

## Parallel Computing Survey

- Basic Definitions
- Illustrative Examples
- Computer models
- Speedup
- Engineering principles
- Multiple processors
- Parallel programming models
- Evolution of Performances
- References:
  - PP Chapter 1,2,3
  - <http://www.top500.org>

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

1

## Parallel Computing

- Parallel computer + parallel program = parallel computing
- A parallel computer is a set of processors that are able to work cooperatively to solve a common computational problem.
  - parallel supercomputers with hundreds or thousands of processors,
  - networks of workstations,
  - multiple-processor workstations, and
  - embedded systems.
- A parallel program is a code being able to be executed by parallel computers.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

2

## Parallel Terminology

- There are many different expressions used in the literature for similar principles.
- Processor, Computer, Node, etc.
- Parallel execution, Concurrent execution, Distributed execution
- Multicomputer, Multiprocessor, Parallel processor, Distributed processor
- Communication channel, Link,
- We will use parallel computer when communication by messages is used and parallel processor if shared memory is used for communication.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

3

## Illustrative Examp.1-Writing Projects

- Writing projects (some communication in the first phase, no communication during work - all writers are independent, and no communication on grading).



- Finally, several projects have been done in the time needed to finish a single project plus some overhead because of initial personalised communication.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

4

## Illustrative Examp.2 - Rowing a Boat

- Synchronisation -> helmsman & oarsmen,
- Communication -> oarsman- oarsman, On-line corrections -> oarsmen and helmsman



- The boat is moving faster if the crew is larger but not proportionally and only for some extent. Too much people could even sink the boat.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

5

## Fundamental Problems to be Solved

- Problem decomposition,
- Functional and data dependencies,
- Synchronisation, Communication,
- Partitioning,
- Visualisation of results,
- Performance measurements.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

6

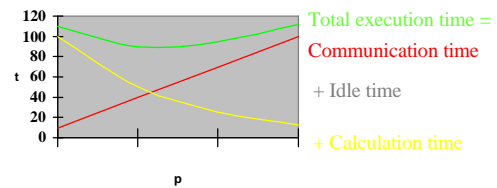
## Parallel Applications

- Numerical simulations of complex systems such as weather, climate, mechanical devices, electronic circuits,
- Control of manufacturing processes, and chemical reactions,
- Commercial applications for large amounts of data,
- Video servers (real-time video).
- The need for faster computers is driven by the demands of both data-intensive applications in commerce and computation-intensive applications in science and engineering.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

7

## Computation + Communication

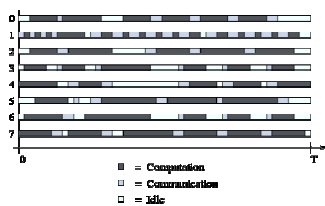


- execution time =  
 computation + communication + idle time
- some optimal number of processors  $p$  exist for a given problem size  $N$ ,
- increase processor performances (technology),
- shorten communication and idle time (algorithm).

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

8

## Computation + Communication (cont.)



Activity plot during execution of a parallel program on eight processors. Each processor spends its time computing, communicating, or idling.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

9

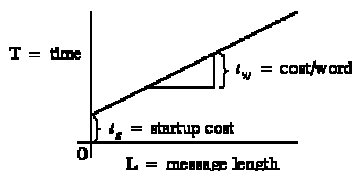
## Overlapping Computation with Communication

- A worker that has sent a request to the input task waits for the parameters to arrive.
- In some cases, efficiency can be improved by pre-fetching, that is, requesting the next parameter before it is needed.
- The worker can then perform computation while its request is being processed by the input task.
- A part of Computation and Communication can be executed in parallel - overlapped in time.
- In such a way idle time can be reduced.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

10

## Communication time - Model

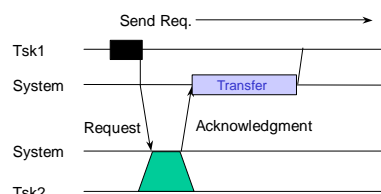


-Time  $T$  versus message length  $L$  is given, the slope of the line corresponds to the cost per word transferred  $t_w$  ( $1/t_w$  is bandwidth).  $T(0)$  is the message startup cost  $t_s$ .  
 -This idealised model of communication performance is adequate for many purposes but does break down in some situations (congestion of communication channels).

