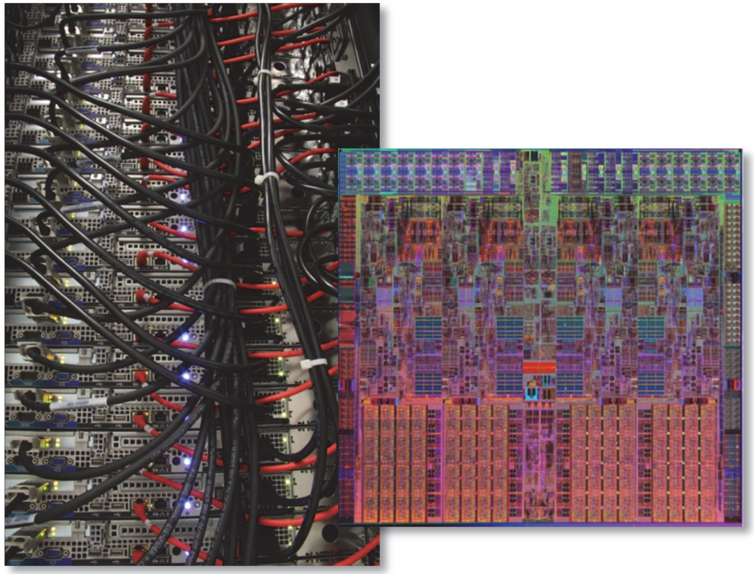


Where a calculator on the ENIAC is equipped with 18,000 vacuum tubes and weighs 30 tons, computers of the future may have only 1,000 vacuum tubes and perhaps weigh 1½ tons.

Popular Mechanics, March 1949

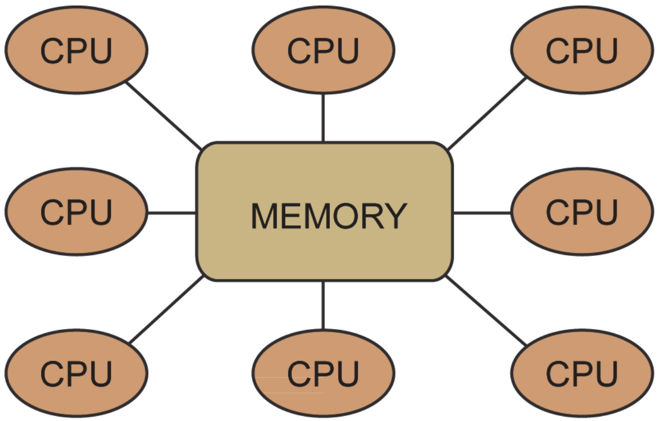
Lecture 1: the anatomy of a supercomputer

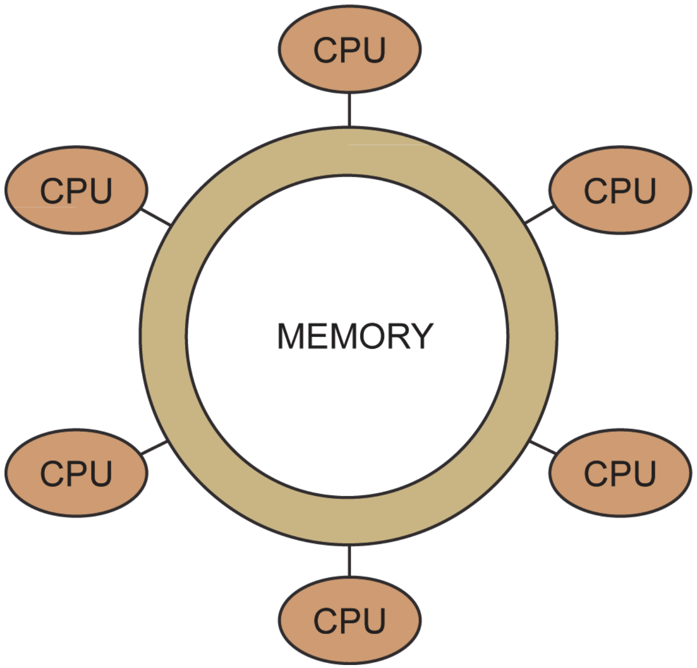
Clusters, shared-memory systems, latencies and interconnects, CPU and memory, power and cooling, storage, etc.

