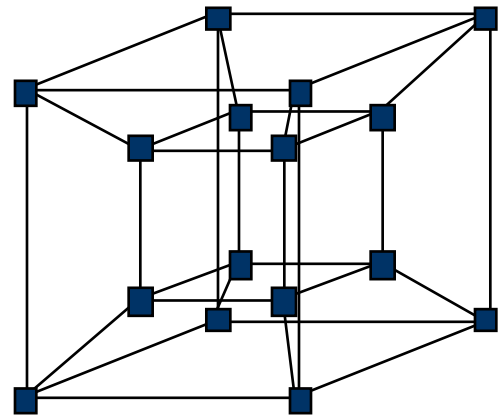


Chapter 2

Parallel Architectures



2.1 Hardware-Architecture

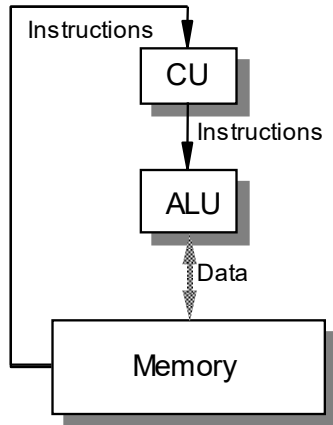
Flynn's Classification

The most well known and simple classification scheme differentiates according to the multiplicity of instruction and data streams.

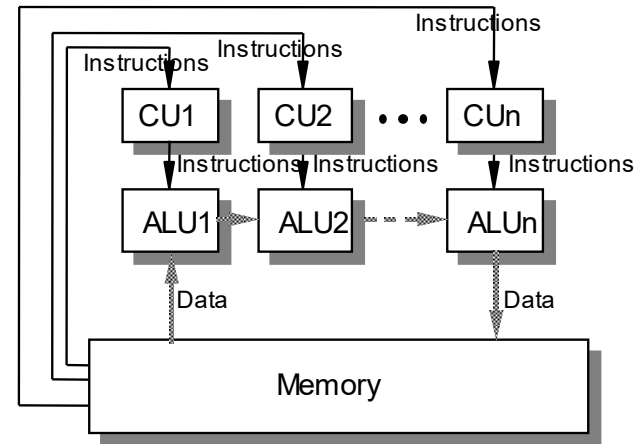
The combination yields four classes:

| | SI (single instruction) | MI (multiple instruction) |
|--------------------|---|---|
| SD (single data) | SISD: Von-Neumann-Computer | MISD: Data flow machines |
| MD (multiple data) | SIMD: Vector computer, (Cray-1, CM2, MasPar, some GPUs) | MIMD: Multiprocessor systems, Distributed Systems (Cray T3E, Cluster, most current computers..) |

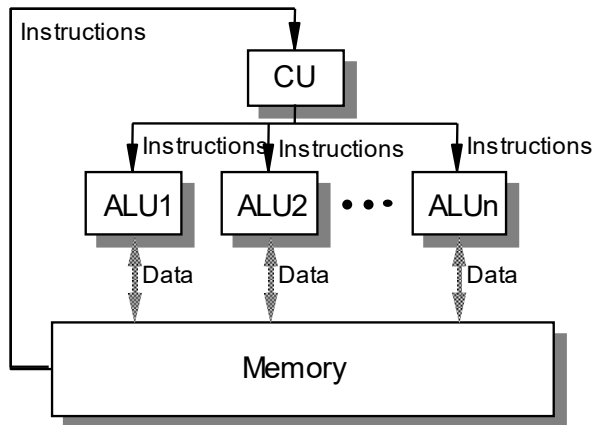
Flynn's Classification scheme



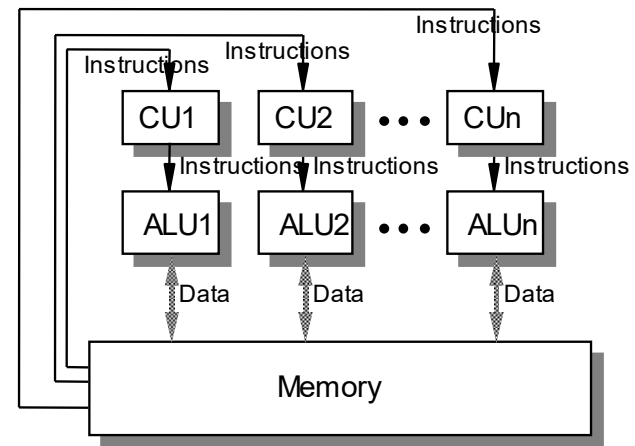
SISD



MISD



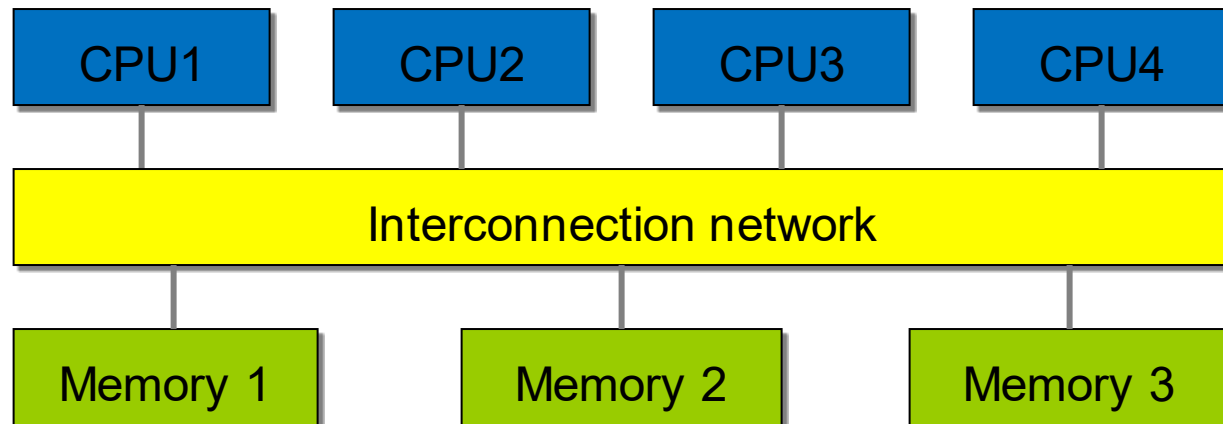
SIMD



MIMD

a) Multiprocessor systems (*shared memory systems, tightly coupled systems*)

All processors have access to common memory modules via a common communication network



Further distinction according to the accessibility of the memory modules:

uniform memory access (UMA):

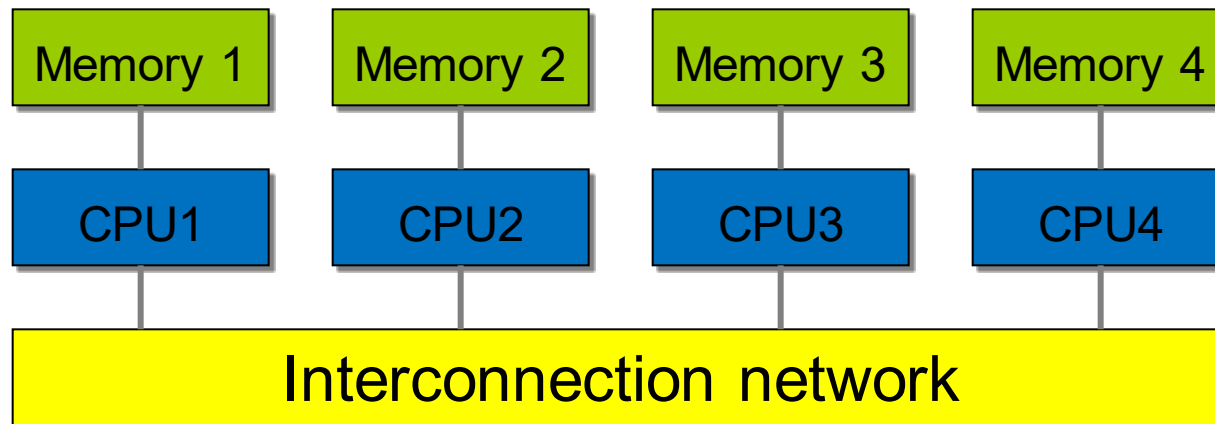
Access to memory module is independent of address of processor and memory module

non-uniform memory access (NUMA):

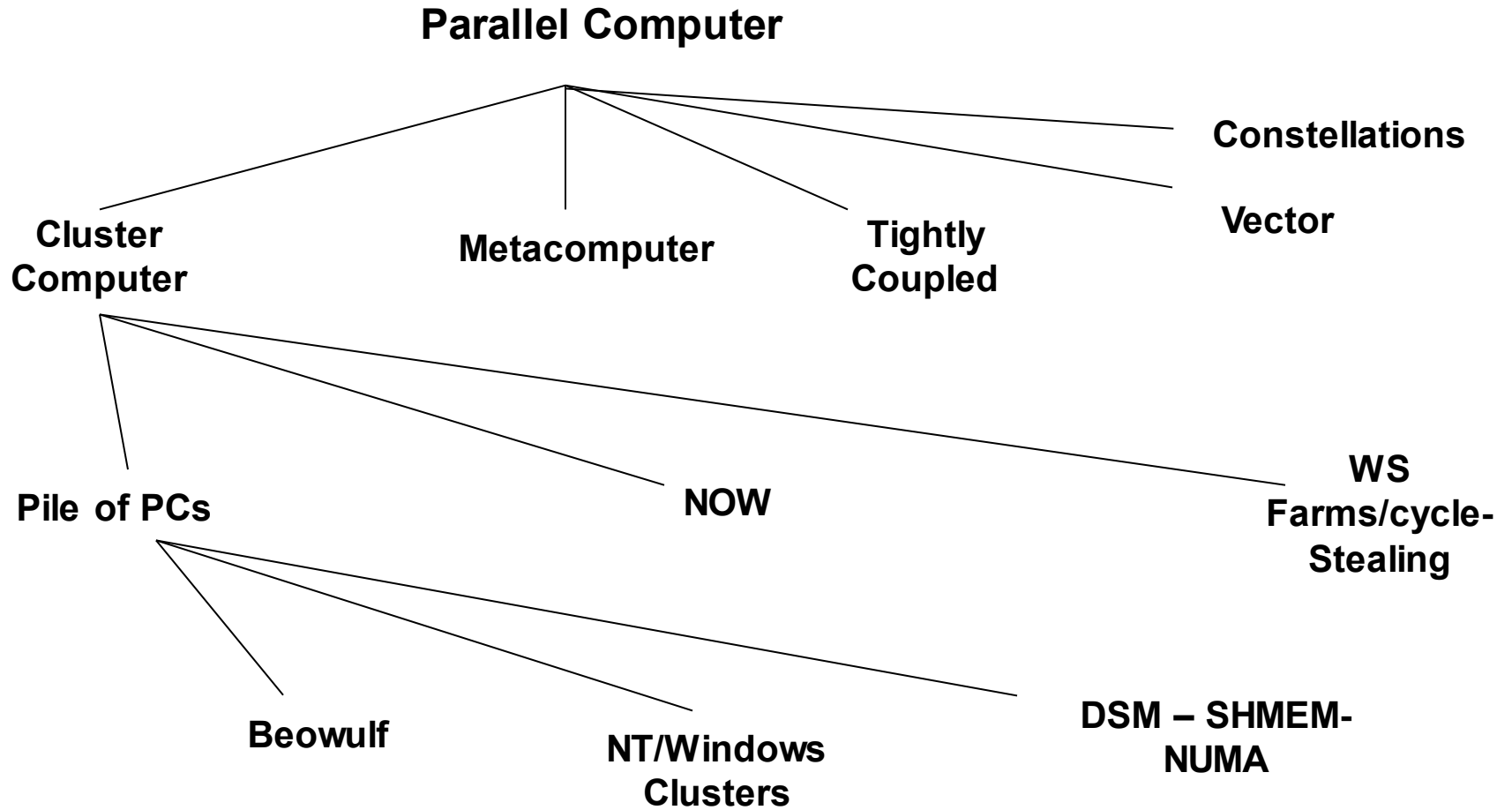
Memory access depends on addresses involved.

b) Multicomputer systems (*message passing systems, loosely coupled systems*)

Each processor has its own private memory with exclusive access
Data exchange takes place by sending messages across an interconnection network.



Pragmatic Classification (Parallel Computer)



2.2 Interconnection networks

General Criteria

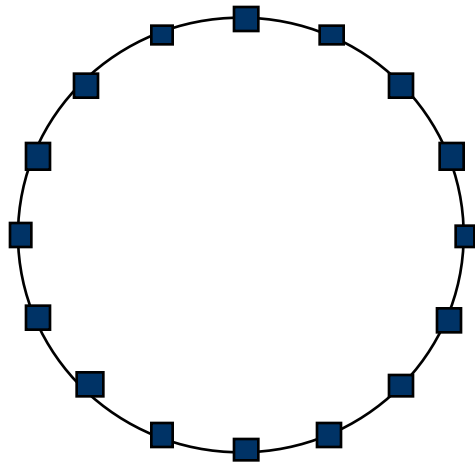
- *Extendibility*
 - Arbitrary increments
- *Performance*
 - Short paths between all processors
 - High bandwidth
 - Short delay
- *Cost*
 - Proportional to number of wires and to number of access points
- *Reliability*
 - Existence of redundant data paths
- *Functionality*
 - Buffering
 - Routing
 - Group communication

Topological Performance Aspects

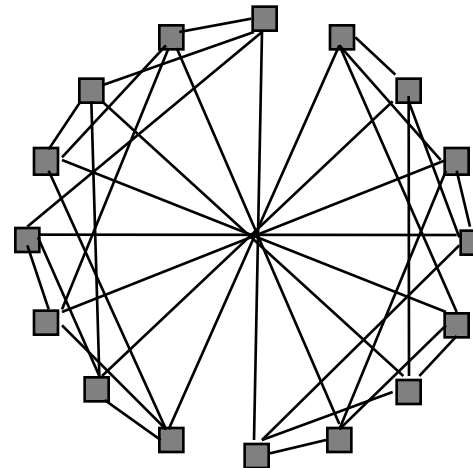
- Node degree
 - Def: Number of directly connected neighbor nodes
 - Goal : constant, small (cost, scalability)
- Diameter
 - Def: Maximum path length between two arbitrary nodes
 - Goal : small (low maximum message delay)
- Edge connectivity
 - Def: Minimum number of edges to be removed in order to partition the network
 - Goal : high (bandwidth, fault tolerance, parallelism)
- Bisection bandwidth
 - Def: Minimum sum of bandwidth of all sets of edges, which partition the network in two equal halves when removed
 - Goal: high (bandwidth, fault tolerance, parallelism)

2.2.1 Static Networks

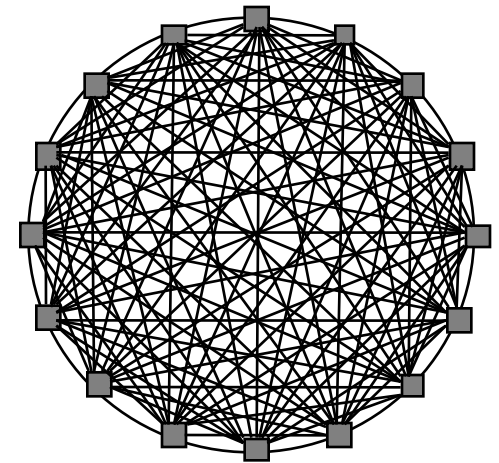
Connectivity spectrum of static networks



Ring



Hypercube



Completely meshed up

No. of links

n

$n/2 \log_2 n$

$n(n-1) / 2$

Accesses/Processors (ND)

2

$\log_2 n$

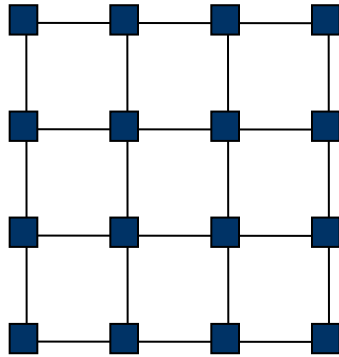
$n-1$

Diameter

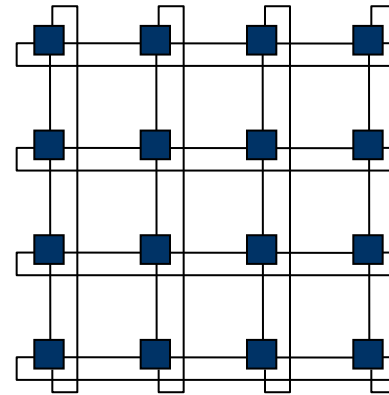
$n/2$

$\log_2 n$

1



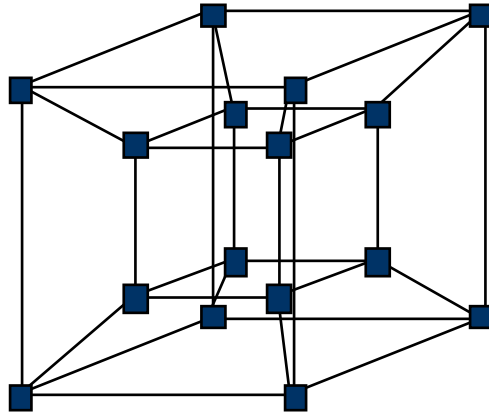
4x4-Grid



4x4-Torus

Properties

- Constant node degree
- Extendibility in small increments
- Good support of algorithms with local communication structure (modeling of physical processes, e.g. LaPlace heat distribution)



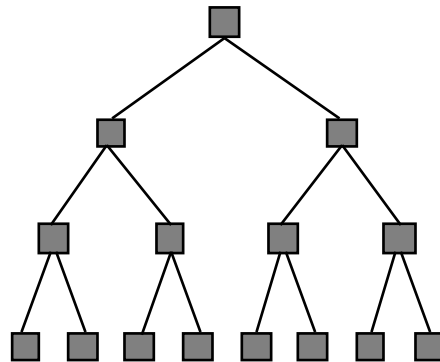
Hypercube of dimension 4

Properties:

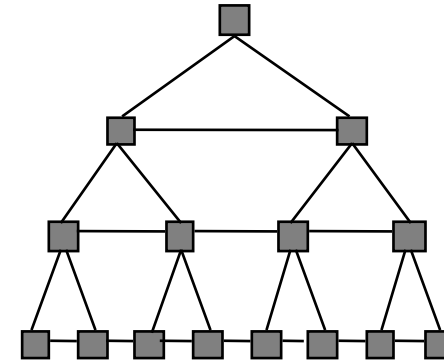
- Logarithmic diameter
- Extendibility in powers of 2
- Variable node degree



Connection Machine CM-2



Binary tree



X-Tree

Properties:

