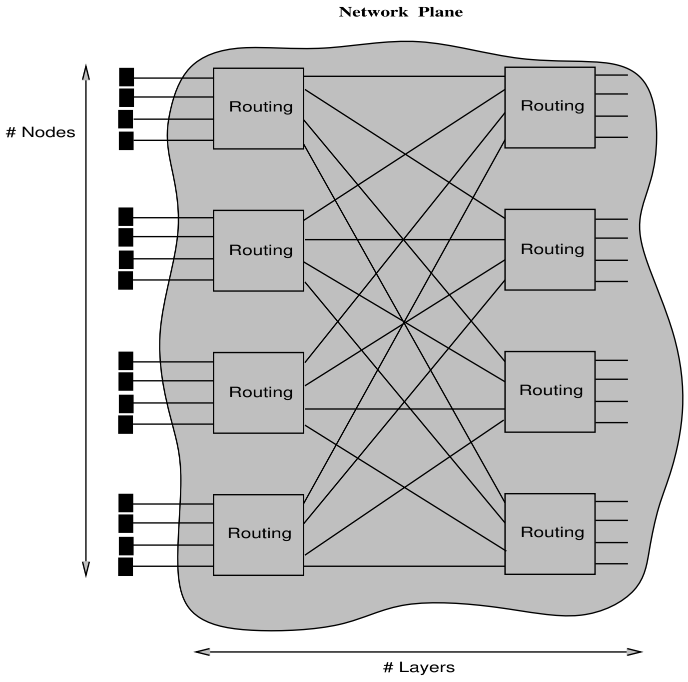
Technical Strengths

- long experience and user base in parallel processing
- "free" hardware development in house

Technical Challenges

- software still immature
- weak I/O system
- transition to real supercomputer company
- Legacy micro (i860)