Dr Ilya Kuprov, University of Southampton, 2012

(for all lecture notes and video records see http://spindynamics.org)

Shared-memory systems





SHMEM system features

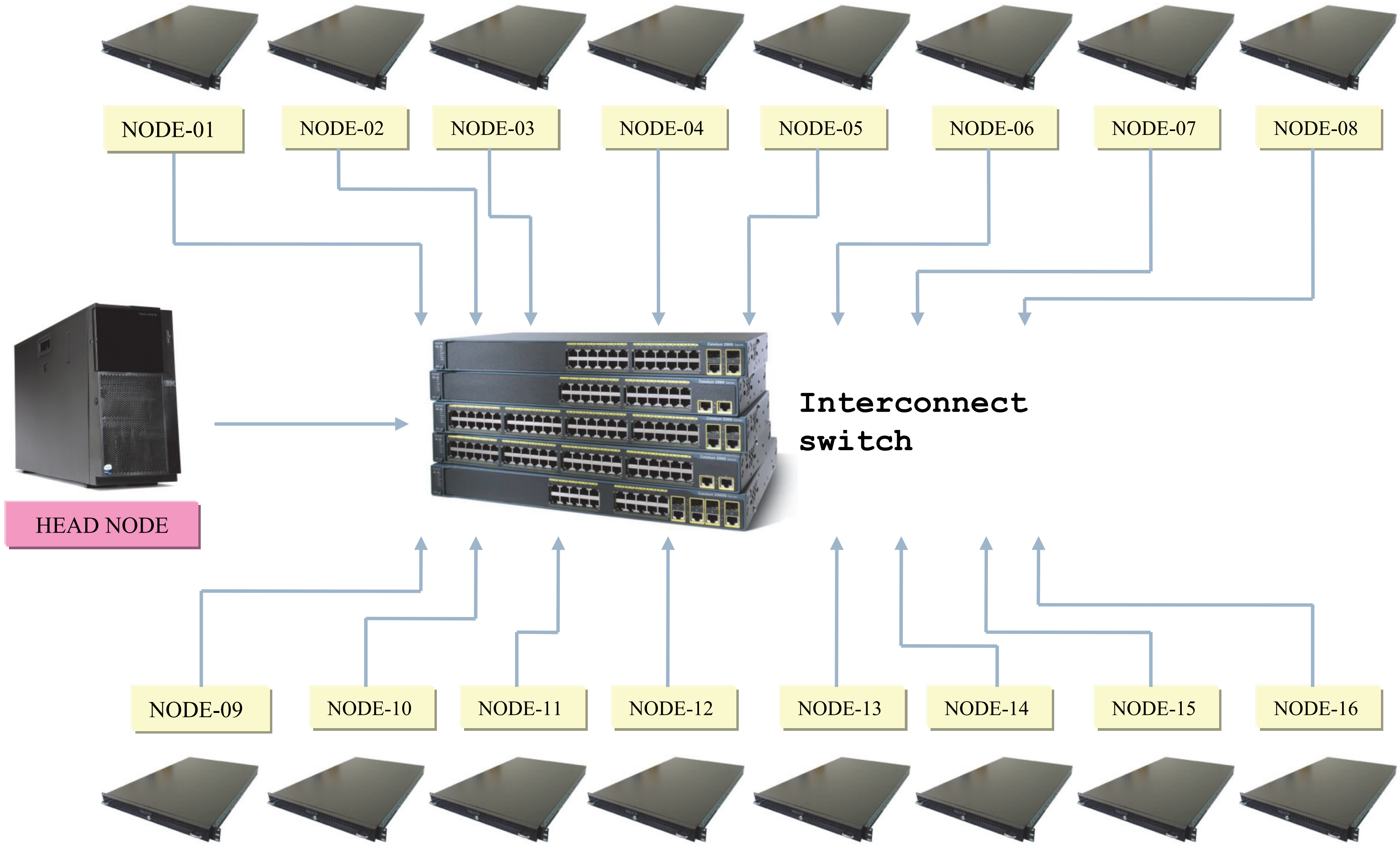
1. Very fast interconnect (nanosecond latencies) – every byte of memory is directly accessible to every CPU.
2. Historically the first type of parallel computer to appear.
3. Very expensive – a niche market of highly specialized applications.
4. Limited size – the largest realistic system has about 2,000 CPUs.
5. Cache coherence issues – the CPUs must update each other on the content of their caches.