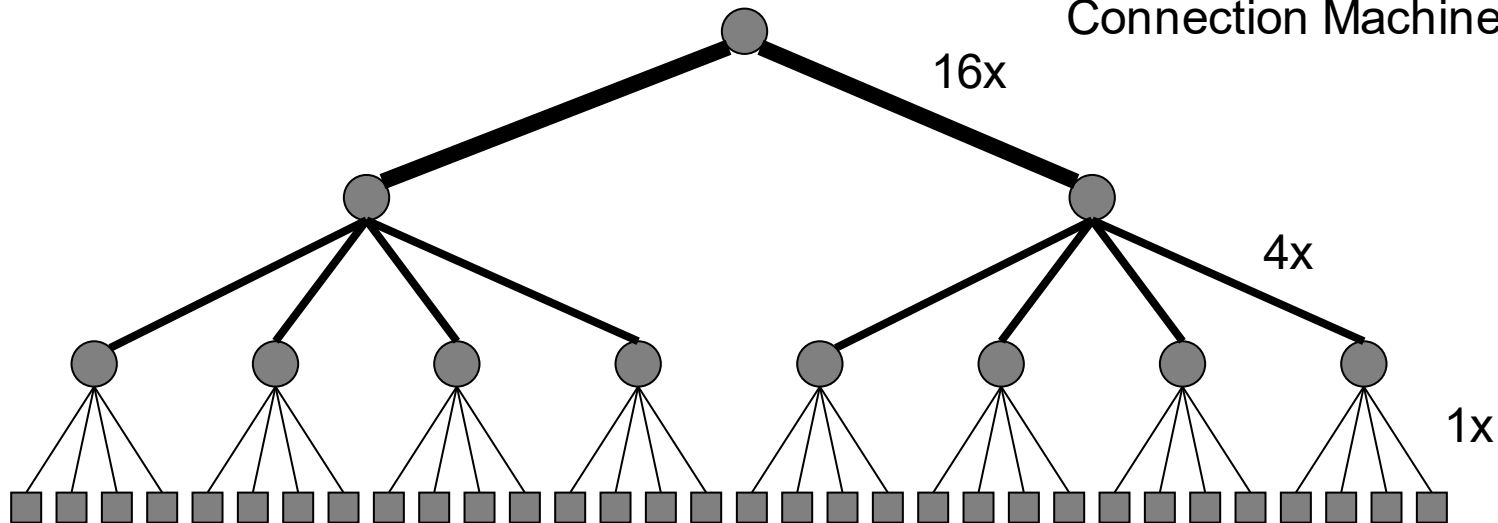
- Logarithmic diameter
- Extendibility in powers of 2
- Node degree at most 3 (at most 5, respectively)
- Poor bisection bandwidth
- No parallel data paths (binary tree)

Fat Tree

n-ary Tree
Higher Capacity (factor n) closer to the root
removes bottleneck

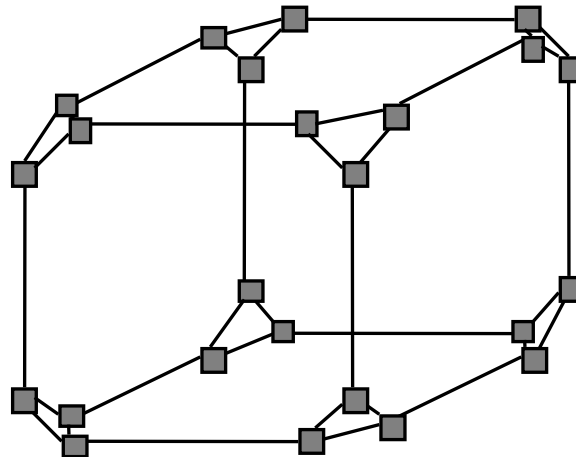


Connection Machine CM-5



d -dimensional Hypercube, where each node is replaced by a ring of size d .

Each of these d nodes has two links in the ring and one more to one of the d hypercube dimensions



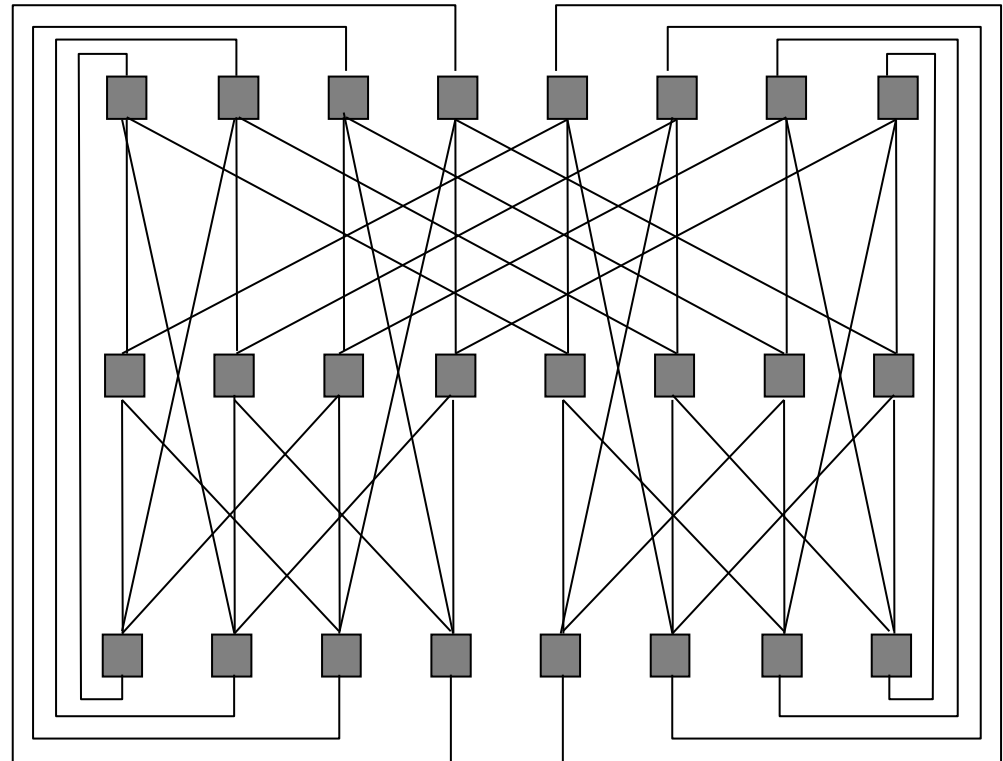
Cube Connected Cycles
of dimension 3
(CCC(3))

Properties:

- Logarithmic diameter
- Constant node degree (3)
- Extendibility only in powers of 2

Properties:

- Logarithmic diameter
- Extendibility in powers of 2
- Constant node degree 4

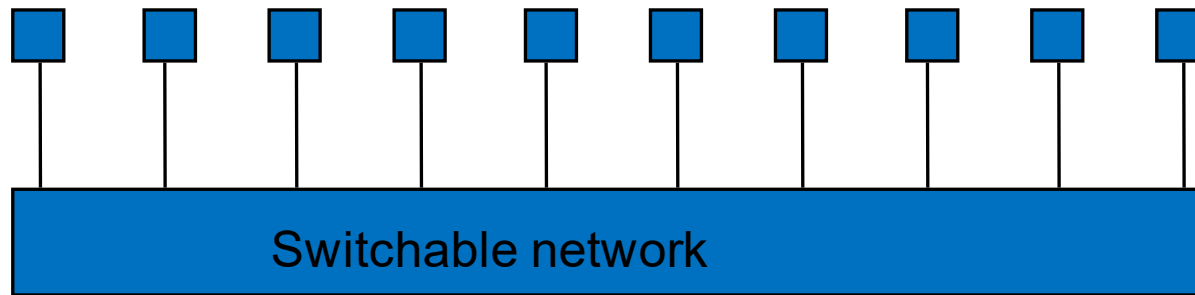


Overview: Properties

| Graph | Number of nodes | Number of edges | Max. Node degree | Diameter |
|---|---------------------|---|------------------|--|
| Grid $G(a_1 \times a_2 \times \dots \times a_d)$ | $\prod_{k=1}^d a_k$ | $\sum_{k=1}^d (a_k - 1) \prod_{i \neq k} a_i$ | $2d$ | $\sum_{k=1}^d (a_k - 1)$ |
| Torus $T(a_1 \times a_2 \times \dots \times a_d)$ | $\prod_{k=1}^d a_k$ | $d \prod_{k=1}^d a_k$ | $2d$ | $\sum_{k=1}^d \lfloor a_k / 2 \rfloor$ |
| Hypercube $H(d)$ | 2^d | $d2^{d-1}$ | d | d |
| Cube Connected Cycles $CCC(d)$ | $d2^d$ | $3d2^{d-1}$ | 3 | $2d + \lfloor d / 2 \rfloor$ |

2.2.2 Dynamic Interconnection networks

All components have access to a joint network. Connections are switched on request.



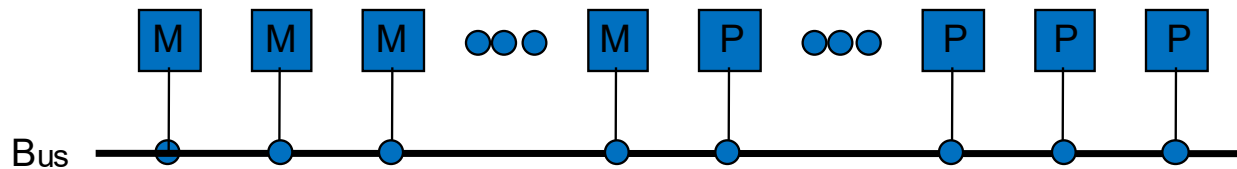
Scheme of a dynamic network

There are basically three different classes

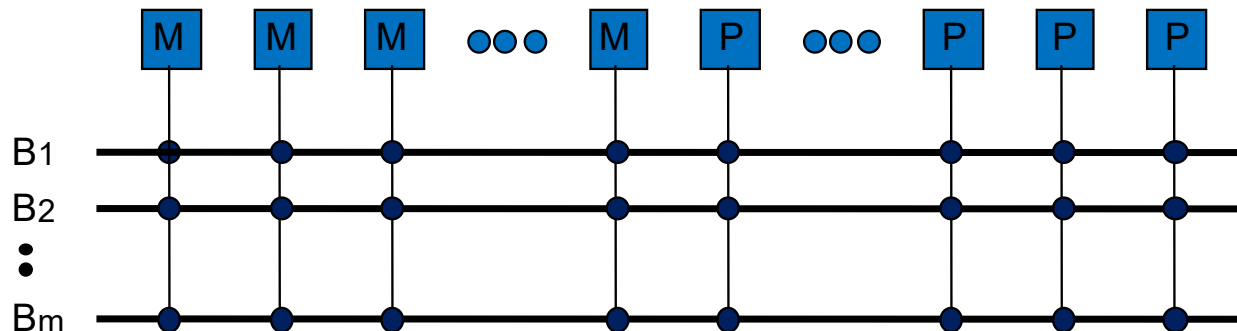
- Bus
- Crossbar switch
- Multistage networks

- cost effective
- blocking
- extendible
- suitable for small number of components

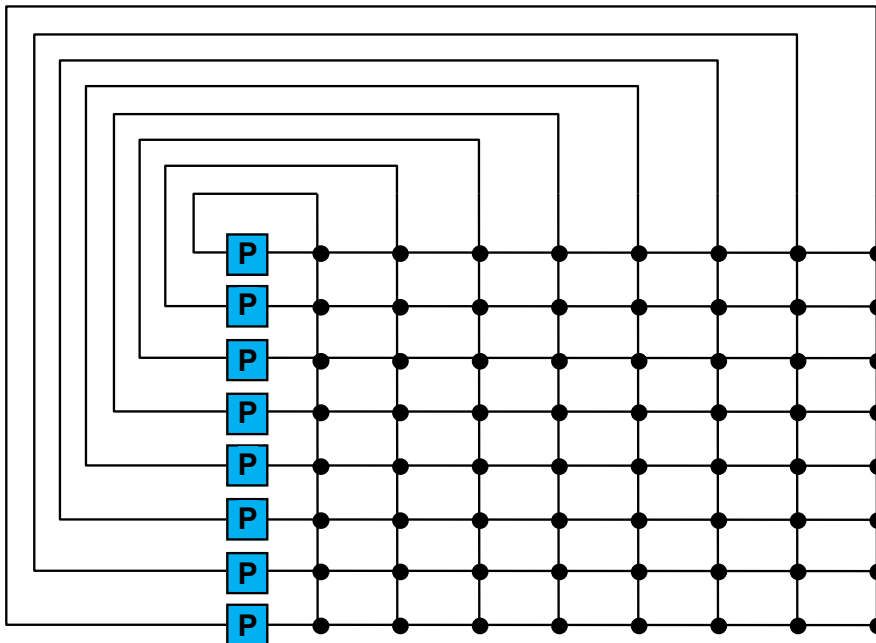
Single bus



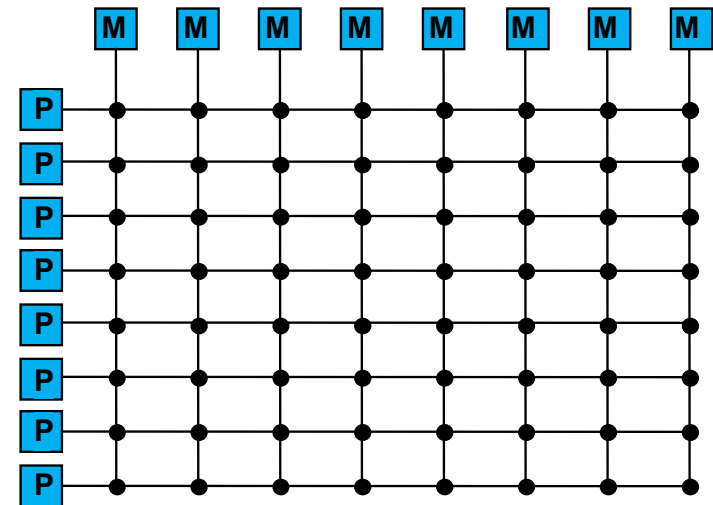
Multibus for multiprocessor architecture



- expensive
- non-blocking, highly performant
- fixed number of access points
- realizable only for small networks due to quadratically growing costs



Interprocessor connection



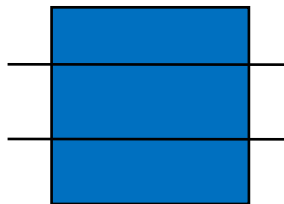
Connection between processors and memory modules

2.2.3 Multistage Networks

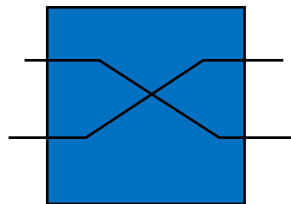
Small crossbar switches (e.g. 2x2) serve as cells that are connected in stages to build larger networks.

Properties

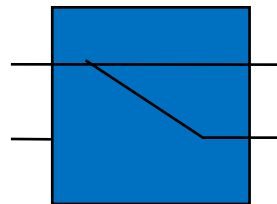
- partially blocking
- extendible



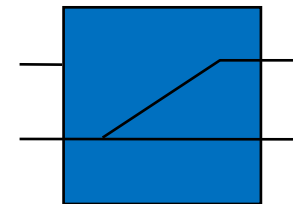
(a)



(b)



(c)

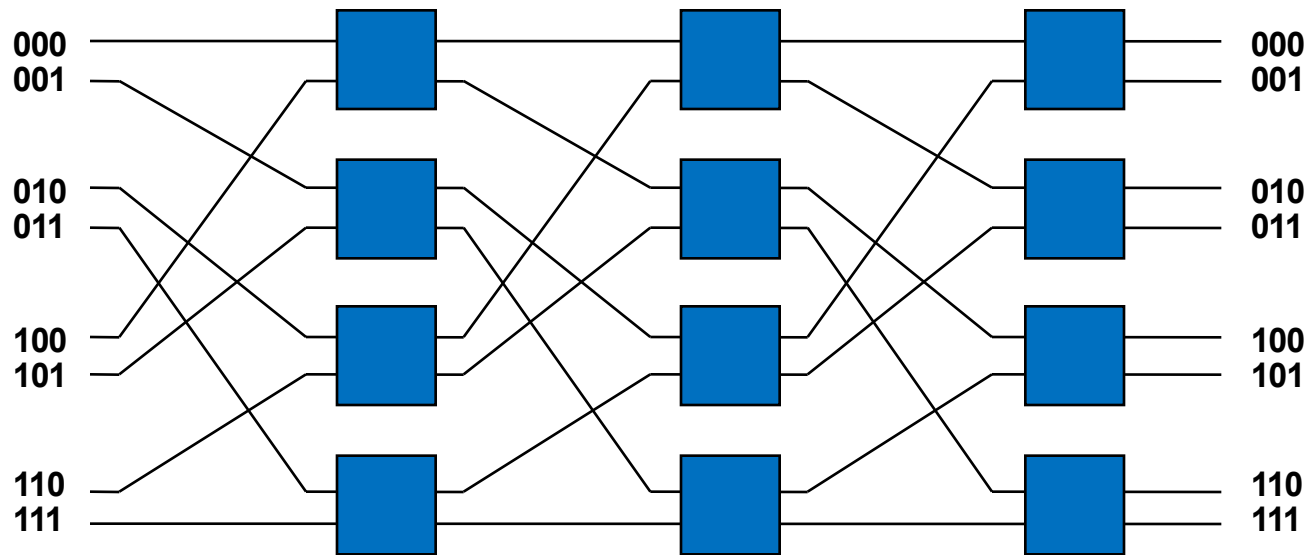


(d)

Elementary switching states:

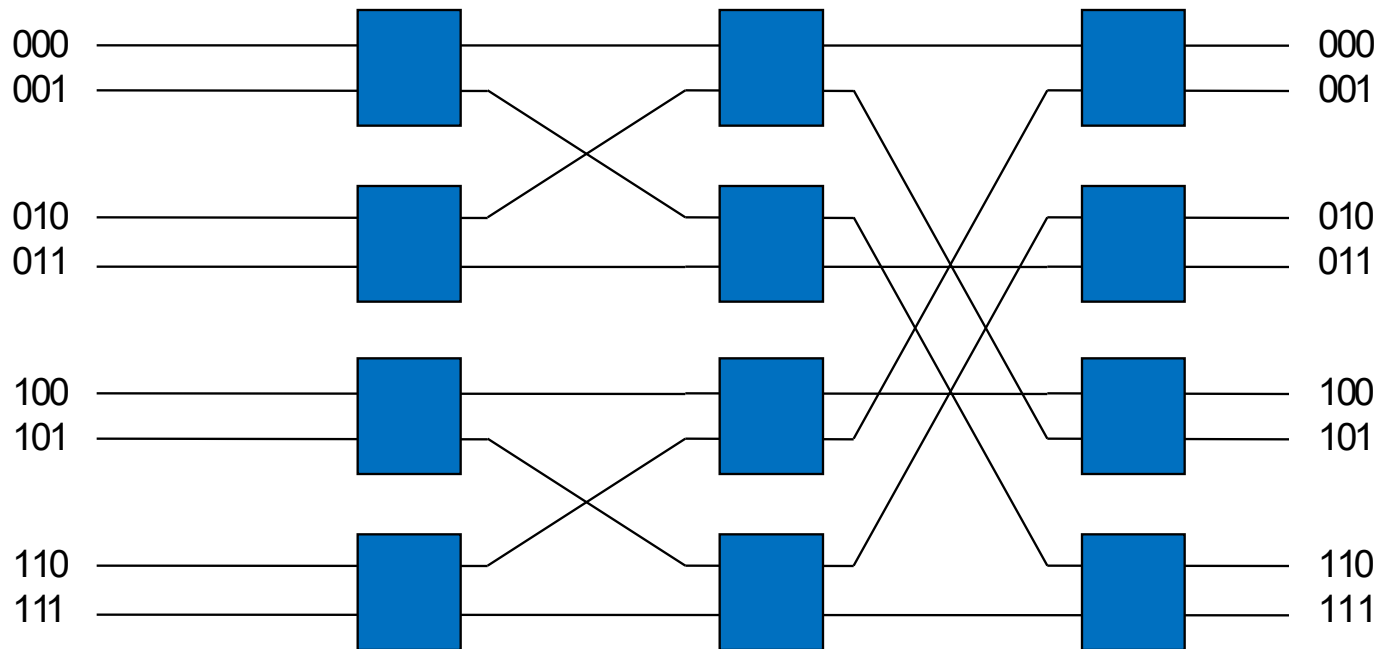
through (a), cross (b), upper (c) and lower (d) broadcast