MIMD, multicomputer : Meiko CS–2



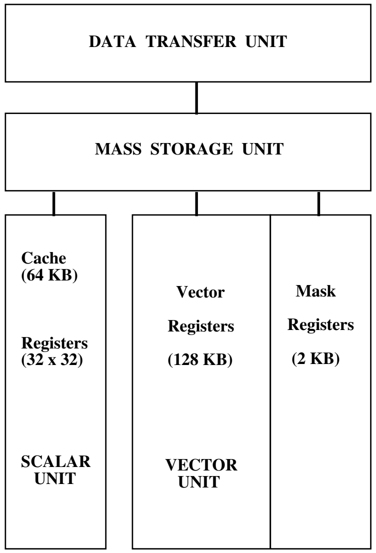
- 16 processors
- 2 switch layers

∴ Bisectional bandwidth

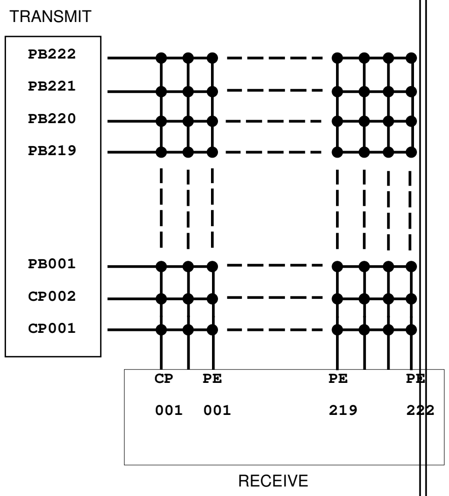
= 1/2 x 16 x 100

= 800 MB / sec

MIMD distibuted : Fujitsu VPP500



Fujitsu VPP500 Processing Elements

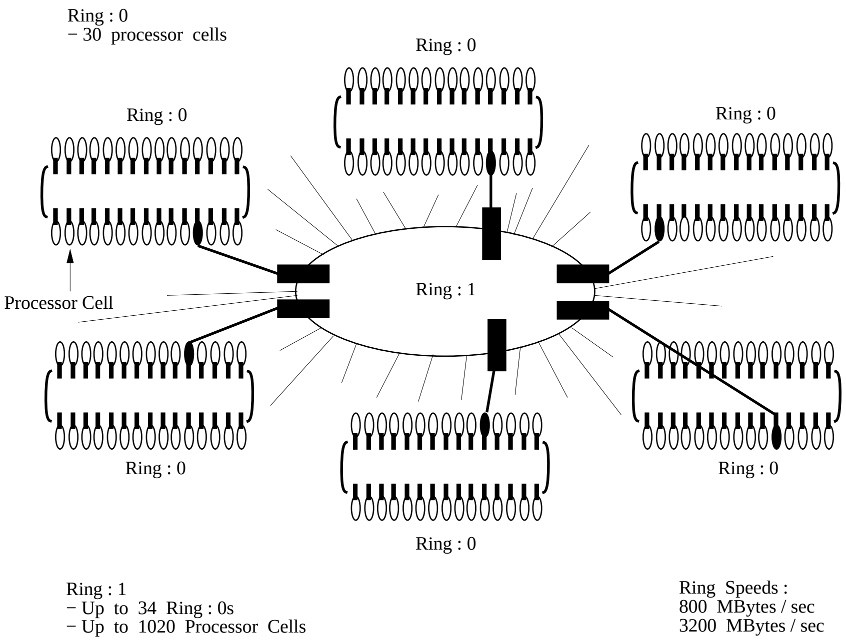


Fujitsu VPP500 Crossbar

MIMD distributed: Fujitsu VPP500

Processors		comments
chips	cpu:	GaAs, 25K gates (60 ps/gate)
		vector units (6-pipes)
clock	100 MHz	
arithmetic	32/64-bit IEEE	
Memory		
PE	128-256 MB BiCMOS	72K gates, 1Mbit RAM
		(200 ps/gate)
	64 KB cache	
Topology		
no. processors (p)	7 to 222	crossbar network
b/w	800 MB/s	
peak Gflop/s	11.2 to 355.2	32-bit (7 to 222 PE's)
I/O	*no data*	

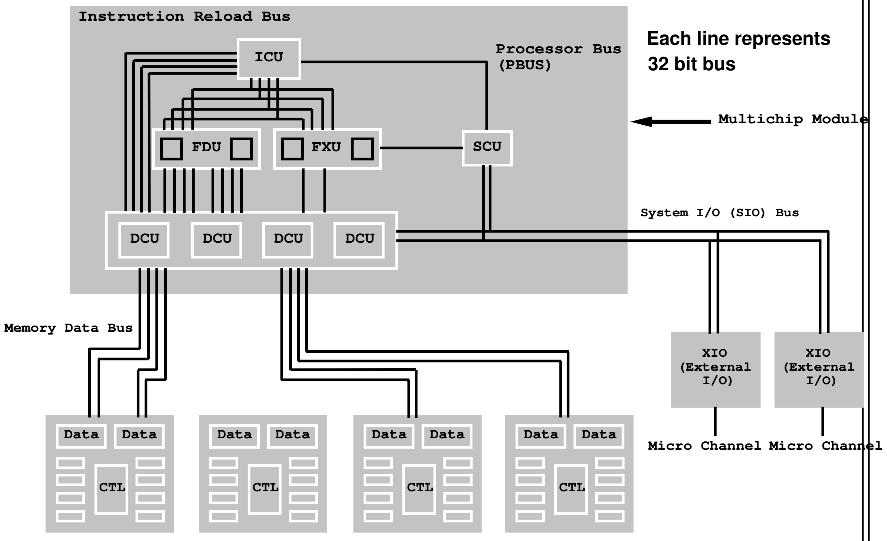
MIMD, distributed : KSR-1



New Features of IBM RS 6000/590

- Third generation RISC
- Dual fixed point, dual floating point and single branch units
- Up to 6 instructions per cycle (8 operations)
  - branch
  - Condition register
  - 2 fixed point
  - 2 floating (4 operations, if multiply and add)
- New instructions
  - Floating point SQRT
  - Convert to integer
- Improvements
  - higher cache/memory bandwidth
  - Reduced translation latency

Block Diagram of IBM RS 6000/590



Power2 Processing Units

Two Branch Processing Units (BPU)

⇒ read intruction, dispatches instructions, perform condition register logic, and perform branches.

Two Fixed Point Units (FXU)

⇒fixed point arithmetic and compare operations, logical and shift operations, address generation, load/store for all data types, etc.