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

11

## Communication time (exm.)

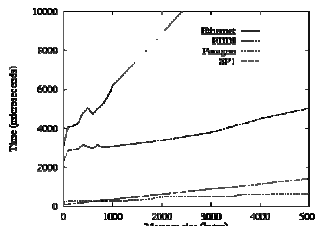


- After a send request a fixed period is spend to service ( $t_s$ ). Then data start to transfer ( $t_w$ ).

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

12

## Communication time - Reality



Round-trip time for a single message between two processors as a function of message length on Ethernet-connected workstations, FDDI-connected workstations, Intel Paragon, and

IBM SP1.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

13

## Speedup and Efficiency

- Execution time of a program on a single processor is  $T_1$ .
- Execution time of the same parallelized program on  $p$  (possibly equally loaded) processors is  $T_p$ .
- $T_p$  is composed of probably shorter execution time, some communication time, and some idle time.
- Speedup  $S = T_1/T_p > 1$ ,
- Efficiency  $E = S/p < 1$ .
- Ideally,  $S=p$  and  $E=1$ , typically, speedup increases slower than  $p$  and efficiency decreases.
- Isoefficiency function tells us how the amount of computation must scale with  $P$  to keep  $E$  constant. Algorithm with isoeff. fun. of  $O(P)$  is highly scalable.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

14

## Performance Analysis

- We estimate the total computation time for a calculation domain of dimension  $f(N)$  by analysing the code and the number of floating point operations.  

$$T_{\text{comp}} = t_c \cdot f(N)$$
- The communication time depends on the number of messages  $M(N,P)$  per calculation step and on the amount of data  $D(N,P)$  to be transferred. Clearly both parameters depend on the number of processors.  

$$T_{\text{comm}} = P \cdot M(N,P) \cdot D(N,P)$$
- If idle time is 0, because of well balanced computation the execution time of a parallelized program is:  

$$T_p = (T_{\text{comp}} + T_{\text{comm}}) / P$$

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

15

## How to increase computer performances ?

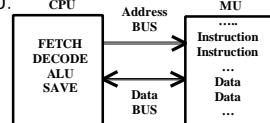
- By increasing the data parallelism in a processor (parallel busses, etc.)
- By using multiple functional units and pipelining,
- By using vectorized processors able to operate on several vectorized items at the same time,
- By using multiple computing and memory units,
- By increasing the system clock.
- ???

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

16

## Single Computer Model (Von Neuman)

- Control unit (CU) - fetch and decode instructions,
- Processing Unit (PU) - execute instructions,
- CU + PU = CPU,
- Random Access Memory (RAM) Unit (MU) - save data and instructions,
- CPU read instructions and data from MU and write results back to the MU.

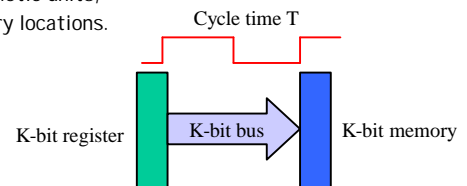


University of Salzburg, Department of Scientific Computing, HPSC SS 2003

17

## Data Parallelism

- Parallel bits  $\Rightarrow$  bytes  $\Rightarrow$  words  $\Rightarrow$  etc.
- Registers, data buses, address busses,
- Arithmetic units,
- Memory locations.