Example: Omega-Network



Target address determines selection of output for each stage (0: up, 1: down)

Example: Banyan-Network



Classes of dynamic interconnection networks with their basic properties in comparison:

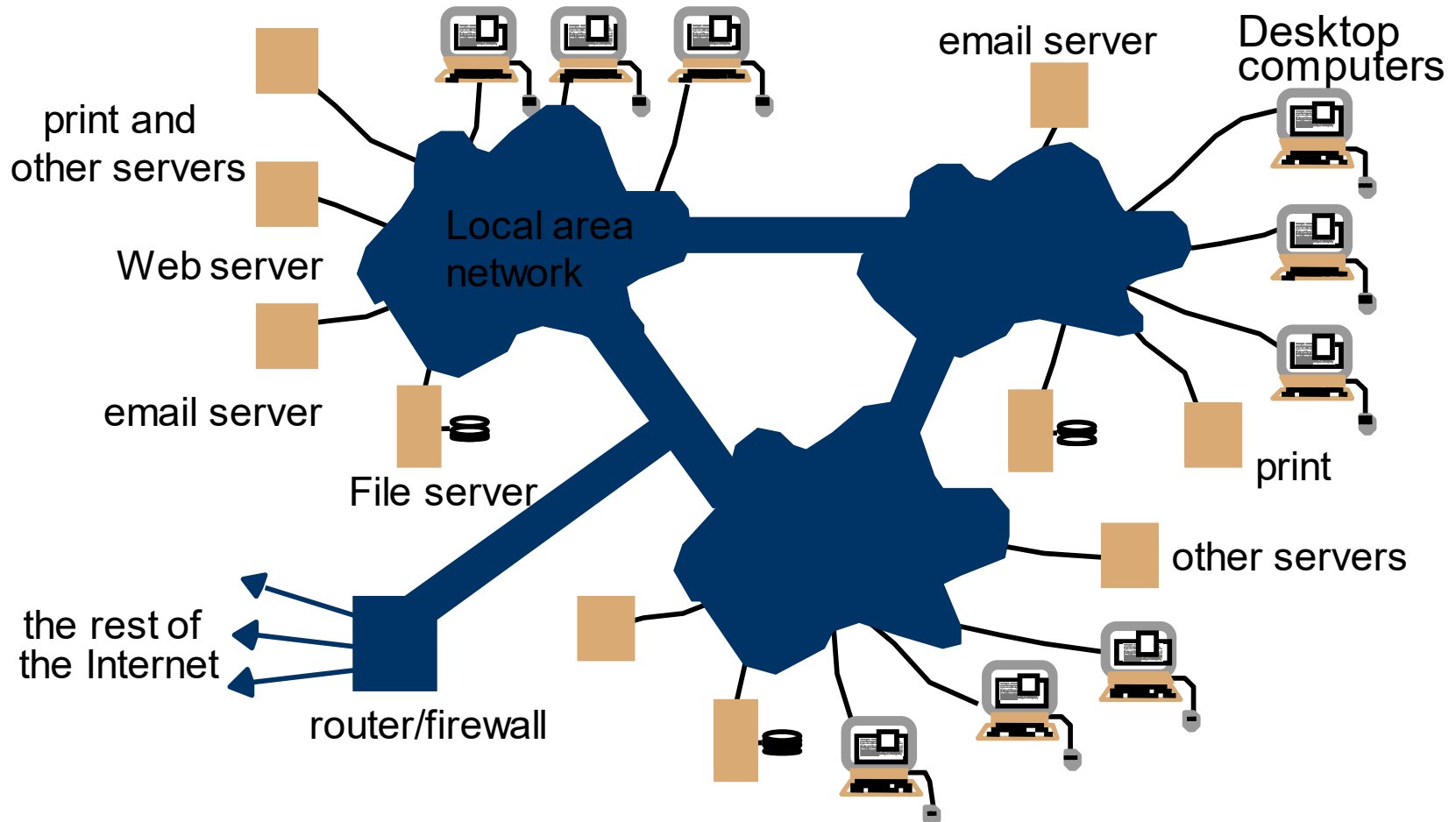
| | Bus | Multistage Network | Crossbar switch |
|----------------------------|-------|--------------------|-----------------|
| Latency (distance) | 1 | $\log n$ | 1 |
| Bandwidth per access point | $1/n$ | < 1 | 1 |
| Switching cost | n | $n \log n$ | n^2 |
| Wiring cost | 1 | $n \log n$ | n |

Asymptotic growth of cost and performance features of dynamic interconnection networks

2.2.4 Local Networks

- mostly bus-like or ring-like topologies
- diverse media (e.g. coaxial cable, twisted pair, optical fiber, infrared, radio)
- diverse protocols (e.g. IEEE 802.3 (Ethernet, Fast-Ethernet, Gigabit-Ethernet), IEEE 802.5 (Token-Ring), FDDI, ATM, IEEE 802.11 (Wlan/WiFi), IEEE 802.15 (Bluetooth), IEEE 802.16 (WiMAX))
- compute nodes typically heterogeneous regarding
 - performance
 - manufacturer
 - operating system
- mostly hierarchical heterogeneous structure of subnetworks (structured networks)

Typical Intranet



2.3 Cluster Architectures

A Cluster is the collection of usually complete autonomous computers (Workstations, PCs) to build a parallel computer. (Some cluster nodes share infrastructure such as power supply, case, fans).

The component computer nodes can but need not to be spatially tightly coupled.

The coupling is realized by a high speed network.

Networks (typically) used:

- Ethernet (Fast, Gigabit, 10 Gigabit, ...)
- Myrinet
- SCI (Scalable Coherent Interface)
- Infiniband

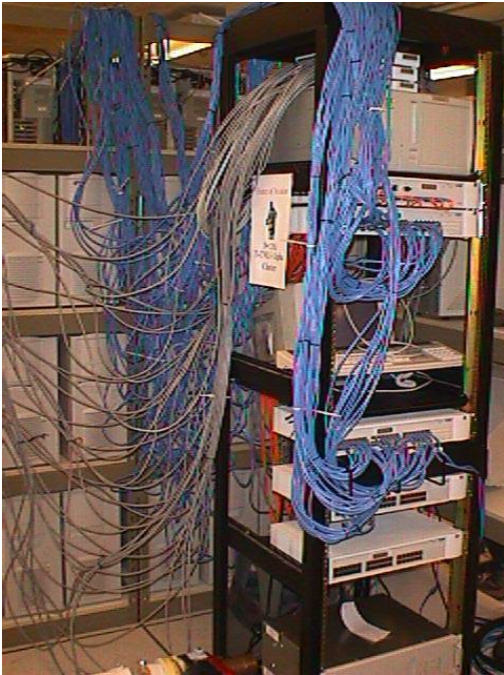


Source: Supermicro

Linux-Cluster (History)

- 1995 Beowulf-Cluster (NASA)
PentiumPro/II as nodes, FastEthernet, MPI
- 1997 Avalon (Los Alamos) 70 Alpha-Processors
- 1998 Ad-hoc-Cluster of 560 nodes for one night (Paderborn)
- 1999 Siemens hpcLine 192 Pentium II (Paderborn)
- 2000 Cplant-Cluster (Sandia National Laboratories)
1000 Alpha-Prozessoren
- 2001 Locus Supercluster 1416 Prozessoren
- 2002 Linux NetworX 2304 Prozessoren (Xeon)
-

Avalon-Cluster (Los Alamos Nat. Lab.)



140 Alpha-Processors

World record 1998: 560 nodes running Linux

WDR-Computer night 1998, HNF / Uni Paderborn



Spontaneously connected heterogeneous cluster of private PCs brought by visitors.

Network: Gigabit-/Fast Ethernet

Supercomputer (Cluster)



ASCI White



ASCI Q



ASCI Blue-Pacific

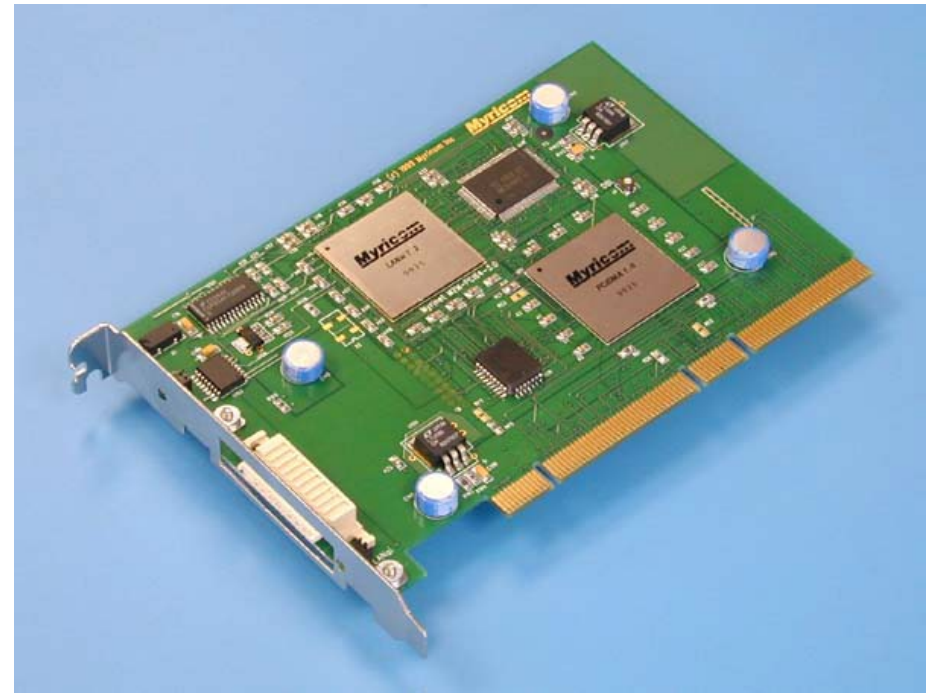
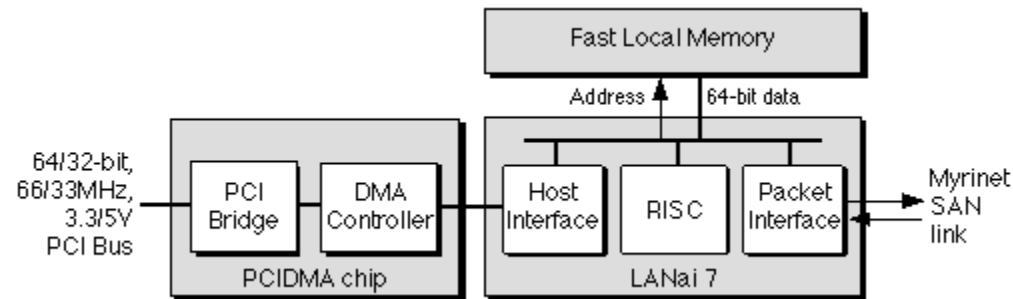
Lawrence Livermore National Laboratory



ASCI Blue Pacific

- **Fast Ethernet**
 - Bandwidth: 100 Mb/s
 - Latency: ca. 80 μ sec (userlevel to userlevel)
 - Was highly prevalent (low cost)
- **Gigabit-Ethernet**
 - Bandwidth: 1 Gb/s
 - Latency: ca. 80 μ sec (userlevel to userlevel, raw: 1-12 μ sec)
 - Today widespread (159 of Top500, Nov 2012)
- **10 Gigabit-Ethernet**
 - Bandwidth: 10 Gbit/s
 - Latency: ca. 80 μ sec (userlevel to userlevel, raw: 2-4 μ sec)
 - Already in use (30 of Top500, Nov 2012)

- Bandwidth: 4 Gb/s
- Real : 495 MB/s
- Latency: ca. 5 μ sec
- Point-to-Point
- Message Passing
- Switch-based
- Still used (3 in Top 500, Nov 2012, using 10G Myrinet)



Scalable Coherent Interface

- SCI = Scalable Coherent Interface, ANSI/IEEE 1596-1992.
- 64-bit global address space (16-bit node ID, 48-bit local memory).
- Guaranteed data transport by packet-based handshake protocol.
- Interfaces with two unidirectional Links that can work simultaneously.
- Parallel copper or serial optical fibers.
- Low latency ($<2\mu\text{s}$).
- Supports shared memory and message passing.
- Defines Cache coherence protocol (not available everywhere).

Scalable Coherent Interface (SCI)

Standard IEEE 1594

Bandwidth: 667MB/s

Real : 386MB/s

Latency: 1,4 μ sec

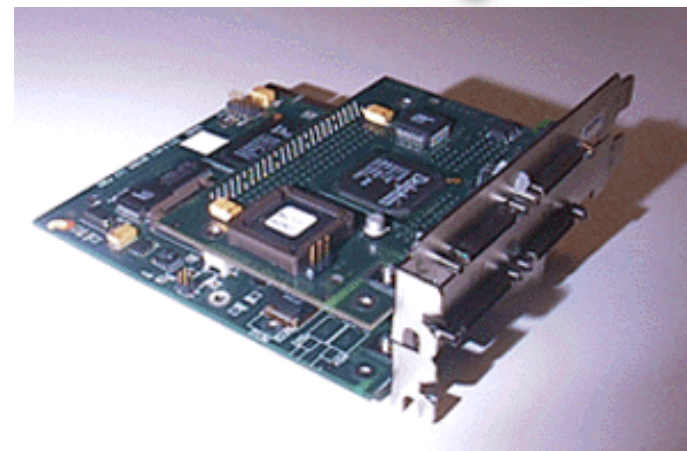
Shared Memory in HW

Usually 2D/3D-Torus

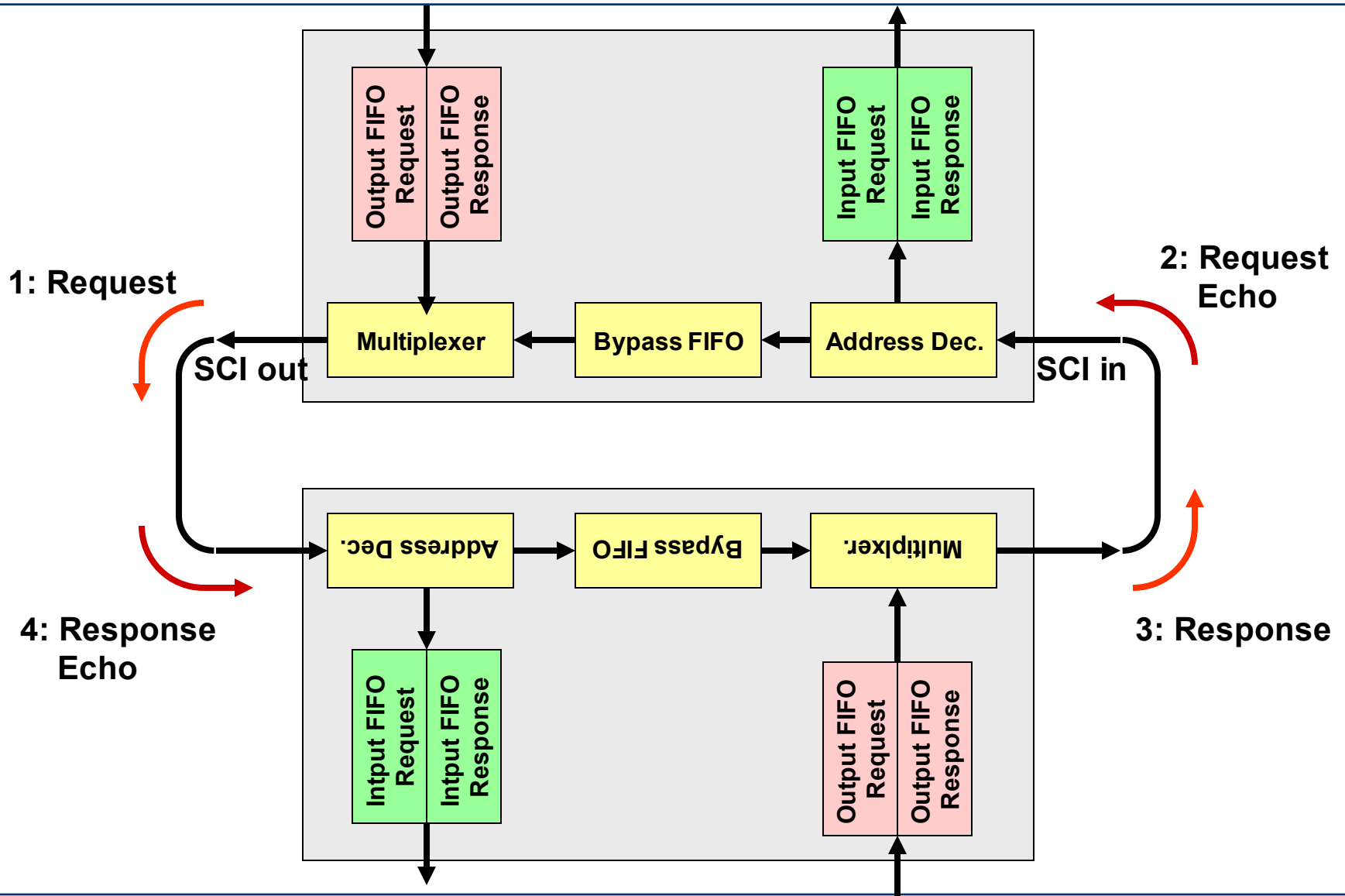
Scarcely used

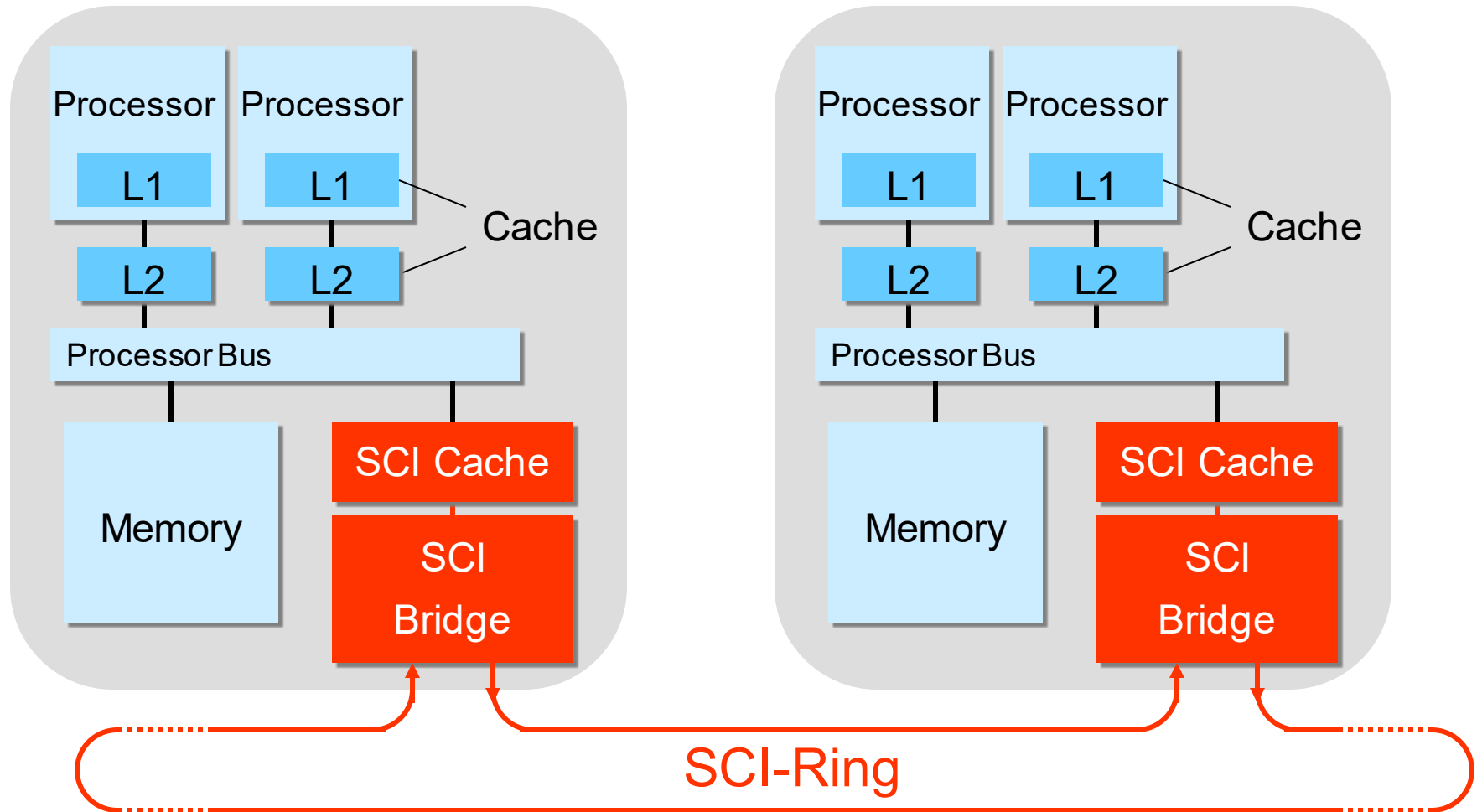
(last time Nov. 2005 in Top500)

(Data from Dolphin Inc.)