OSC **Orac** system:

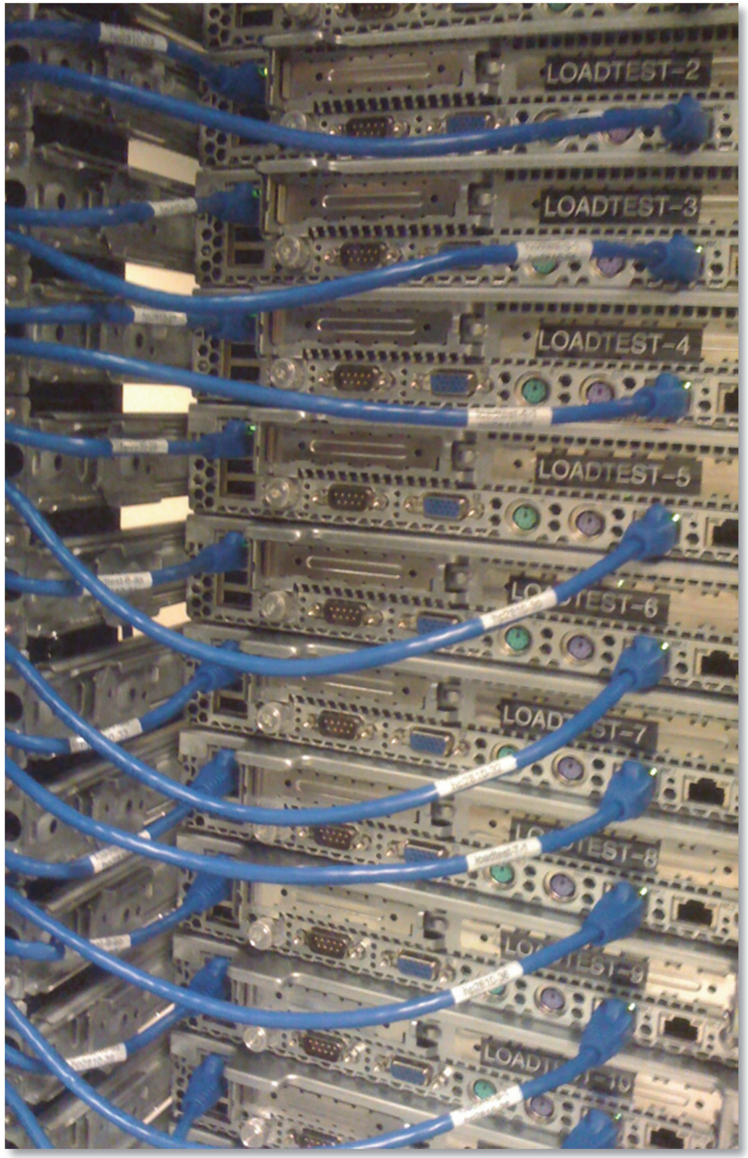
256 Itanium2 cores, 1024 GB of RAM, 5.5 TB of storage space, 1.6 TFlop peak.

Orac is often used for Gaussian jobs that require large-scale diagonalization.

Clusters


NODE-01
NODE-02
NODE-03
NODE-04
NODE-05
NODE-06
NODE-07
NODE-08
HEAD NODE
Interconnect switch
NODE-09
NODE-10
NODE-11
NODE-12
NODE-13
NODE-14
NODE-15
NODE-16