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

18

## Physical Limits

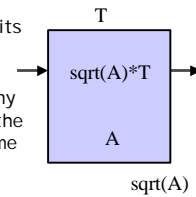
- Speed of light and electromagnetic field  
 $v = 3 \cdot 10^8 \text{ m/s}$ ,
- $f = 10^9 \text{ Hz} = 1 \text{ GHz} \Rightarrow T = 10^{-9} \text{ s} = 1 \text{ ns} \Rightarrow \text{Distance}$   
 $d = v \cdot T = 3 \cdot 10^8 \text{ [m/s]} \cdot 10^{-9} \text{ [s]} = 30 \text{ cm}$ ,
- For 100 GHz we get the distance of 3mm,
- Two objects at the distance of 3mm, with an ideal optical communication channel can not communicate faster than by 100GHz synchronous clock (clock delays should be in accordance with data delays).
- We can put the clock source in the middle of both objects, but the problem might appear for other objects...

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

19

## Potential Problem - $AT^2$ Complexity(\*)

- internal concurrency in a chip, operating simultaneously on 64 bits of two numbers that are to be multiplied,
- transitive computation in which any output may depend on any input, the chip area  $A$  and the computing time  $T$  are related as  $AT^2 = \text{const.}$
- on an area of  $n^2 A$  we may build a single component that needs time  $T/n$ , or, use  $n^2$  parallel components each on area  $A$  with the time  $T/n^2$ .
- Parallel  $n^2$ -component system can be  $n$ -times faster.



$$AT^2 = (n^2 A)(T/n)^2$$

$$n^2(A/(T/n^2)^2)$$

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

20

## Pipelining

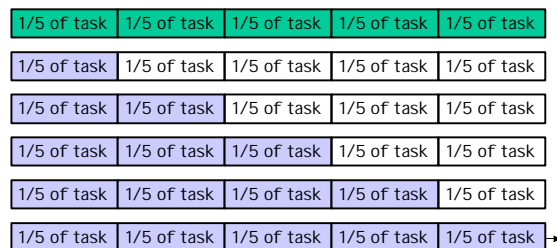
- Suppose that for finishing a complex repetitive task (e.g., car production) time  $T$  is needed for one person.
- If the task can be divided into  $k$  equally time-demanding subtasks,  $k$  workers can work in parallel, each its own subtask.
- The resources needed for the implementation of our task can be exploited totally with  $k$  workers.
- The first car will come from the pipe in time  $T$ , but the second car will be out already after  $T + T/k$ , and third after an additional delay of  $T/k$ .

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

21

## Pipelining (cont.)

Time  $T$  needed for a single person to complete the task



- What will happen if subtasks have diff. time complexity?

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

22

## Multiple Functional Units

- A central processing unit CPU has a task to execute instructions,
- Subtasks: Fetch from memory, decode, calculate, save results,
- More arithmetic units with Adders, Dividers, etc.,
- Pipelining can be implemented in these functional units,
- Several functional units can be present to be able to execute several operations in the same time slot.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

23

## Vector Computers

- Sophisticated Von Neuman design.
- More identical functional units (VPU, IP/ALU, FPU, I/OP) on the same bus that connect memory (MU)
- Vector data manipulated in parallel; vector registers; no cache memory; mask registers to enable operation on a selected subset of elements.
- Bottleneck - the bandwidth between memory and the VPU. For example, the vector operation  $A=B+C$ , where vector operands are represented by multiple (i.e.:64) data items, can be performed in a single instruction time, with  $2 \times 64$  memory accesses.
- Japan is the leading manufacturer of Vector Processors.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

24

## Multiple Computing and Memory Units

- On the highest level we may use more single Von Neuman computers (nodes), able to communicate by an interconnection network.
- Communication channels for messages
  - distributed memory computer - parallel computer,
  - Message Passing Processor (MPP),
- Common data locations in RAM, accessible by parallel buses
  - shared memory computer - multiprocessor,
  - Shared Memory Processor (SMP).