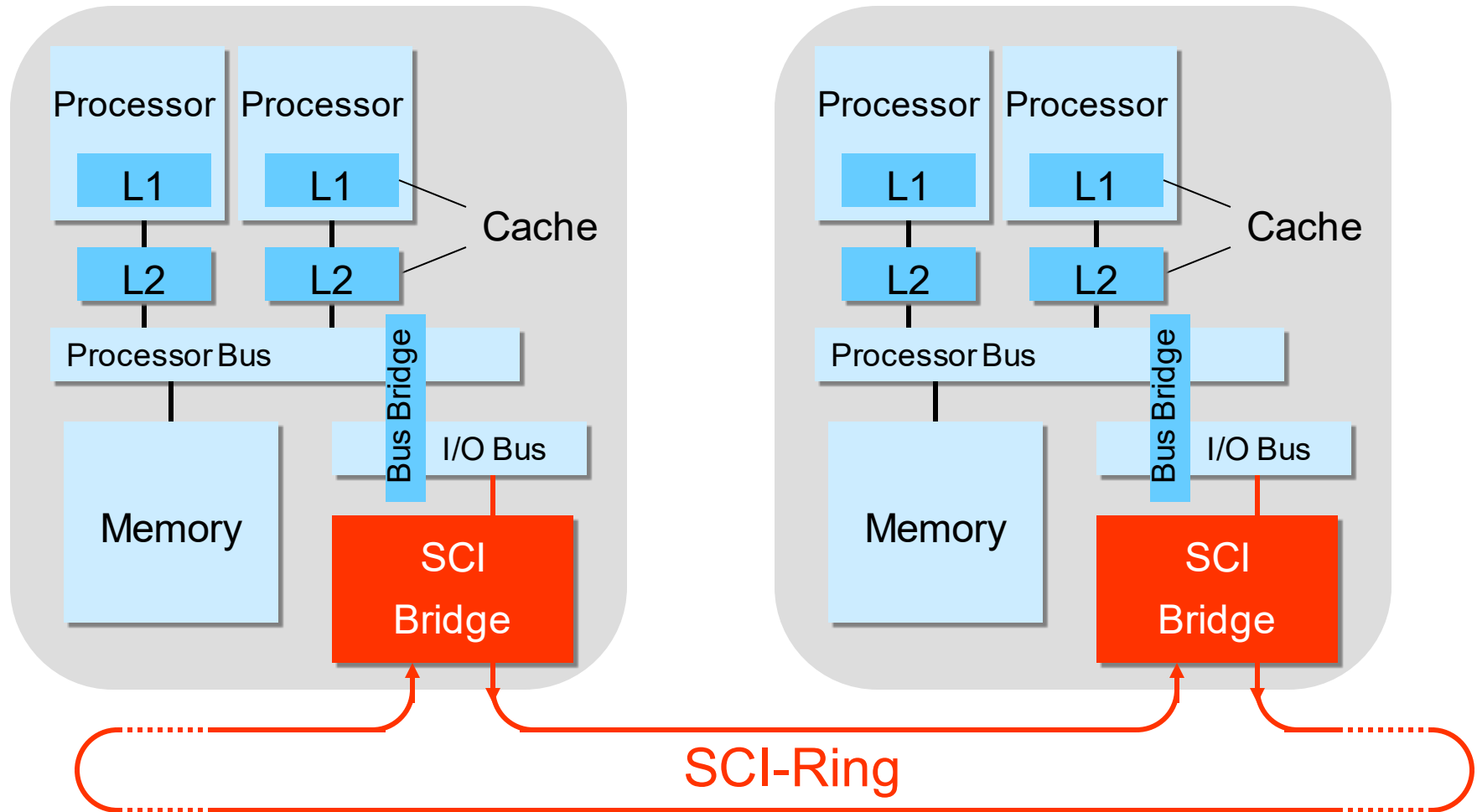


SCI Nodes





SCI for PC-Cluster (Dolphin-Adapter)



Example: SCI-Cluster (Uni Paderborn)

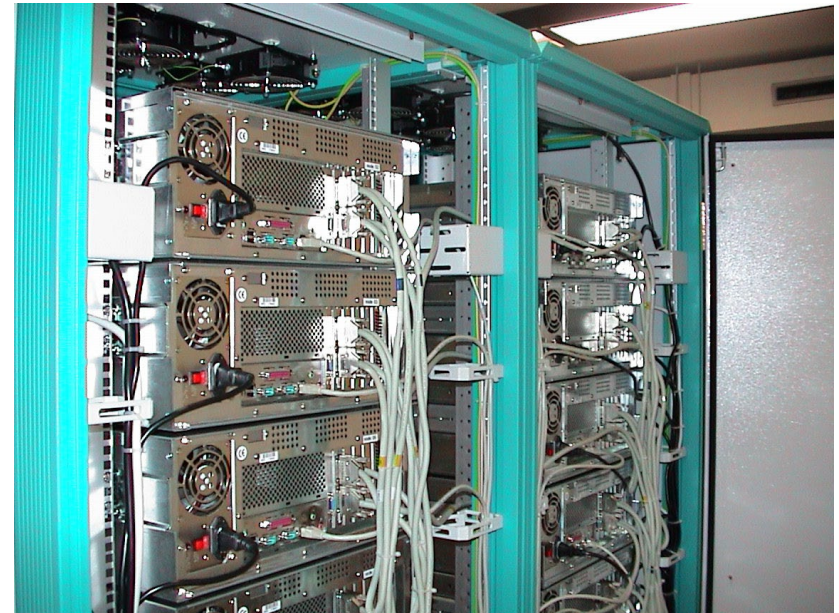
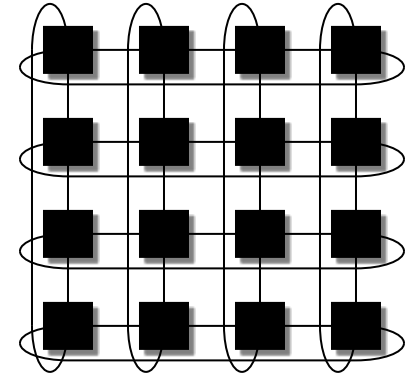
- 96 Dual processor-PCs (Pentium III, 850 MHz) with total 48 GB memory, connected as SCI-Rings (16x6-Torus).
- NUMA-Architecture: Hardware access to remote memory



Siemens
hpc-Line

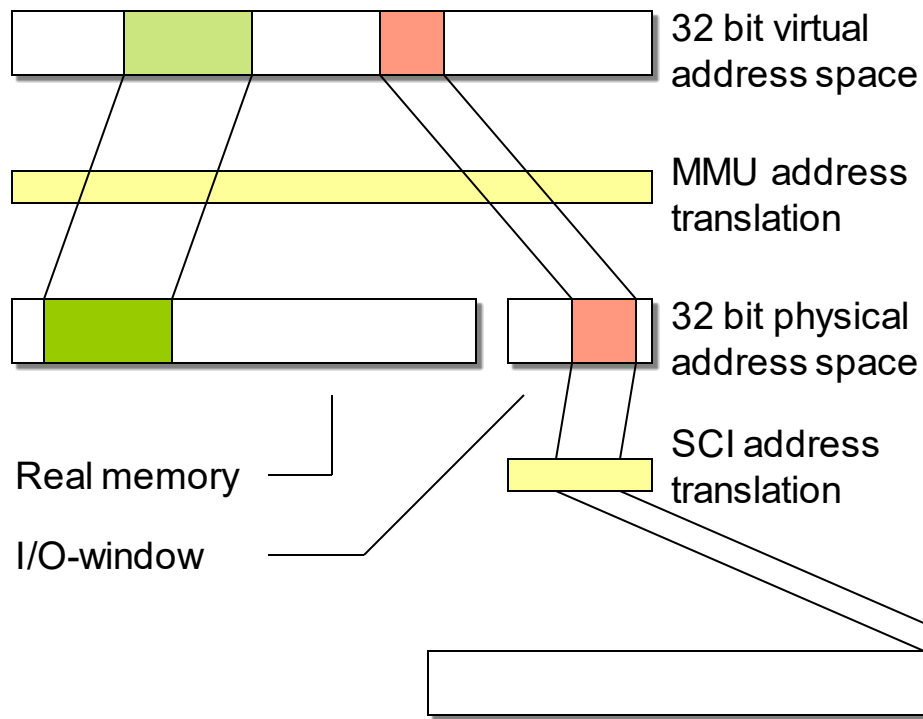
SCI-Cluster (KBS, TU Berlin)

- Linux-Cluster
- 16 Dualprocessor-nodes with
 - 1,7 GHz P4
 - 1 GB memory
 - 2D-Torus SCI (Scalable Coherent Interface)

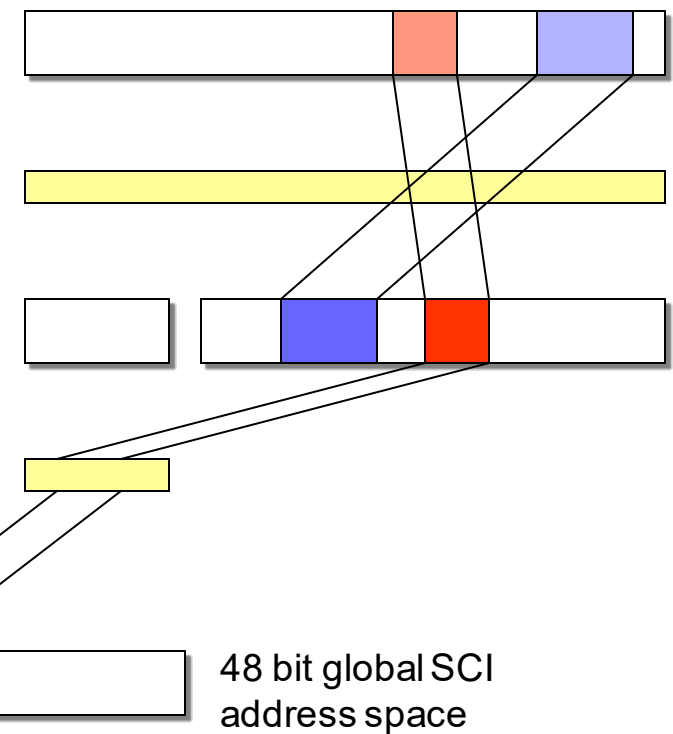


SCI address translation

Process 1 on node A

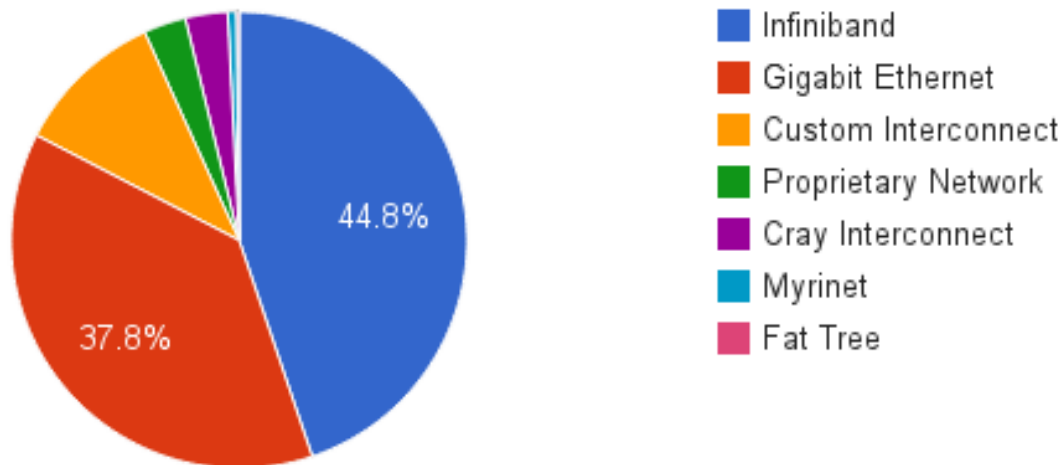


Process 2 on node B



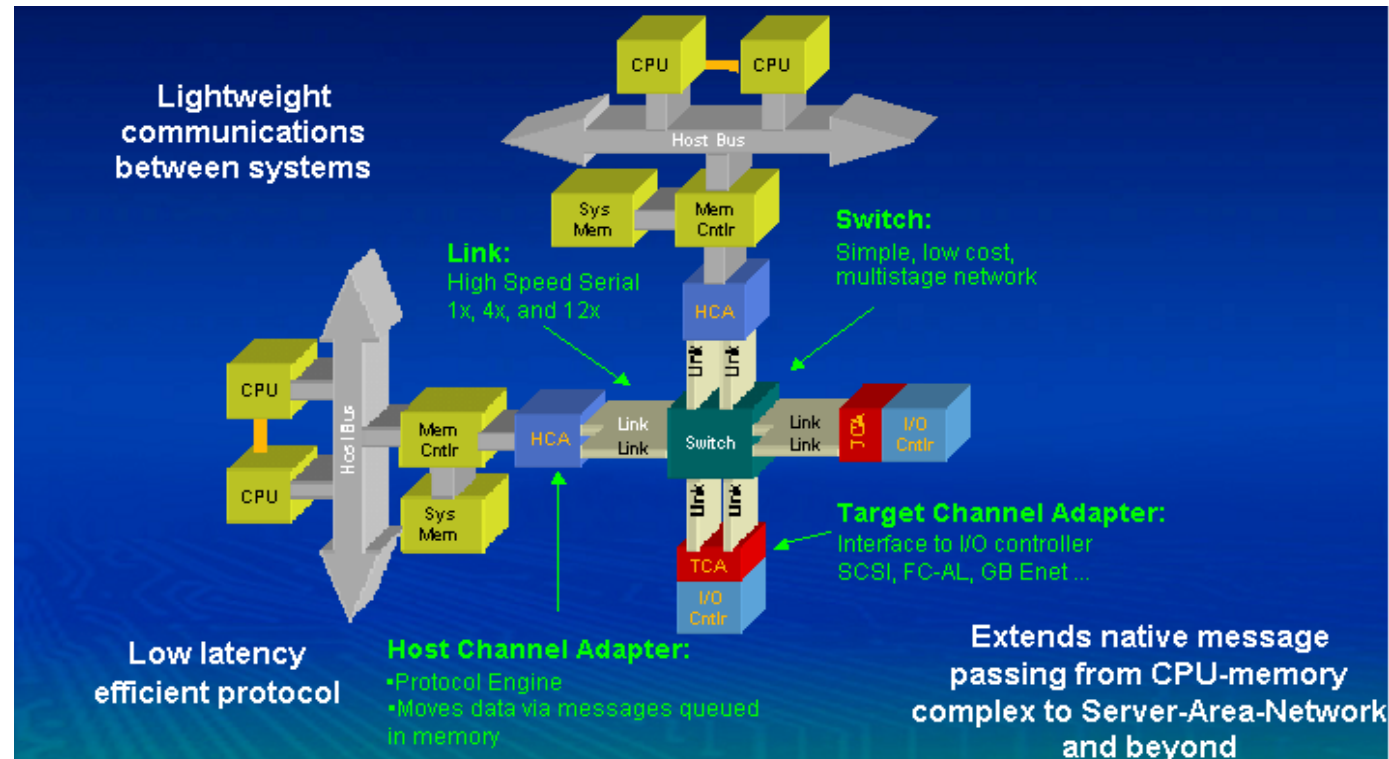
- New standard for coupling devices and compute nodes (replacement for system busses like PCI)
- single link: 2,5 GBit/sec
- Up to 12 links in parallel
- Most used today (224 of Top500, Nov 2012, different configurations)

Interconnect Family System Share



Source: top500.org

- Current technology for building HPC-Cluster
- 2.5 Gbps per Link
- ca. 2 μ sec Latency