Clusters



Cluster features

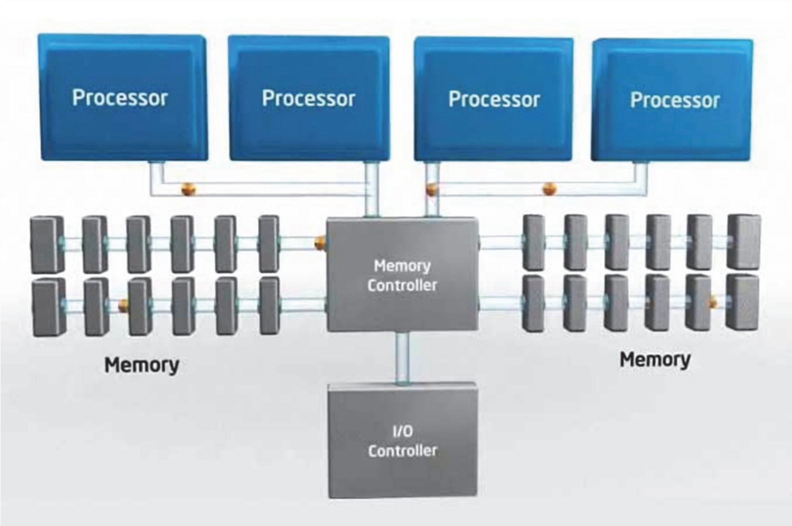
1. Heterogeneous interconnect – slow between nodes (micro- to millisecond latencies) and fast (nanoseconds) between the processors on the same node.
2. Quite cheap – the cost does not significantly exceed the cost of commodity hardware.
3. Large-scale deployments possible – up to 10,000 nodes on the biggest current installations.
4. Fault tolerant – failure of a single node does not shut down the cluster.
5. Green – idle nodes may be shut down.

OSC **Hal/Sal** system:

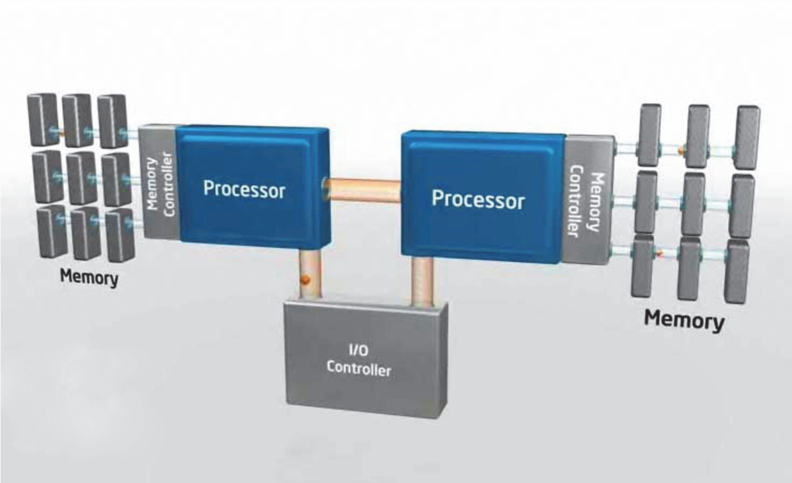
1152 Xeon cores, 3 TB of RAM, 13 TFlop peak, Infiniband interconnect.

Hal and Sal are mostly used for molecular dynamics and plane wave DFT jobs.