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

25

## Flyn's Classification

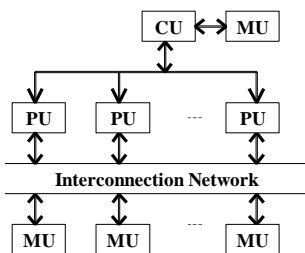
- SISD (Single Instruction Single Data) - Von Neuman computer
- SIMD (Single Instruction Multiple Data) - Distributed and shared memory computers (i.e.: Connection Machine, vector processors).
- MISD (Multiple Instruction Single Data) - no existing prototype??
- MIMD (Multiple Instruction Multiple Data) - Distributed and shared memory computers (i.e.: Network of processors, Computer clusters, Symmetric processors, Parallel Random Access Machine (PRAM)-theoretical model)

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

26

## SIMD - processor-arrays

- distributed memory  
SIMD (single CU multiple PU and MU); no synchronisation, fine-grain parallelism, specialised
- vector computers can be seen as shared memory SIMD (single CU, multiple PU and single MU); more general

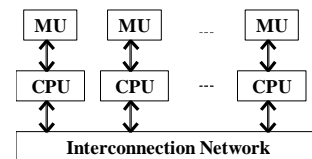


University of Salzburg, Department of Scientific Computing, HPSC SS 2003

27

## MIMD - multiple CU, PU and MU

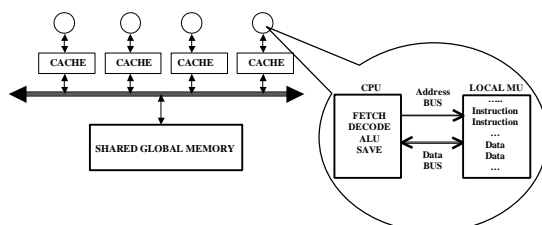
- MIMD (multiple CU, PU and MU & interconnection network)
- coarse-grain parallelism
- fastest growing family of high-performance computers.



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

28

## Shared Memory MIMD Computer-tightly coupled



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

29

## Shared Memory MIMD Computer-tightly coupled (cont.)

- can execute multiple tasks in the same time
- communication and synchronisation through the shared memory locations
- interconnection network should enable ideally a full connected system (one-to-one, one-to-all, all-to-all)
  - bus  $O(N)$
  - multi-stage dynamic interconnection networks  $O(N \log N)$ ,
  - cross-bar  $O(N^2)$
- memory should be extremely fast to enable effective global communication and to follow the processors speed.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

30

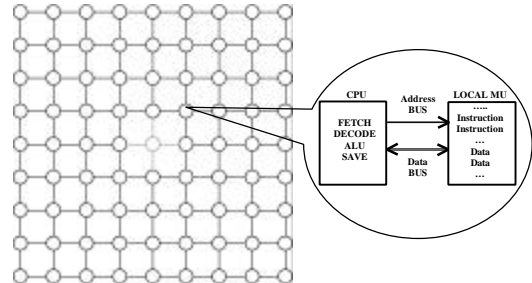
## Properties of SM-MIMD

- Shared Memory Processors (SMP)
- Advantages
  - no need for explicit communication mechanisms
  - the data distribution is completely transparent to the user
  - simpler and more compact program code
- Disadvantages
  - the connection of the CPUs to each other and to the memory (caches can alleviate the bandwidth problem)
  - speed of the memory
  - in practice the number of processors limited to 10-20

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

31

## Distributed Memory MIMD Computer- loosely coupled



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

32

## Distributed Memory MIMD Computer- loosely coupled (cont.)

- can execute multiple tasks in the same time
- data communication and synchronisation through messages
- interconnection network should enable routing of messages
- communication hardware (routers, queues) should work in parallel with the node processor
- local memory should be balanced with a number of directly connected processors and the speed of a node processor

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

33

## Properties of DM-MIMD Message Passing Processors (MPP)

- Advantages
  - communication bandwidth scales up automatically with the number of processors
  - no need for very fast memory
  - unlimited number of processors
- Disadvantages
  - global communication is much slower (synchronisation slower too)
  - user has to distribute the data over the processors
  - data exchange between processors has to be performed explicitly
  - more complex program code (actual speed-up << theoretical speed-up)