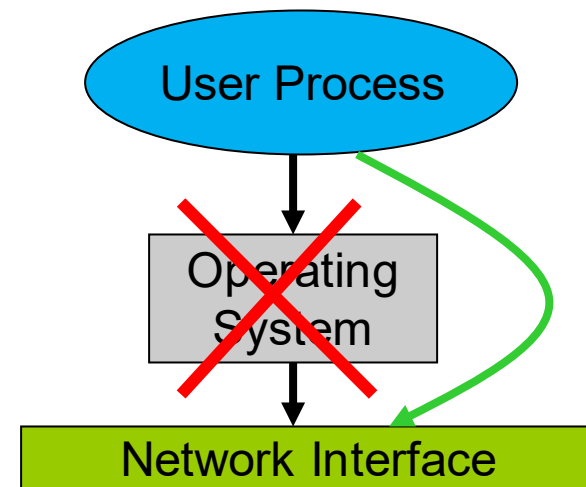


Virtual Interface Architecture

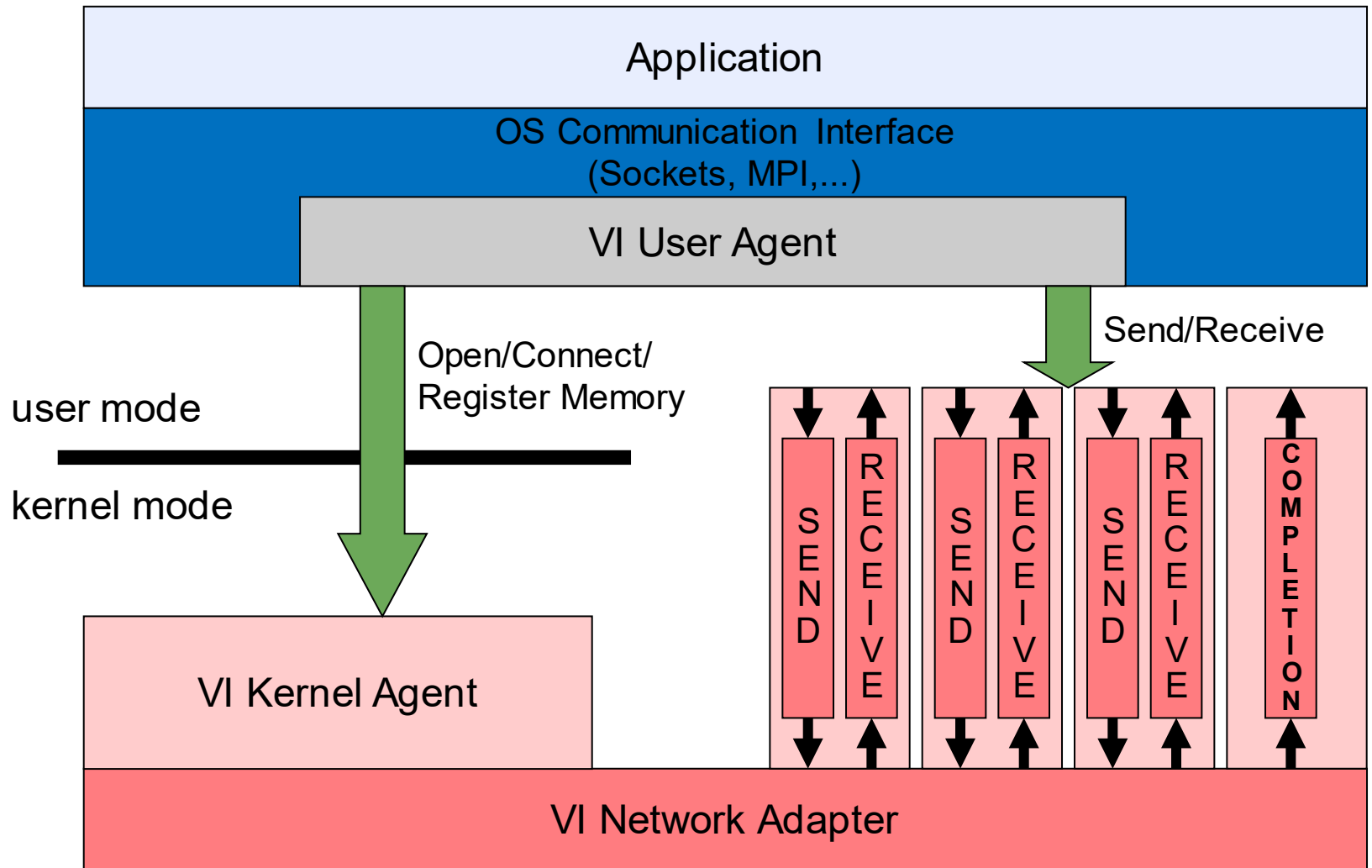
- To standardize the diversity of high performance network protocols, an industry consortium, consisting of Microsoft, Compaq (now HP) and Intel defined the Virtual Interface Architecture (VIA).
- Central element is the concept of a user-level interface, that allows direct access to the network interface bypassing the operating system.

Examples:

- Active Messages (UCB);
- Fast Messages (UIUC);
- U-Net (Cornell);
- BIP (Univ. Lyon, France);

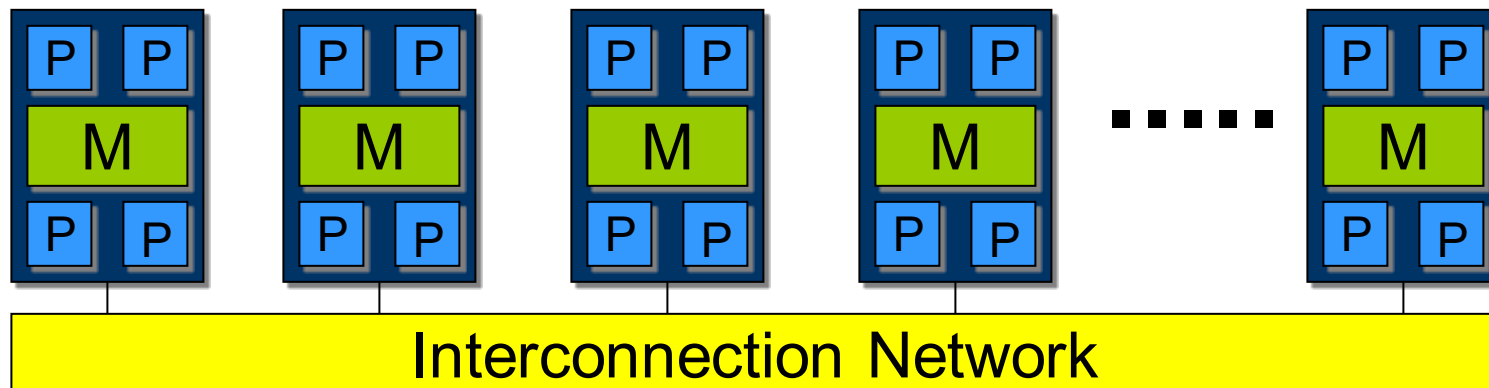


source: M. Baker

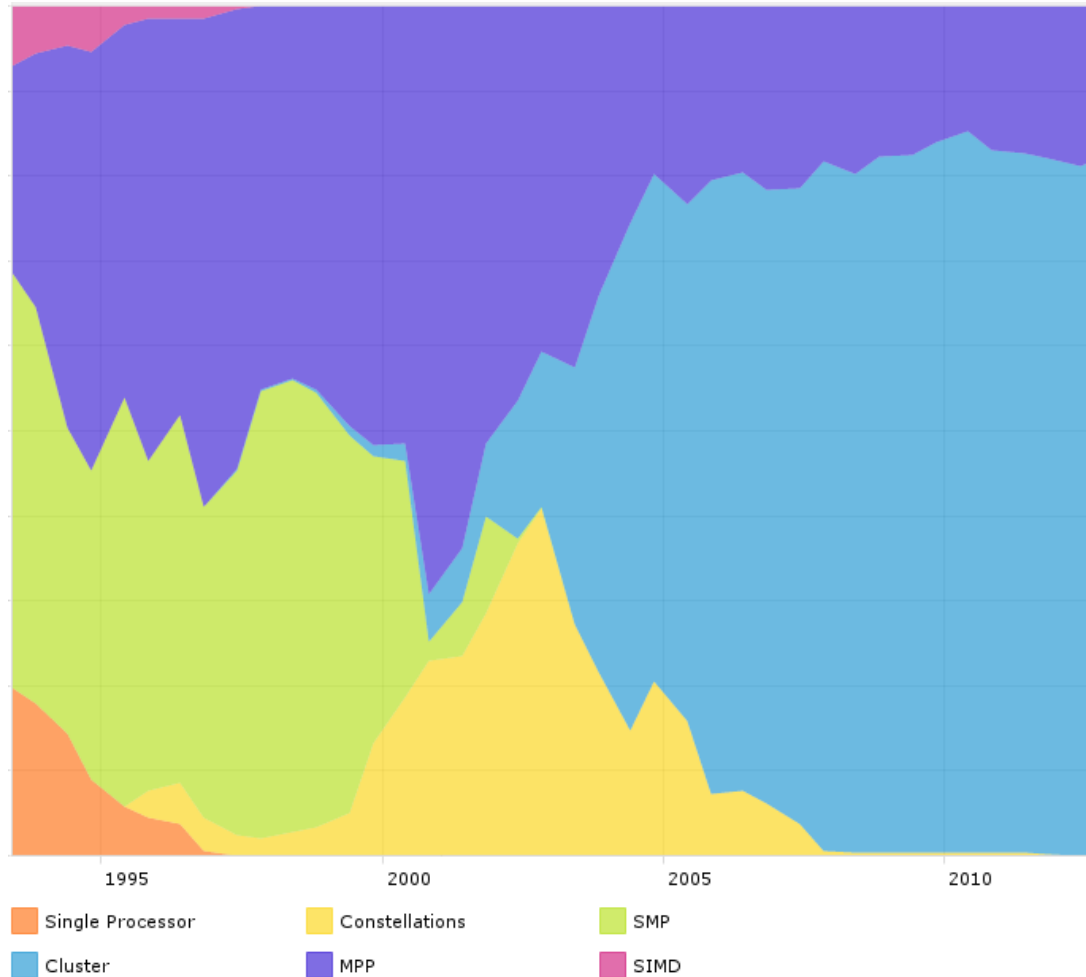


2.4 Architectural Trends in HPC

- Cost saving by usage of mass products
- Usage of top-of-line standard processor
- Usage of top-of-line standard interconnect
- SMPs/multicore/manycore-processors as nodes
- Assemble such nodes to build arbitrarily large clusters



Top 500: Architecture

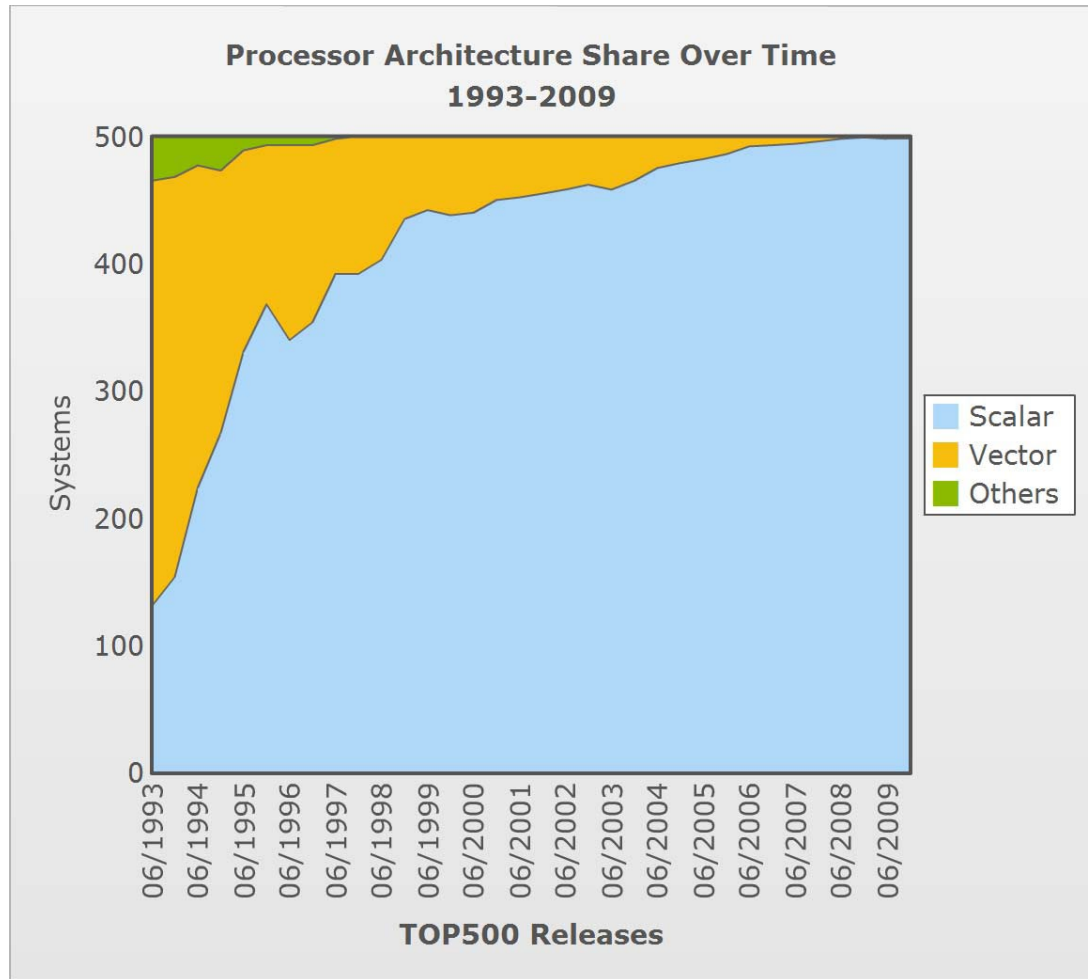


With respect to number of systems, TOP500, Nov 2012, www.top500.org

Notion of "Constellation"

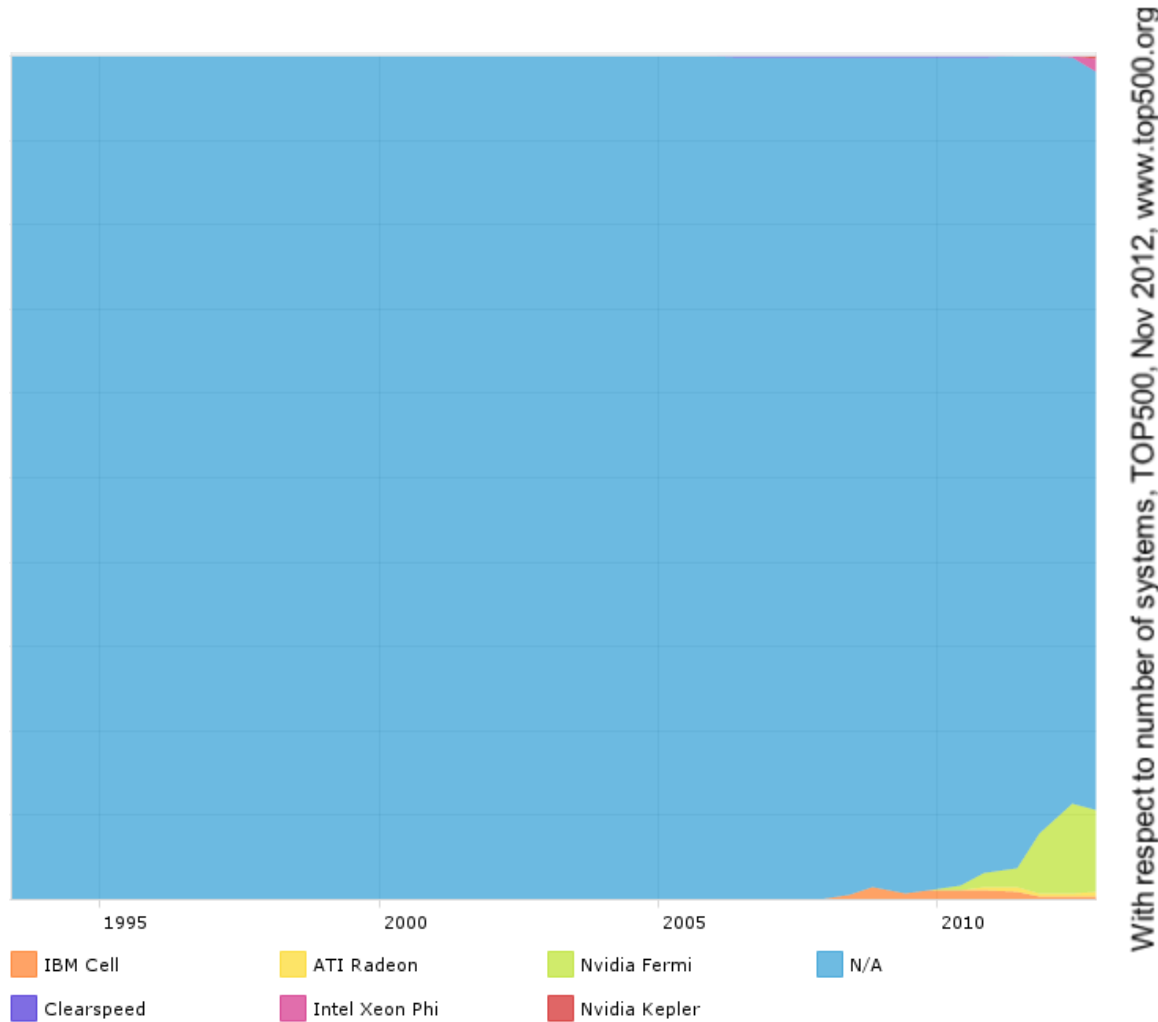
- Nodes of parallel computer consist of many processors (SMP) or multicore processors
- Thus, we have two layers of parallelism: the "Intranode-parallelism" and the "Internode parallelism"
- In the TOP500 terminology there is a distinction between *Cluster* and *Constellation*, depending on what form parallelism is dominant:
- If a machine has more processor cores per node than nodes at all, it is called *Constellation*.
- If a machine has more nodes than processor cores per node, than it is called *Cluster*.