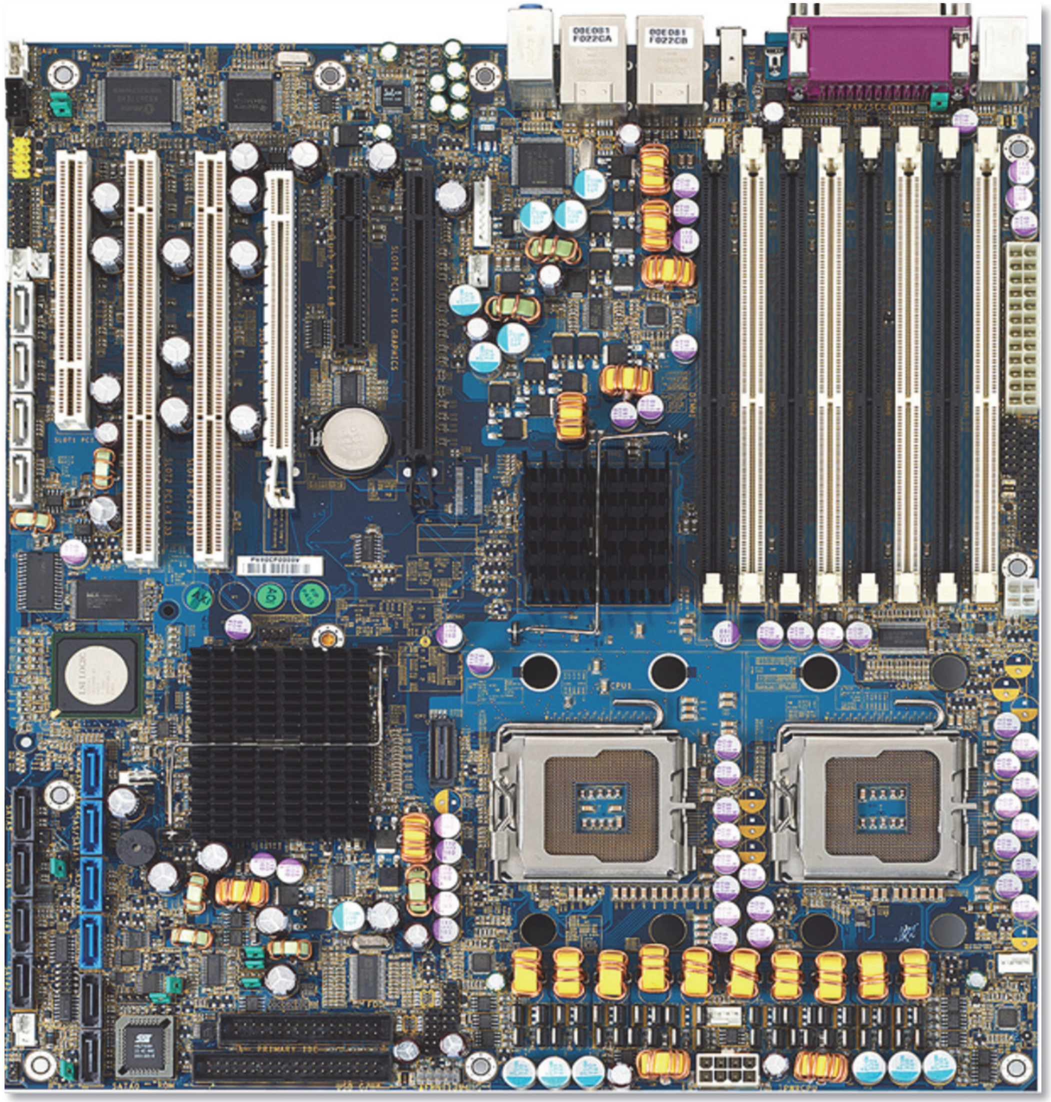
Anatomy of a cluster node



Front Side Bus architecture used in Intel Pentium chips.

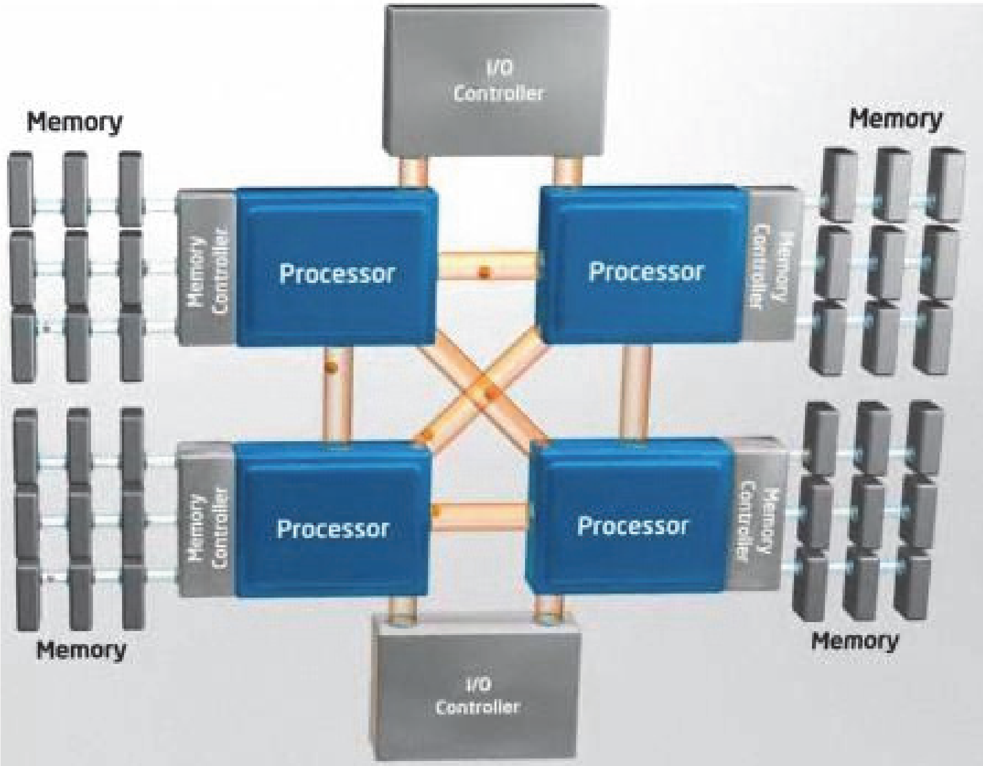


Point-to-point interconnect used in modern Intel and AMD chips.