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

34

## Network of Processors

- If the network part represent the foundation of the system, that connect usually heterogeneous computers, we talk about Network of Workstations (NOW).
- Internet computing (millions of connected computers), Grid Computing,
- Cluster Of Workstations (COW).
- Constellations (clusters of SMP or MPP)
- Distributed computing (reliability, security, heterogeneity etc.),

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

35

## Performance Measures

- Floating Point Operation is an arithmetic operation on 64 bit wide data.
- Number of Floating Point Operations per second (R): Mflop/s =  $10^6$ , Gflop/s =  $10^9$ , Tflop/s =  $10^{12}$
- Clock of the Central Processing Unit (MHz =  $10^6$ /s, GHz =  $10^9$ /s),
- Memory (Cache, Bulk, Back-up) (Mbit=Mb, Mbyte=MB, Mword=Mw).
- A problem has order of magnitude  $f(N)=O(g(N))$  if there is a positive constant  $c$  such that  $|f(N)| \leq c \cdot |g(N)|$  (e.g., for  $2/3 \cdot N^2 + N$  operations the order of magnitude is  $O(N^2)$ , because terms of lower order than  $N^2$  become insignificant if  $N$  becomes large.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

36

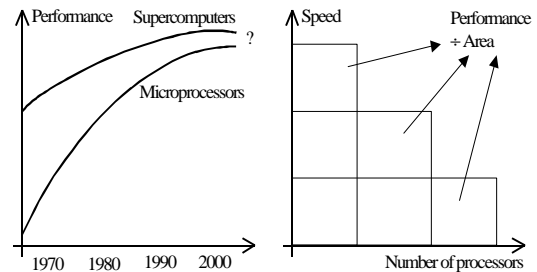
## LINPACK Benchmark

- The solution problem used in the LINPACK Benchmark is to solve a dense system of linear equations (e.g.: 1000 equations).
- The algorithm used in solving the system of equations in the benchmark procedure is LU factorisation with partial pivoting with an approximate number of flops:  $\frac{2}{3}n^3 + O(n^2)$
- By measuring the actual performance for different problem sizes  $n$ , a user can get:
  - Rmax - maximal LINPACK performance achieved (in MFLOPS)
  - Nmax - problemsize for achieving Rmax
  - Rpeak - Theoretical peak performance (in MFLOPS)

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

37

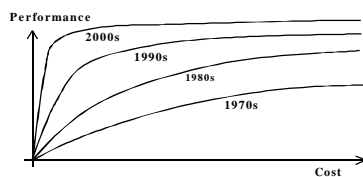
## Improving Parallel Performance



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

38

## Cost Versus Performance

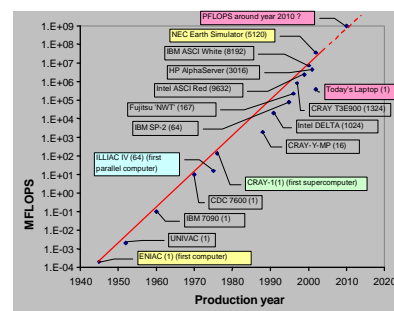


- At the lower end performance increases linearly with cost (ordinary market).
- Beyond a certain point, curves start to saturate (technological limits).
- The transition point become sharper with time (advances in technology & parallel computers).

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

39

## Peak Performance 1950-2010



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

40

## Moore's Law

- We can learn from the previous graphs that the computer performance increased exponentially.
- From  $10^4$  in 1955 to  $10^{11}$  in 1995 (factor of  $10^7$  in 40 years)
- From  $10^{11}$  in 1995 to  $7.2 \cdot 10^{12}$  in 2002 (factor of 60 in 7 years).
- $(R_2 - R_1) = 2^{(t_2 - t_1)/T_d}$  ( $T_d$  = time for double performance;  $\log_2 = 0.3$ )
- $T_d = (40/7) \cdot \log_2 = 1.7$   $T_d = (7/\log(60)) \cdot \log_2 = 1.2$
- Moore's law states that the computer performance doubles each one and half year.
- The performance of storage media is for an order of magnitude less intensive.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