Top 500: Processor Types

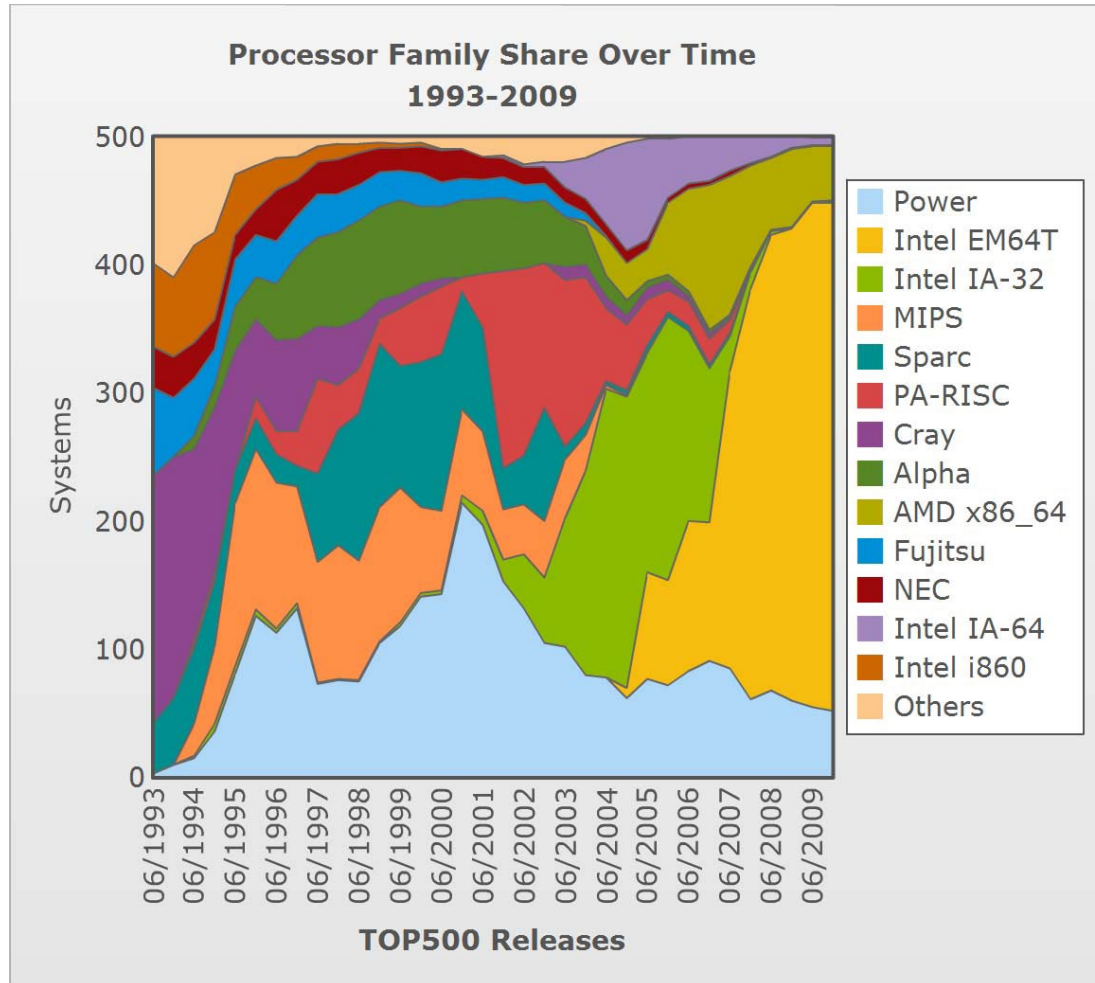


With respect to number of systems, TOP500, Nov 2012, www.top500.org

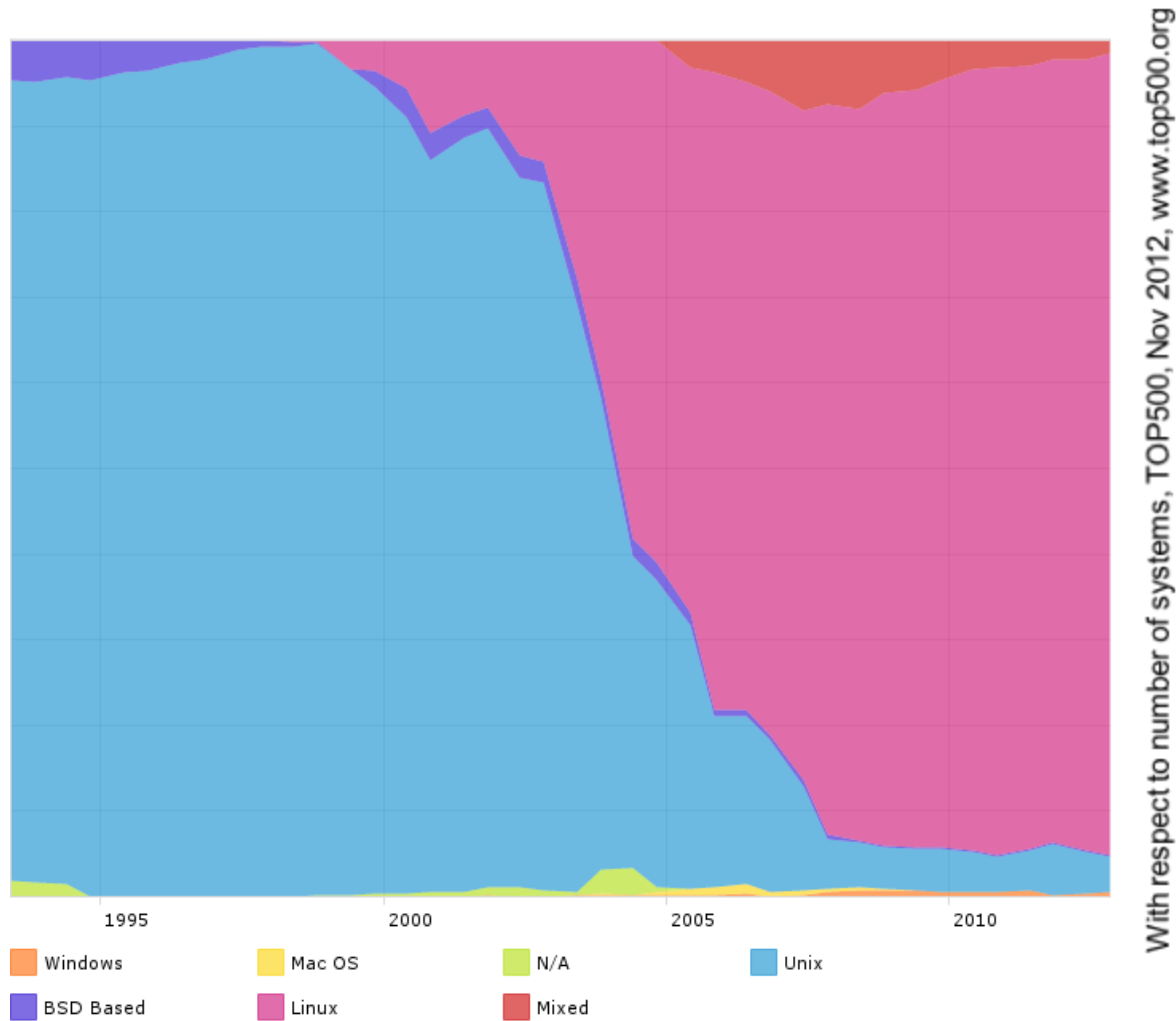
Top 500: Coprocessors



Top 500: Processor Family

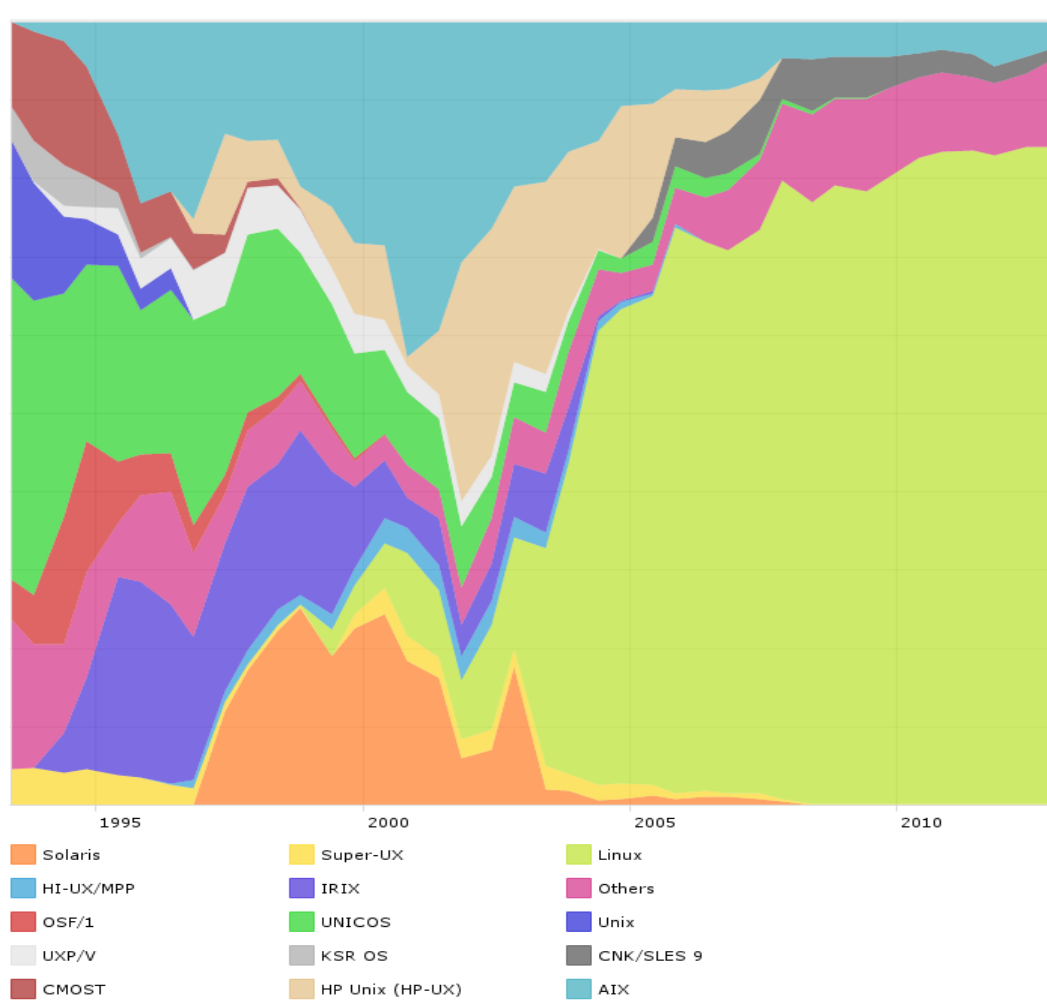


Top 500: Operating system families



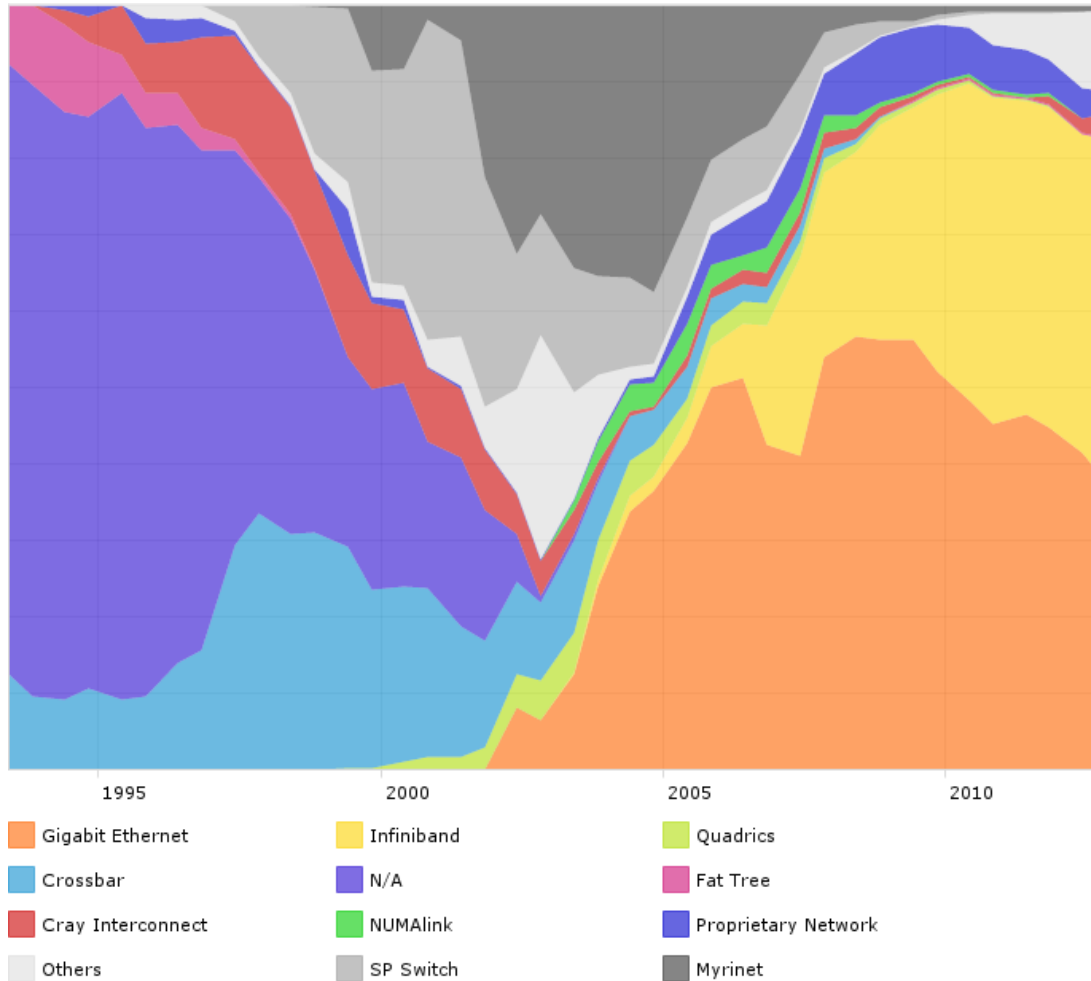
With respect to number of systems, TOP500, Nov 2012, www.top500.org

HPC-Operating systems



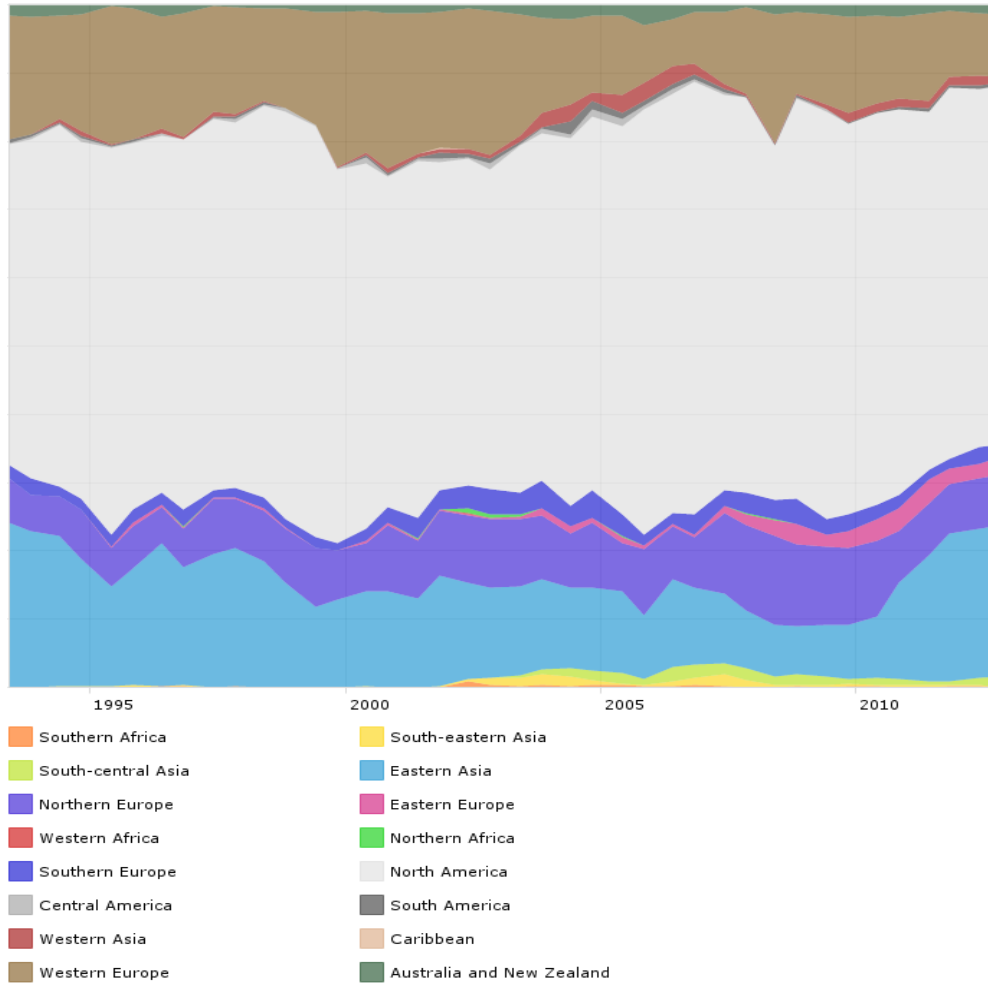
With respect to number of systems, TOP500, Nov 2012, www.top500.org

Top 500: Interconnect



With respect to number of systems, TOP500, Nov 2012, www.top500.org

Top 500: Geographical Regions



With respect to number of systems, TOP500, Nov 2012, www.top500.org

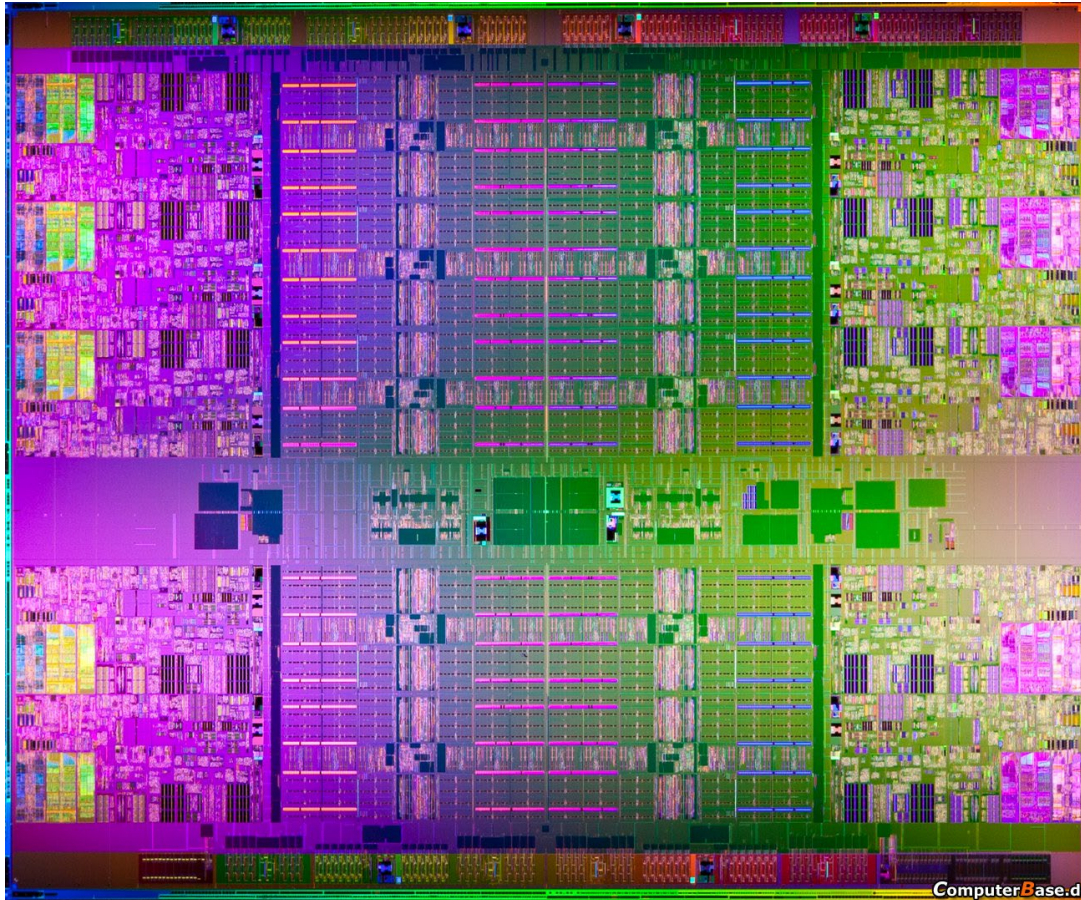
Transistors per processor

| Processor | Year of introduction | Transistor count |
|----------------|----------------------|------------------|
| Intel 4004 | 1971 | 2,300 |
| Intel 8008 | 1972 | 3,500 |
| MOS 6502 | 1975 | 3,510 |
| Zilog Z80 | 1976 | 8,500 |
| Intel 8088 | 1979 | 29,000 |
| Intel 80286 | 1982 | 134,000 |
| Motorola 68020 | 1984 | 200,000 |
| Intel 80386 | 1985 | 275,000 |
| Motorola 68030 | 1987 | 273,000 |
| Intel 80486 | 1989 | 1,180,000 |
| Motorola 68040 | 1990 | ~1,200,000 |