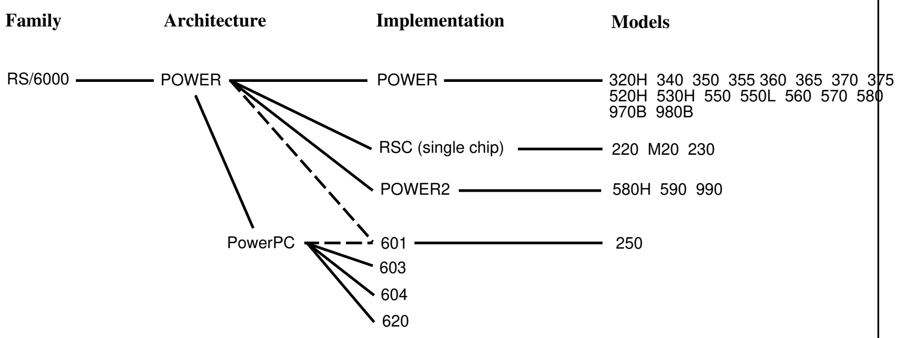
Two Floating Point Units (FPU)

⇒ floating point arithmetic and compare operation.

RS 6000/590 Microprocessor – POWER2

Characteristic	Specification
Type of microprocessor	RS 6000/590, also called POWER2
Technology	CMOS, 0.45 micron
Number of transistors	23 million
Physical dimensions	12.7x12.7 mm, 8 chips on MCM
Clock speed	66.5 MHz
Mflop rate of one microprocessor	266 Mflop/s, Four flops per cycle
Data cache	256 KB, Four–way set associative
Instruction cache	32 KB, Two–way set associative
Latency of data cache to memory	15 cycles
Bandwidth of data cache to memory	2128 MB/second – 32 bytes/cycle
Floating point instruction unit	IEEE – POSIX standard
Issue rate	Six instructions per clock cycle
Number of floating point registers	54 physically but only 32 logical
Size of floating point registers	64 bit
Number of integer registers	32
Size of integer registers	32 bit

RISC System/6000 Family



The SP2 High-Performance Switch

- Multistage, buffered wormhole routing packet-switch
- Based upon Vulcan switch chip section, it's a 170K gate ASIC, CMOS, 1.0 micron process with approximately 170 pins
- Scales up to 128 nodes, or up to 512 nodes upon special request
- The rough stage relationship is O(log_4n) where n is number of nodes. SP2 switch uses
  - 1 switch board per 16 nodes for 1 – 80 nodes
  - 1.5 boards for 65 – 512 nodes. 1.5 boards means that for every 32 nodes there is a second-stage board.
  - 2 boards for 129 – 512 nodes
- THe SP2 adapters have 1 processor (intel i860), the SP1 adapters have none. Processors are used to provide intelligence that can off-load work from the main processors (RS 6000/590) so that it can compute during some communication operations.

Latency or Bandwidth

Study by Van Voorst, Seidel, and Barsczc at NASA. Based on communication data for 14 days on iPSC/860 in production use

Latency in micro sec.	Communication Speed Up Bandwidth in MB per second					
	2.8	10	20	40	80	160
75	1.0	2.2	2.8	3.2	3.5	3.6
50	1.2	2.7	3.7	4.5	5.0	5.3
25	1.3	3.6	5.4	7.3	8.9	10
1	1.4	5.0	10	19	36	64

Current Machines	Message Passing Library	Latency micro sec.	Bandwidth MB/s
Cray T3D	SHEMM_GET/ PUT	1.5	30/120
IBM SP2	MPL	40	32

Bisection Bandwidth and Memory Bandwidth

Machine	Number or Processors	Bisection Bandwidth in GB /s	Bisection Bandwidth/ processors in MB/ s
Cray T3D	128	9.6	75
IBM SP−1	64	3.2	50
Intel Paragon	208	3.2	16
CM−5	128	1.3	10