41

## Top10 from Top500

Rank	System Name	Processor	Peak Performance (MFLOPS)	System Name	Processor	Peak Performance (MFLOPS)
1	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
2	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
3	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
4	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
5	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
6	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
7	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
8	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
9	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000
10	IBM SP-LC	IBM SP-LC	1.000.000	IBM SP-LC	IBM SP-LC	1.000.000

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

42

## Highlights from the Top10

- **Earth Simulator Center, Japan 2002, NEC is #1 with 35 TF/s on the Linpack**
- Accelerated Strategic Computing Initiative - ASCI Q is #2 with 7 TF/s on the Linpack.
- 47 systems have Linpack performance above 1 Tflop/s.
- 3 Tflop/s is the entry point for the Top 10.
- Smallest number of processor in Top500 is 16.
- All vector based systems are of Japanese origin.
- Leading manufactures: NEC, Hewlett-Packard, IBM, SGI, Cray Inc., Sun.

University of Salzburg, Department of Scientific Computing, HPSC SS 2003

43

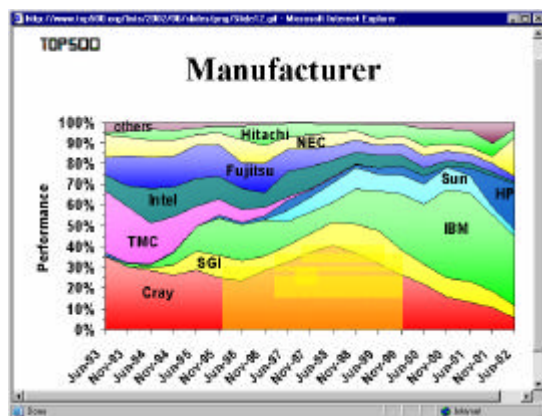
## Austria on TOP500 - November 2001/02

Nov./01	356	IBM	SP Power3	112.00	Banco Vitalico	Austria	2001	Industry	108
Nov./01	402	IBM	SP Power3	100.00	Telecom	Austria	2001	Industry	96
			375 MHz						

Rank	System	Processor	Speed (Mflop/s)	Configuration	Organization	Country	Year	Category	Notes
1	NEC SX6R	NEC SX6R	35.00	128000	Earth Simulator Center	Japan	2002	Industry	
2	ASCI Q	Intel Pentium III	7.00	128000	Advanced Strategic Computing Initiative	USA	2001	Industry	
3	NEC SX6R	NEC SX6R	6.00	128000	Earth Simulator Center	Japan	2002	Industry	
4	NEC SX6R	NEC SX6R	5.00	128000	Earth Simulator Center	Japan	2002	Industry	
5	NEC SX6R	NEC SX6R	4.00	128000	Earth Simulator Center	Japan	2002	Industry	
6	NEC SX6R	NEC SX6R	3.00	128000	Earth Simulator Center	Japan	2002	Industry	
7	NEC SX6R	NEC SX6R	2.00	128000	Earth Simulator Center	Japan	2002	Industry	
8	NEC SX6R	NEC SX6R	1.00	128000	Earth Simulator Center	Japan	2002	Industry	
9	NEC SX6R	NEC SX6R	0.50	128000	Earth Simulator Center	Japan	2002	Industry	
10	NEC SX6R	NEC SX6R	0.25	128000	Earth Simulator Center	Japan	2002	Industry	