Transistors per processor

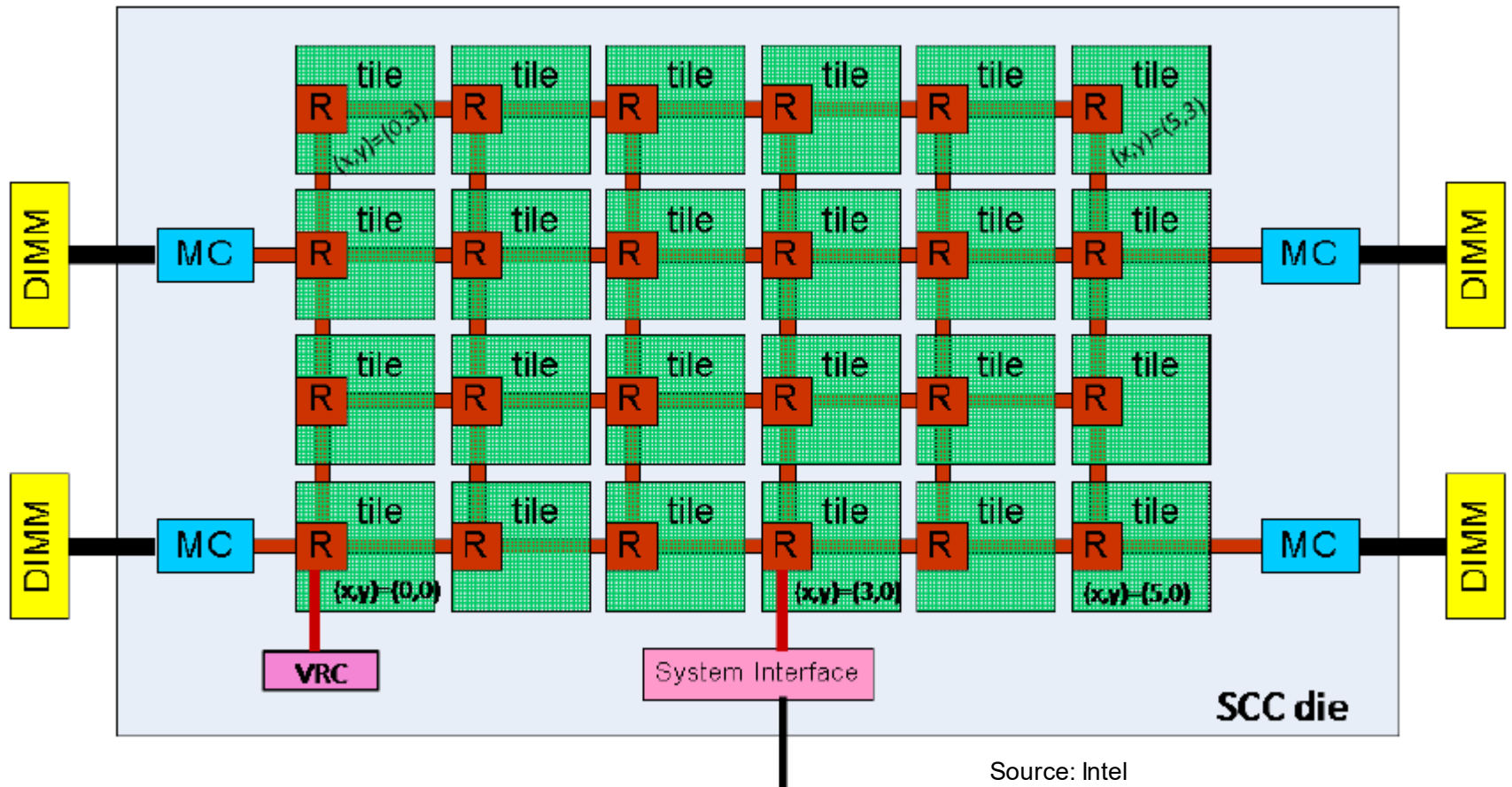
| Processor | Year of introduction | Transistor count |
|-------------------------|----------------------|------------------|
| Intel Pentium | 1993 | 3,100,000 |
| Intel Pentium Pro | 1995 | 5,500,000 |
| AMD K6 | 1997 | 8,800,000 |
| AMD K7 | 1999 | 22,000,000 |
| Intel Pentium 4 | 2000 | 42,000,000 |
| AMD K8 | 2003 | 105,900,000 |
| AMD K8 Dual Core | 2005 | 243,000,000 |
| IBM Power 6 | 2007 | 789,000,000 |
| AMD Opteron (6 core) | 2009 | 904,000,000 |
| Intel Xeon EX (10 core) | 2011 | 2,600,000,000 |
| Intel Itanium (8 core) | 2012 | 3,100,000,000 |

Xeon EX (Westmere EX) with 10 Cores

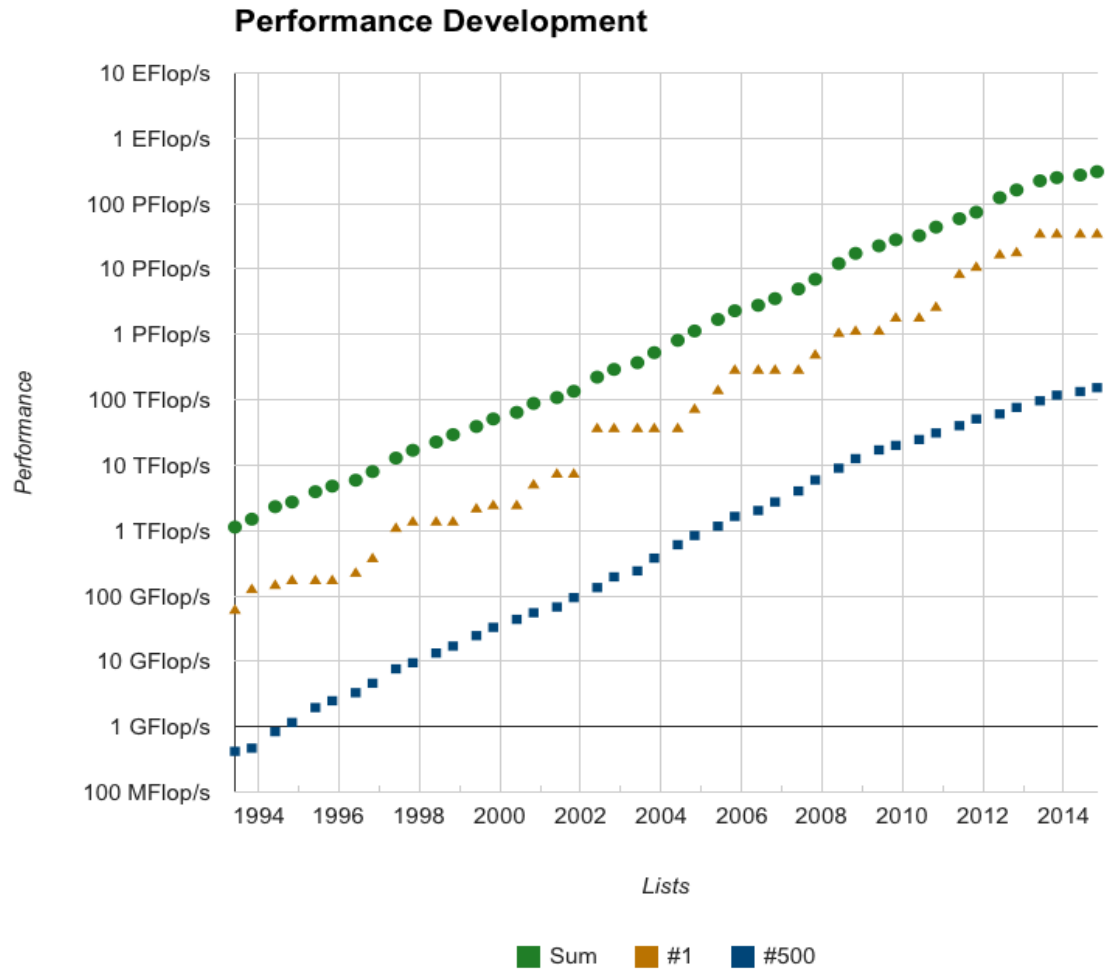


Source: computerbase.de

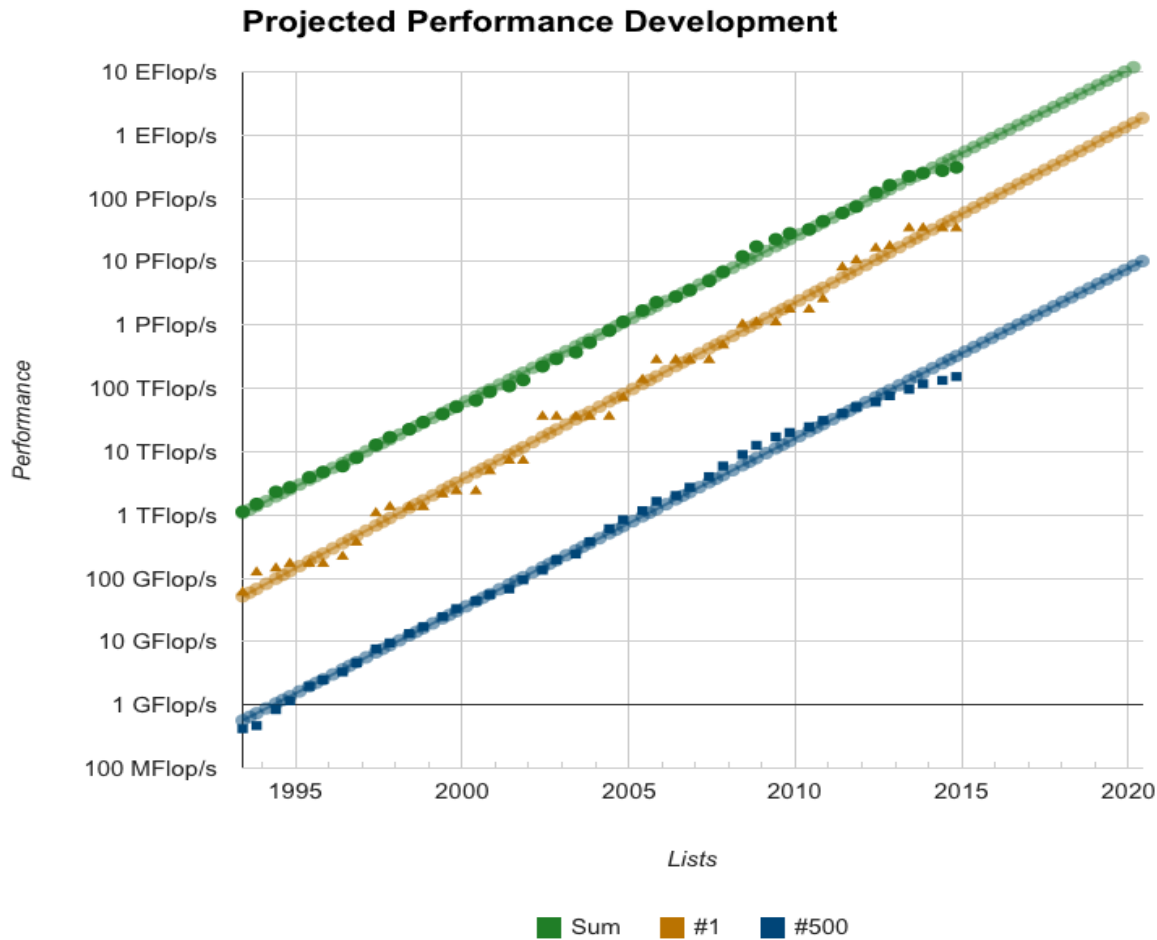
Intel Single Chip Cloud Computer: 48 Cores



Performance of Top 500

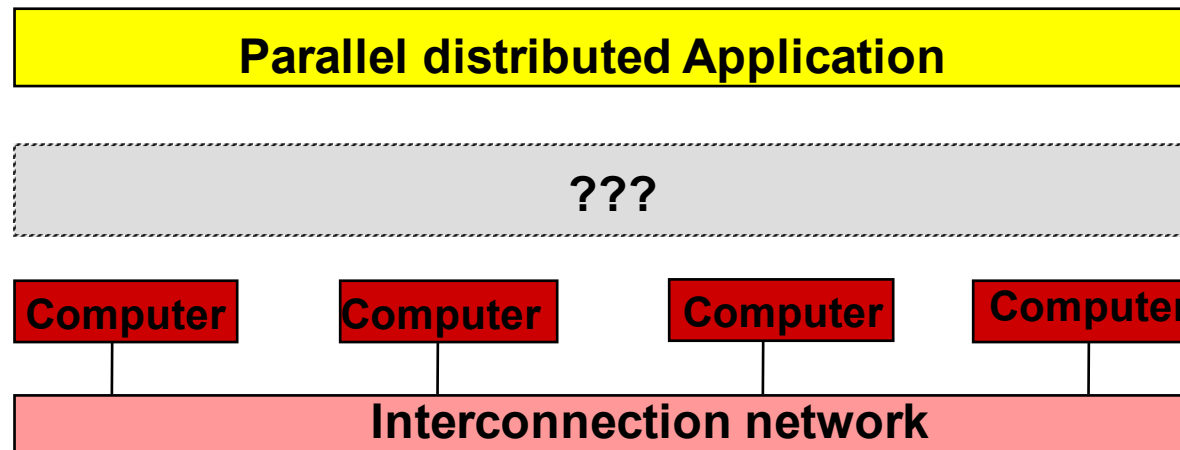


Top 500: Forecast



2.4 Software Architecture of Distributed Systems

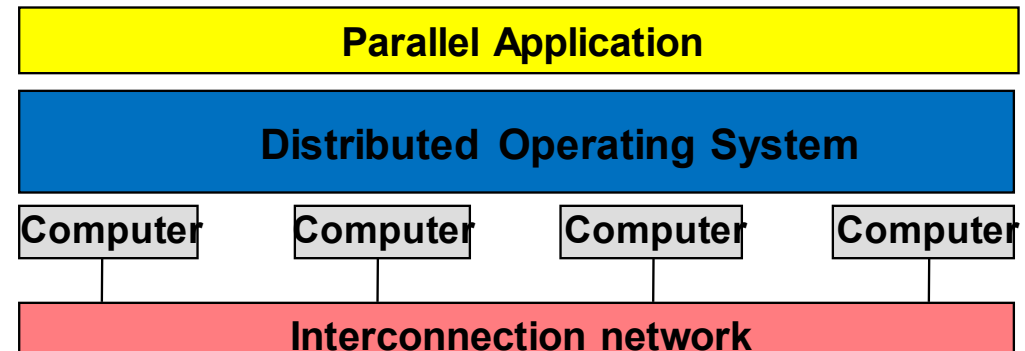
- Applications should be supported in a transparent way, i.e., we need a software infrastructure that hides the distribution to a high degree.