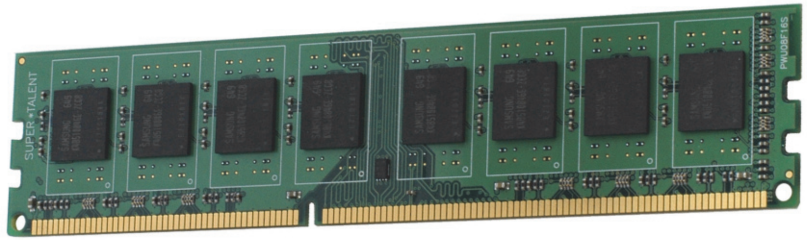


Tyan S2696 motherboard.

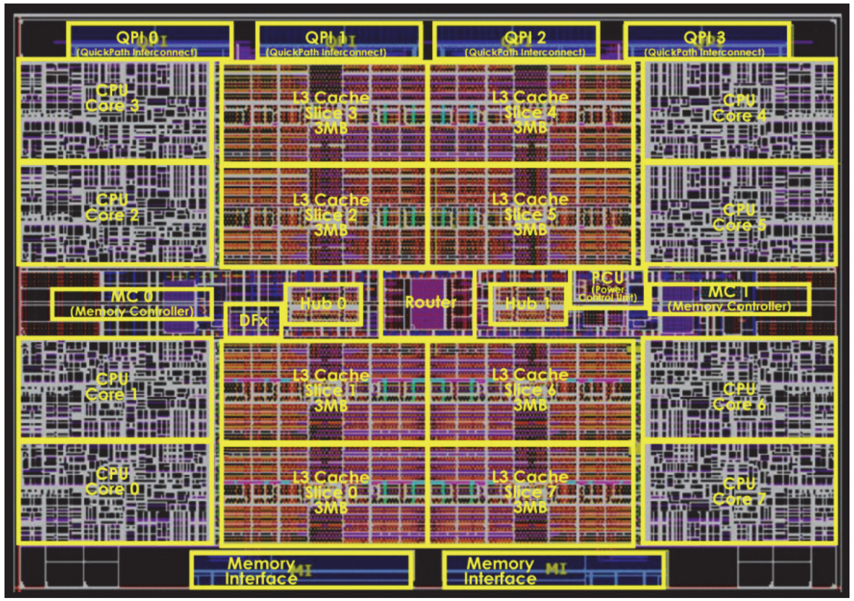
Anatomy of a cluster node



Block schematic of a quad-socket node using Intel Nehalem-EX CPUs.



DDR3 memory chip.



Physical layout of an Intel Nehalem-EX CPU.

ICH5 controller hub.

<u>Rules of thumb:</u> access to own memory is fast, access to another CPU’s memory is somewhat slower, access to another node’s memory is *very* slow.