Cray C90 (1 processor) has 11.5 GB/s memory bandwidth

Note: These are "peak" numbers for the machines

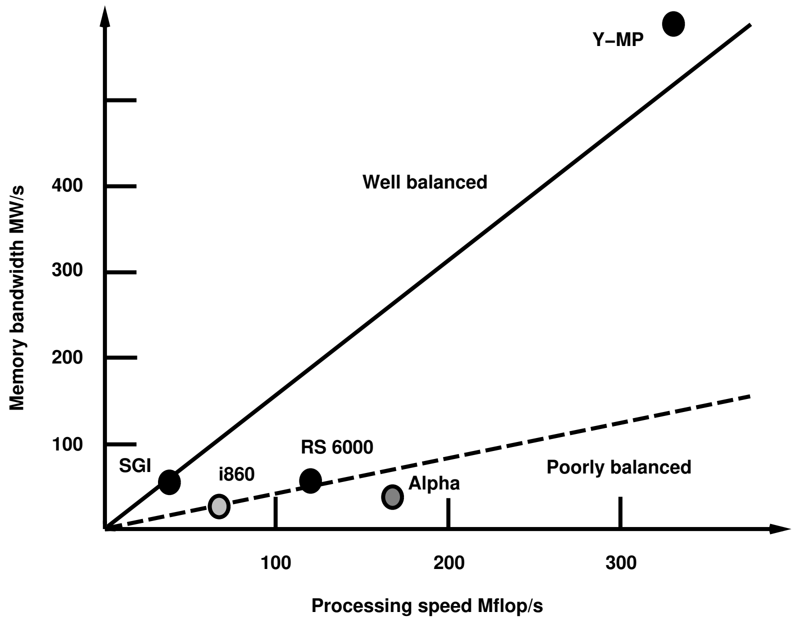
Comparison of RISC Micros and Y−MP

Characteristics /Machines	Cray Y−MP	Intel i860	RS 6000	Alpha
Clock in MHz	167	40	62.5	150
Divide Clocks	4	38	20	63
Cache Size (KB)	none	8	64	8
Memory Bandwidth in MW/sec.	500	17	125	37.5
Peak Mflop/s	330	60	125	150
Ratio	1.5	0.28	1.0	0.25

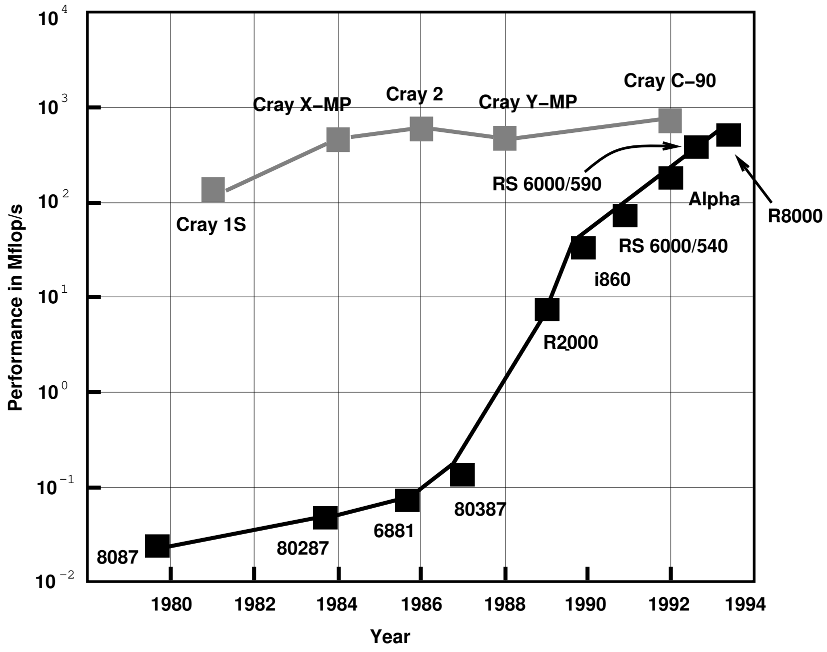
Scientific applications need at least a ratio of 1.0.

Alpha repeats the mistakes of i860

Comparison RISC Micros and Y–MP



Growth in Microprocessor Performance in 1980's



Who will win?

Four important factors independent of technology:

- financial resources
- management team
- time to market
- user support

Trends and Issues in the Quest for the Teraflop

- SIMD approach has been dropped
- MIMD distributed memory (KSR, Cray, Tera) has re-emerged as a valid alternative
- custom processors versus commodity micro processors is still an open issue
- the field is still full of vaporware and premature announcements
- a Tflops peak machine at a reasonable price ($25 M) by 1995 is feasible
- sustained performance of CFD codes on such a Tflops machine is probably in the 10's Gflops range
- networked workstations will provide the most useful and cost-effective