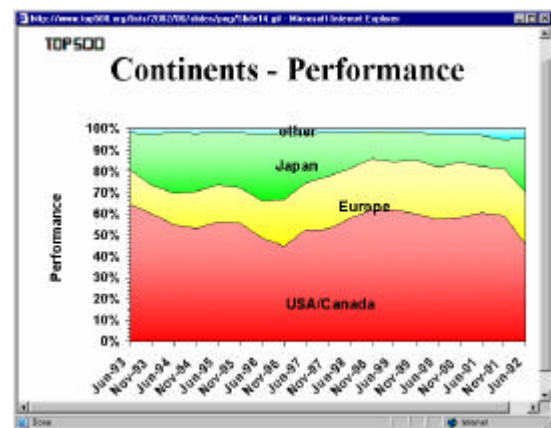
University of Salzburg, Department of Scientific Computing, HPSC SS 2003

44



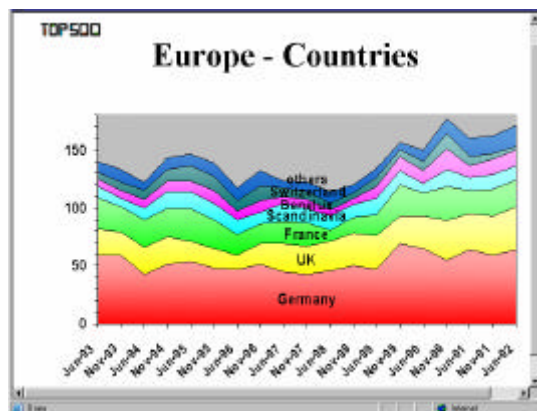
University of Salzburg, Department of Scientific Computing, HPSC SS 2003

45



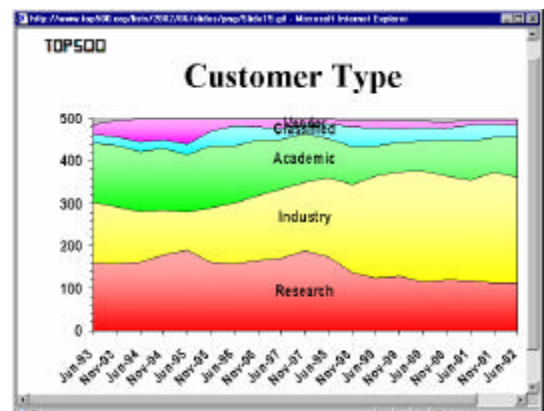
University of Salzburg, Department of Scientific Computing, HPSC SS 2003

46



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

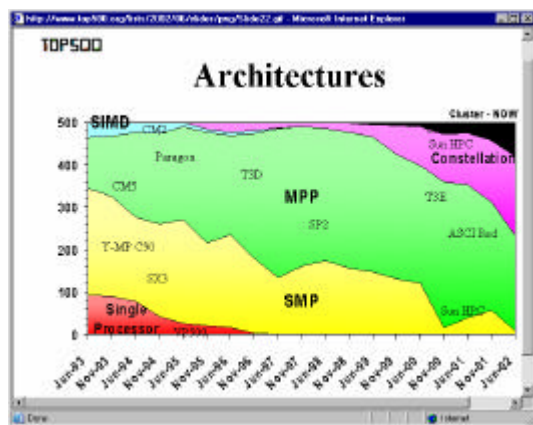
47



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

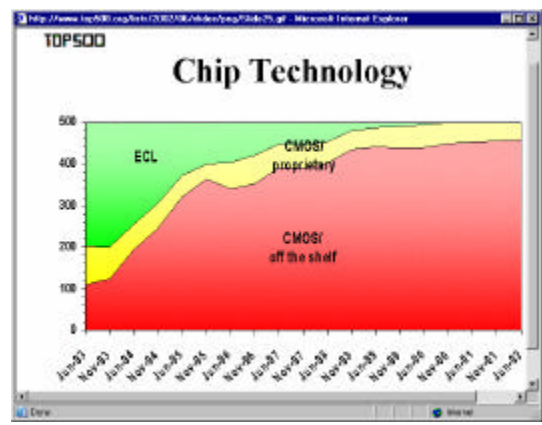
48





University of Salzburg, Department of Scientific Computing, HPSC SS 2003

49



University of Salzburg, Department of Scientific Computing, HPSC SS 2003

50