Two Architectural Approaches for Distributed Systems

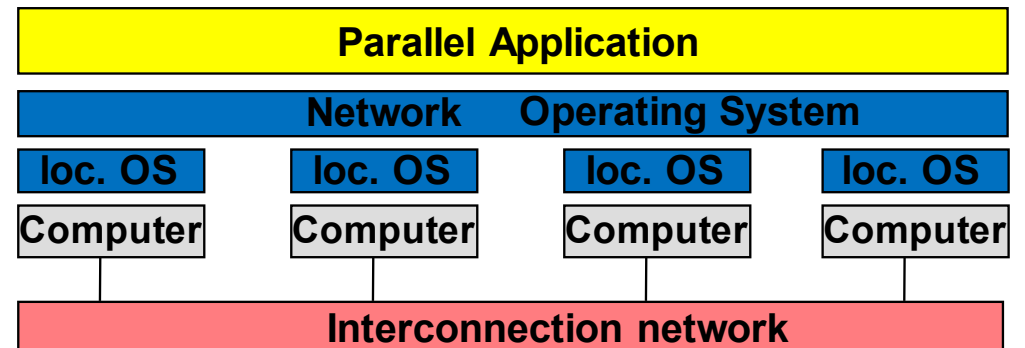
Distributed Operating System

The operating system itself provides the necessary functionality

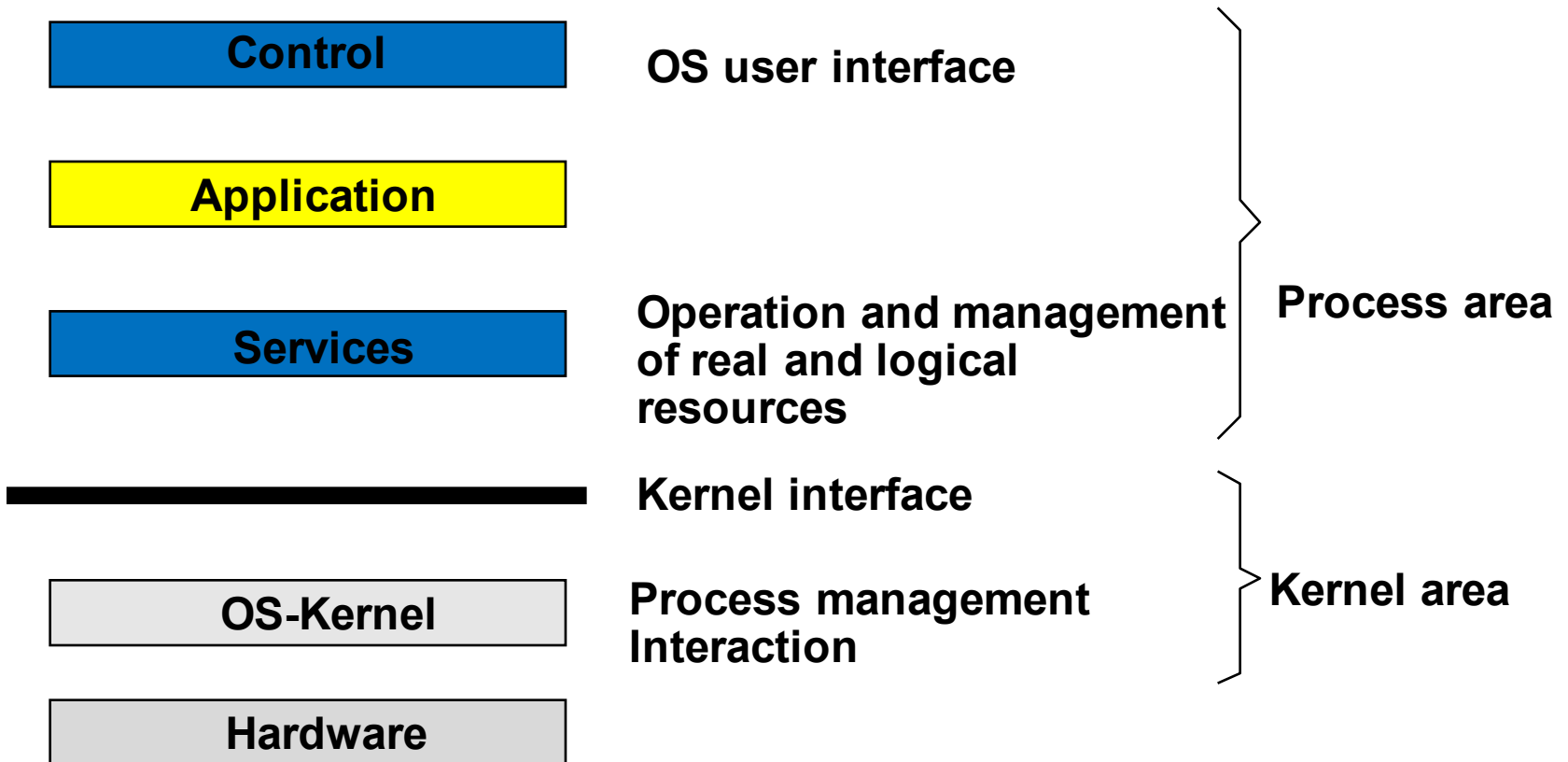


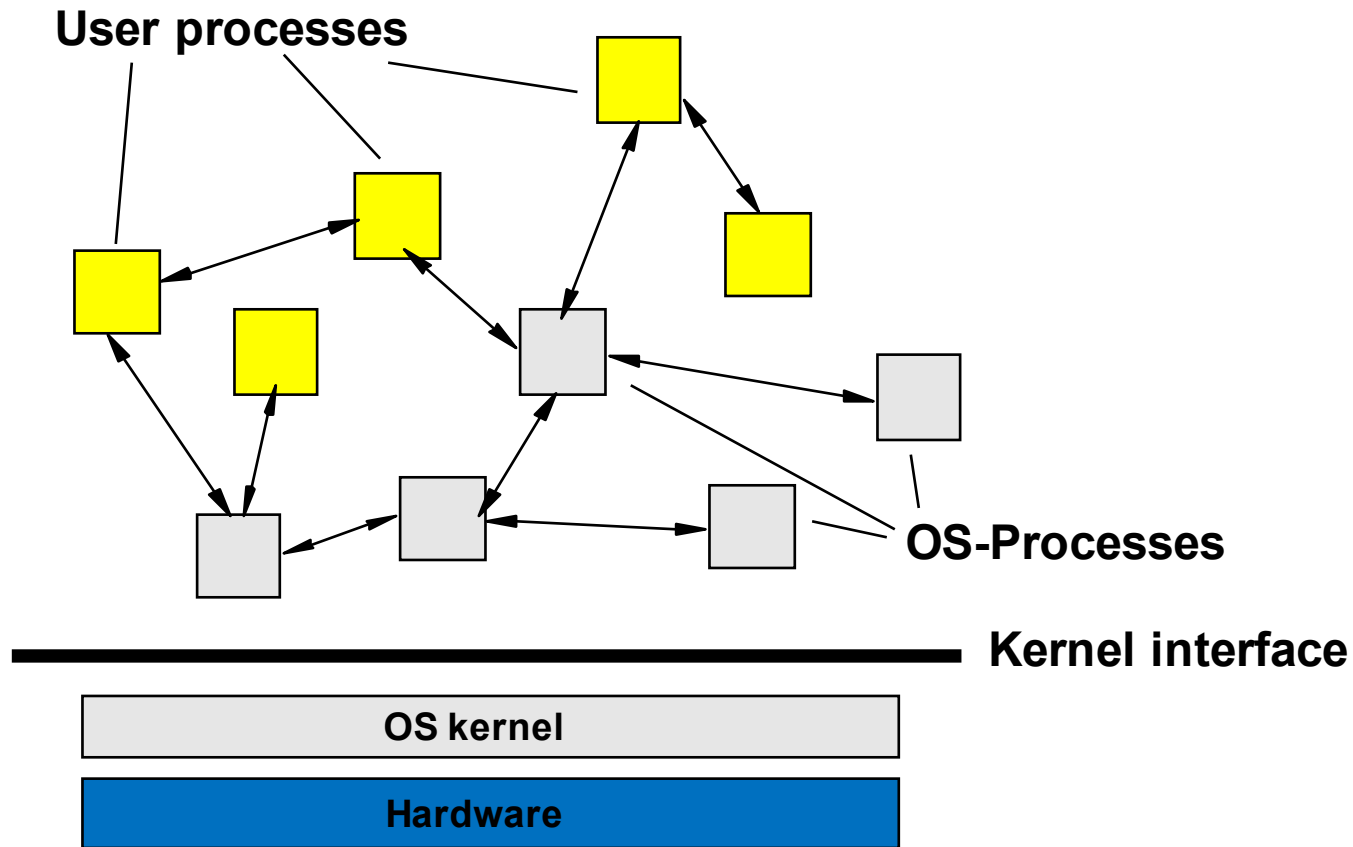
Network Operating System

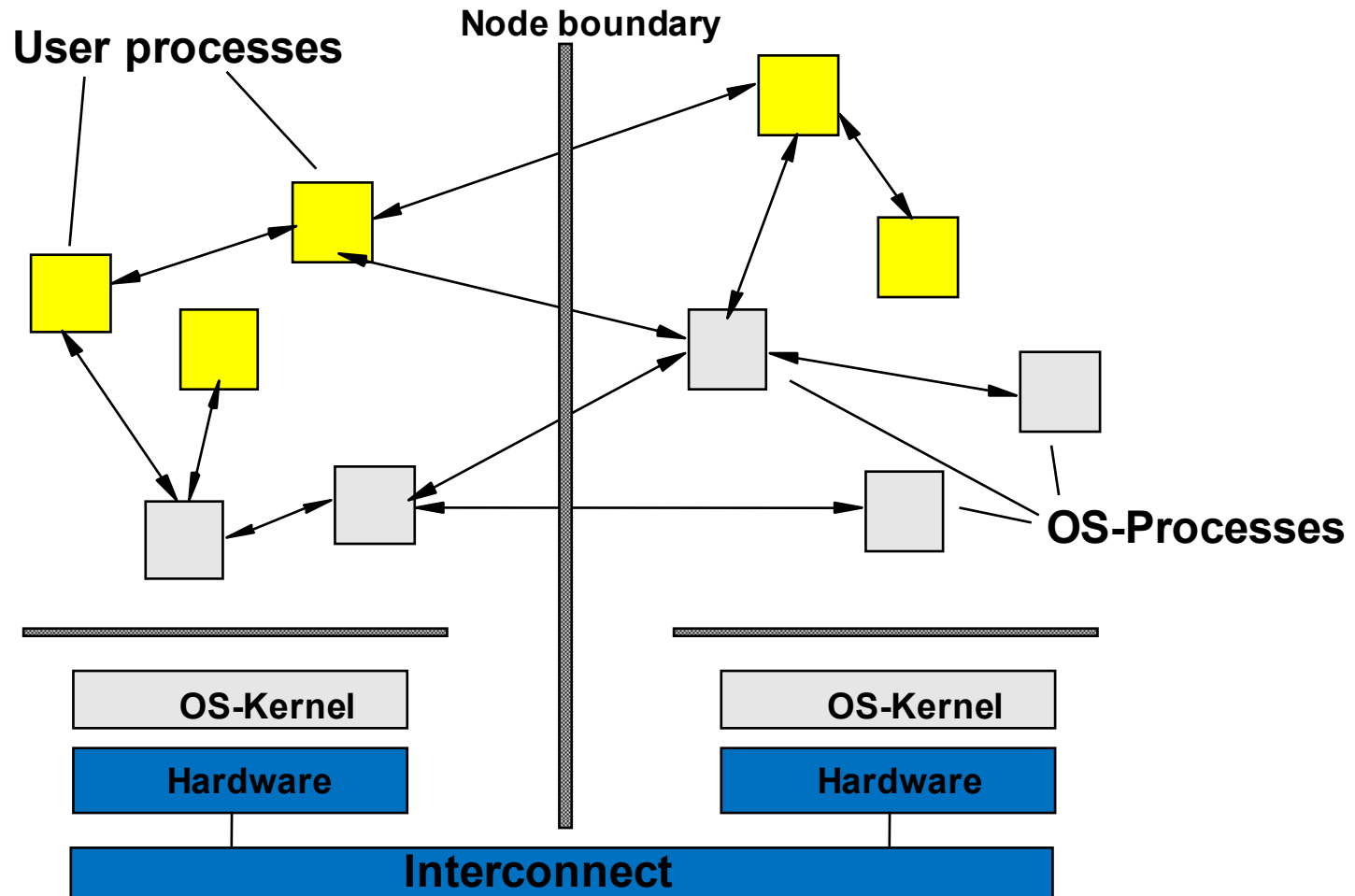
The local operating systems are complemented by an additional layer (Network OS, Middleware) that provides distributed functions

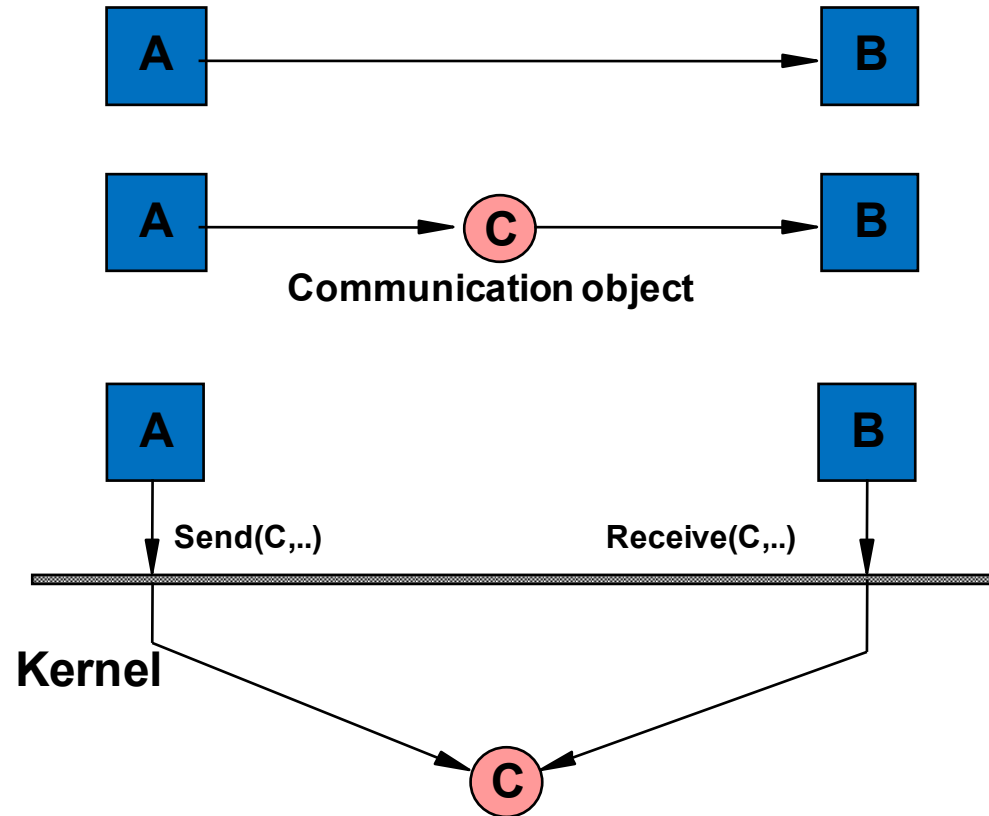


General OS Architecture (Microkernel)

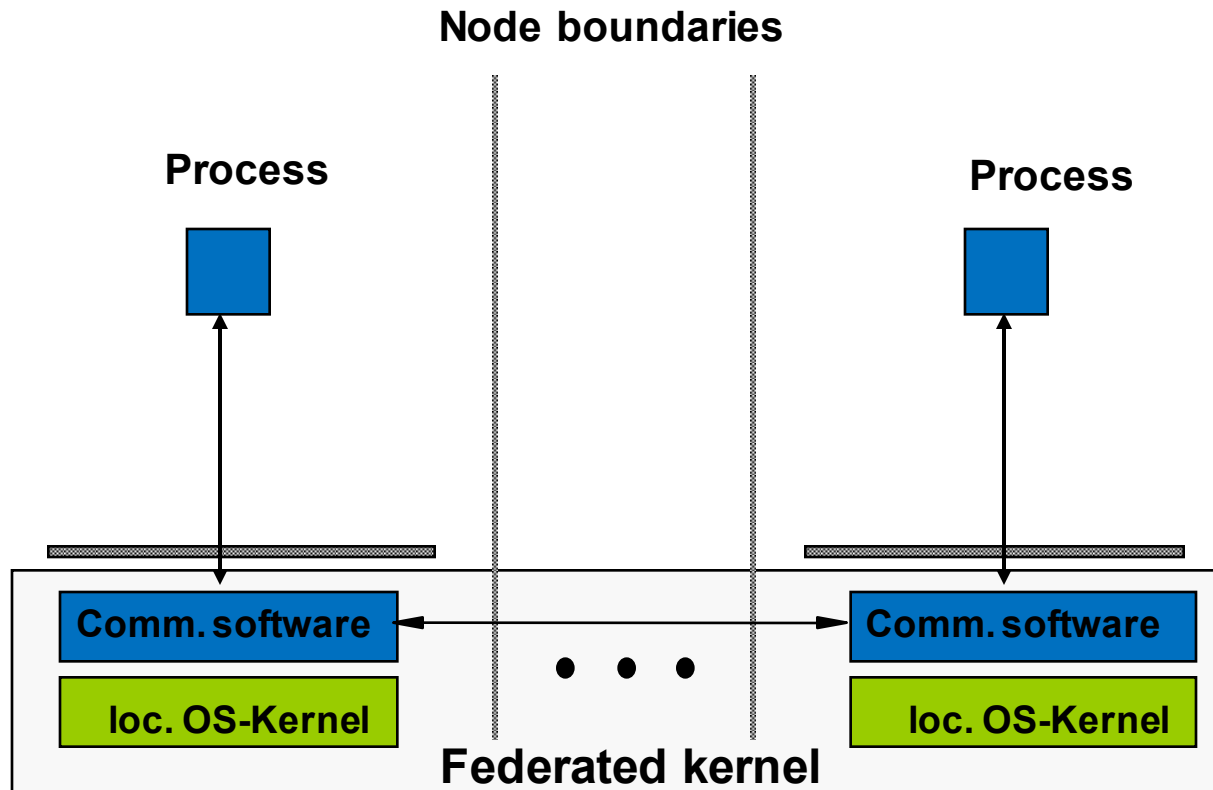


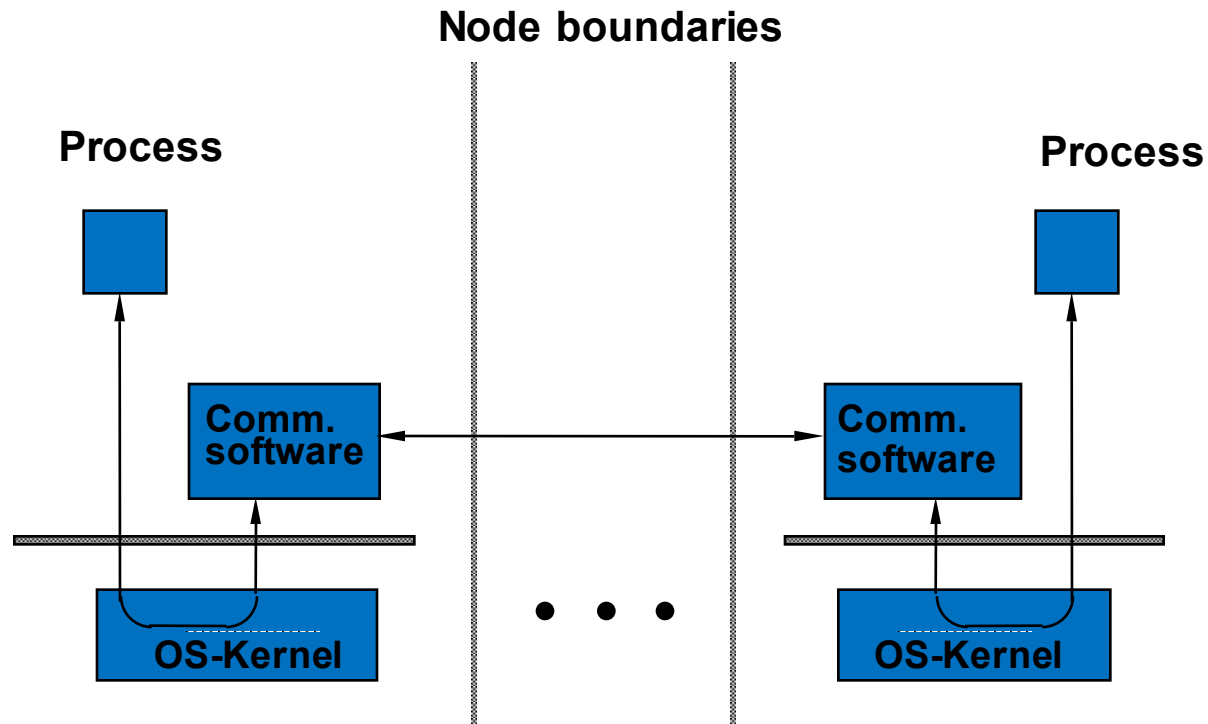






- To overcome the node boundaries in distributed systems for interaction, we need communication operations that are either integrated in the kernel (**kernel federation**) or as a component outside the kernel (**process federation**).
- When using the **kernel federation** the "distributedness" is hidden behind the kernel interface.
Kernel calls can refer to any object in the system regardless of its physical location.
The federation of all kernels is called the **federated kernel**.
- The **process federation** leaves the kernel interface untouched.
The local kernel is not aware of being part of a distributed system. This way, existing OS can be extended to become distributed OS.





Features of Distributed OS

| | Multiprocessor-OS | Distributed OS | Network-OS |
|------------------------------------|--------------------------|-------------------------|--|
| All nodes have the same OS? | yes | yes | no |
| How many copies of OS ? | 1 | n | n |
| Shared ready queue? | yes | no | no |
| Communication | Shared memory | Message exchange | Message exchange / Shared files |

Transparency I

- Transparency is the property that the user (almost) does not realize the distributedness of the system.
- Transparency is the main challenge when building distributed systems.
 - Access transparency
Access to remote objects in the same way as to local ones
 - Location transparency
 - Name transparency
Objects are addressed using names that are independent of their locations
 - User mobility
If the user changes his location he can address the objects with the same name as before.
 - Replication transparency
If for performance improvement (temporary) copies of data are generated, the consistency is taken care of automatically.

- Failure transparency
Failures of components should not be noticeable.
- Migration transparency
Migration of objects should not be noticeable.
- Concurrency transparency
The fact that that many users access objects from different locations concurrently must not lead to problems or faults.
- Scalability transparency
Adding more nodes to the system during operation should be possible. Algorithms should cope with the growth of the system.
- Performance transparency
No performance penalties due to uneven load distribution.

- Erich Strohmaier, Jack J. Dongarra, Hans W. Meuer, and Horst D. Simon: **Recent Trends in the Marketplace of High Performance Computing**, <http://www.netlib.org/utk/people/JackDongarra/PAPERS/Marketplace-top500-pc.pdf>
- Gregory Pfister: **In Search of Clusters** (2nd ed.), Prentice-Hall, 1998
- Thorsten v.Eicken, Werner Vogels: **Evolution of the Virtual Interface Architecture**, IEEE Computer, Nov. 1998
- Hans Werner Meuer: **The TOP500 Project: Looking Back over 15 Years of Supercomputing Experience**
http://www.top500.org/files/TOP500_Looking_back_HWM.pdf
- David Culler, J.P. Singh, Anoop Gupta: **Parallel Computer Architecture: A Hardware/Software Approach**. Morgan Kaufmann, 1999
- Hesham El-Rewini und Mostafa Abd-El-Barr: **Advanced Computer Architecture and Parallel Processing: Advanced Computer Architecture** v. 2; John Wiley & Sons, 2005
- V. Rajaraman und C.Siva Ram Murthy: **Parallel Computers: Architecture and Programming**; Prentice-Hall, 2004