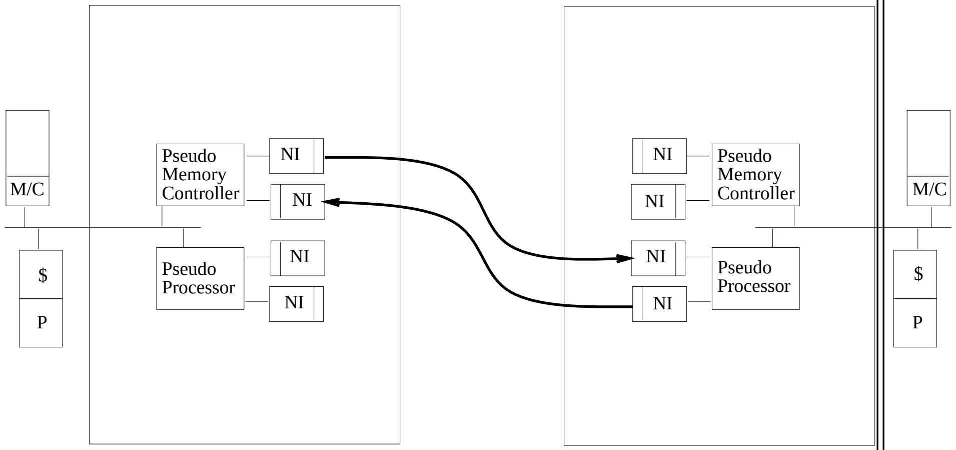
The Confusion in the Midrange

a) scaled down products from the big science market
– minisuper descendants (Convex)
– scaled down PVP (YMP EL)
– small scale MPP (Intel, TMC lowend machines)
b) scaled up products from the workstations market
– workstation servers
– MPP built on workstation technology
(IBM, Convex/HP, SGI)
– distributed computing (workstation farms)
c) new entries
– MPP for the midrange (KSR, MassPar)
– networking products

Projections for Big Science Market

– PVP and MPP will coexist
– difference will be in degree of parallelism
Cray Triton 64 proc @ 2 Gflops = 128 Gflops
Intel Paragon 2048 proc @ 80 Mflops = 160 Gflops
– system reliablility and software will be the discriminator

HW Complexity: Shared physical address space



Evolution and Convergence

- Early hypercubes provide FIFO on each link
  - HW close to Prog. Model
  - Soon replaced by DMA transfers and non-blocking operation
- Diminishing role of topology
  - store & forward in the comm. software, in the routers
  - general purpose (cut-through) rougint
- Constraint that only local addresses are used in artifact of P.M.
- User-level networks (CM-5)
- Complete message processor (Paragon, Meiko CS-2)
- User-to-user DMA
- Most shared-memory efforts now optimized toward message passing (Alewife, Flash)
  - A single-address space machine is specialized to handle memory read/write messages in hardware and is thus faster at such operations.

MIMD, multicomputer: networked workstations

Enabling software technology: PVM (parallel virtual machine) available from netlib@ornl.gov

very active research area; about 150 software products;

catalog available from Louis Turcotte at Miss. State

e-mail: turcotte@wes.army.mil

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.

MIMD, multicomputer: networked workstations

A number of vendors are offering clustering products (IBM, Convex/HP, SGI).

Advantage:

can handle easily increased computation demands from workstation environment.

Accomplishment:

SGI connected 16 Challenge Series 20 processors servers

with FDDI and obtained 5Gflop/s on a CFD calculation.

Networking Technology for Workstations

- **Currently:**

Network	Year	Bandwidth in Mbit/s	Cost in $
Ethernet	1981	10	100
FDDI	1989	100	1 – 2 K
ATM DS–3	1993	45	1,500

- **In the near future:**

Network	Year	Bandwidth in Mbit/s
ATM OC–3	1994(?)	155.2
FDDI	1994(?)	100/140

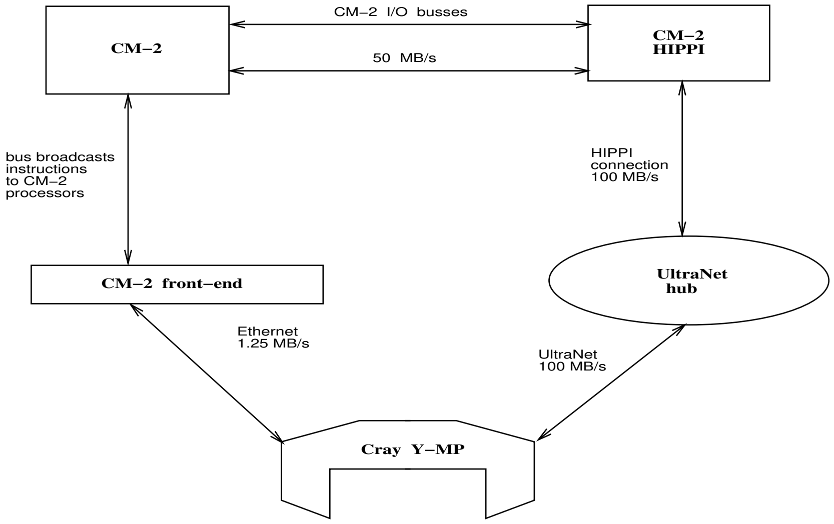
- **In two years:** ATM OC–12 622 Mbit/s

- At the times of availability ATM cards plus cost of hub will cost about $5K – $10K per workstation

MIMD, multicomputer : heterogeneous distributed environments

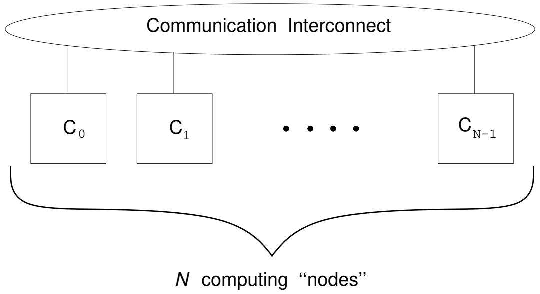
Heterogeneous distributed supercomputing architecture

MB/s = measure of peak bandwidth in megabytes per second



MIMD, multicomputer : networked workstations

- nodes are independent (micro)computers
- no shared resources
- explicit message passing



MIMD, multicomputer: heterogenous distributed environments

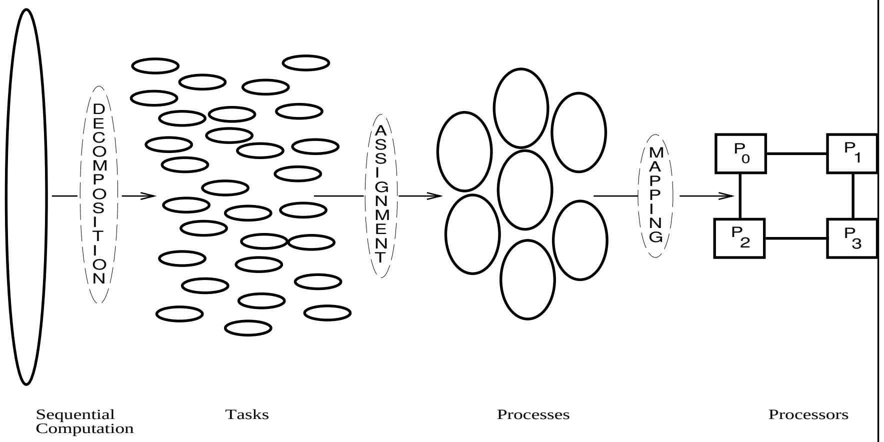
Work by J. Becker, UC Davis, and Leonardo Dagum, NAS Applied Research Branch

- particle sort accounts for 38% of the CM-2 computation time
- sorting is up to 10 times faster on one processor of the Cray Y-MP
- perform sort operation on the Y-MP using the HIPPI-UltraNet connection between the two machines.
- need to transpose data on the CM-2 from parallel to serial
- need to pack/unpack data on the Y-MP
- written in C with sockets

MIMD, multicomputer: heterogenous distributed environments

- distributed code takes longer than just running on CM-2 alone
- for large problem sizes (> 500,000 particles) less than a factor of two slower
- data conversion is significant overhead factor, taking 25% – 33% of the total time for large 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

Decomposition and Assignment

- Determine how work is divided up among collaborating processes
  - structural approaches
    - parallel loops or loop nests
  - application domain decomposition
  - data decomposition
  - task queues
- Usually adopt the view: process $\leftrightarrow$ processor
  - multiprogramming?
  - dynamic resources?

Algorithm and Architecture

The key to performance is to understand the algorithm and the architecture.

A significant improvement in performance can be obtained by matching algorithm to the architecture or vice versa.

A cost-effective way of providing a given level of performance.

Some URLs for Manufactures

- **Major one**
  http://www.netlib.org:80/nse/catalog.html
- **Cray Research**
  http://www.cray.com:80/PUBLIC/product-info/T90/T932.html
  http://www.cray.com/PUBLIC/product-info/J90/J916.html
- **IBM**
  http://lscftp.kgn.ibm.com/pps/products/sp2.html
- **Intel**
  http://www.ssd.intel.com/paragon.html#system
- **Meiko**
  http://www.meiko.com/info/TechnicalDescription/TechnicalDescription.html
- **NEC**
  http://www.nec.co.jp/english/product/computer/sx/index.html
- **SGI**
  http://www.sgi.com/Products/hardware/Power/index.html
- **Tera Computer**
  http://www.tera.com/tera/beyond.html