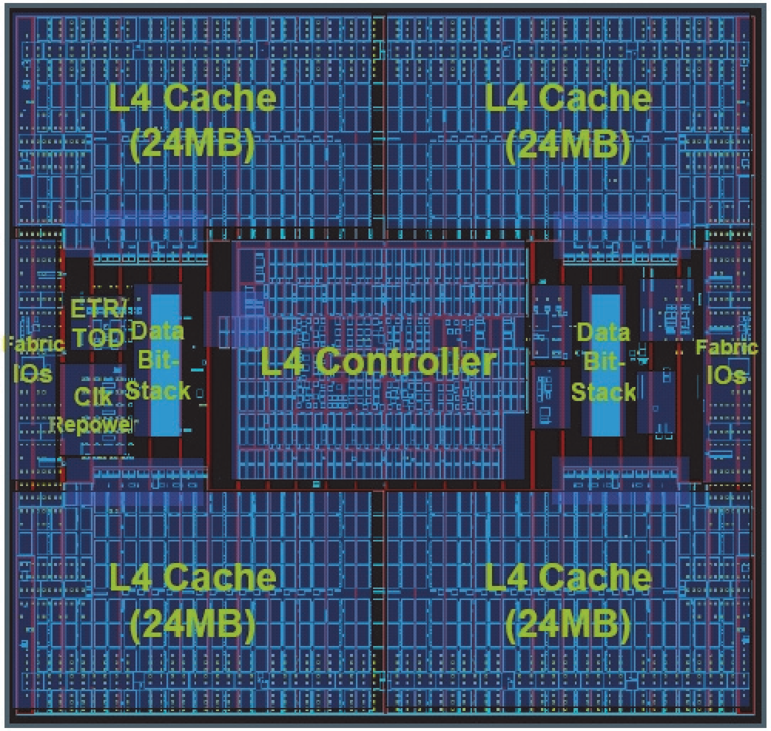
Memory bandwidth and latency

Latency is the time it takes for the data requested by the CPU to start arriving.

Bandwidth is the rate at which the data arrives.

Location	Latency	Bandwidth
CPU registers	1 clock	-
L1 cache	1 clock	150 GB/s
L2 cache	3 clocks	50 GB/s
L3/L4 cache	10 clocks	50 GB/s
Own RAM	100 clocks	10 GB/s
NUMA RAM	150 clocks	10 GB/s
Other node RAM	10k clocks	1 GB/s
Flash storage	10k clocks	1 GB/s
Magnetic storage	10k clocks	0.1 GB/s

(approximate data for an Intel Nehalem system)



Physical layout of IBM zEnterprise 196 hub chip.

Smaller programs and datasets are processed faster because they fit into a higher level cache.

Random hits to the main RAM can make a program very slow.

CPU speed and power consumption

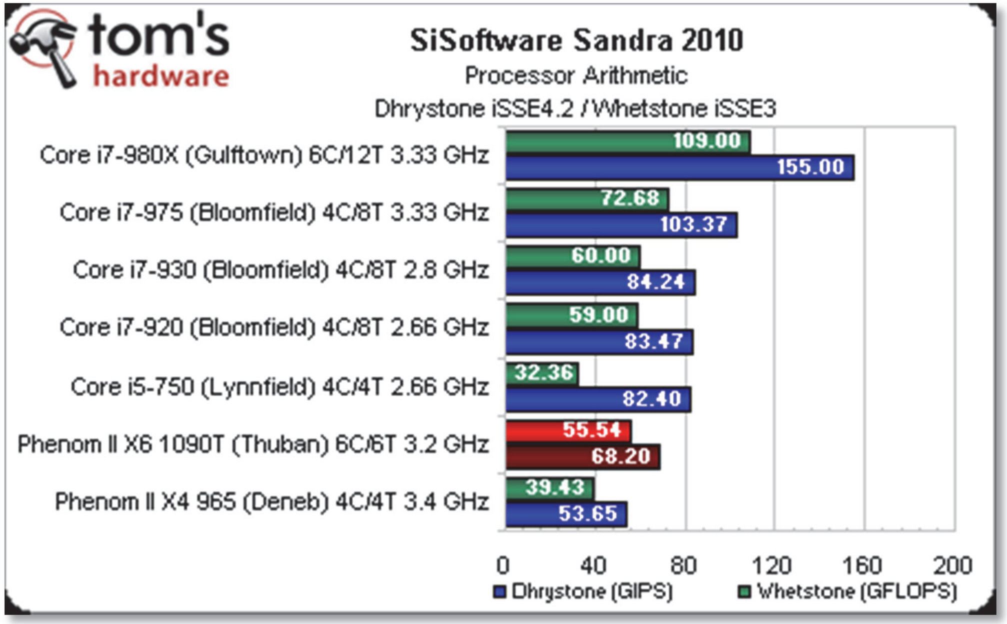
The two often used performance characteristics of a CPU are:

GIPS (billion instructions per second) – how many machine code instructions a processor can execute per second. This metric is very architecture-specific.

GFLOPS (billion floating-point operations per second) – how many single- or double-precision multiplications a processor can perform per second. Scientific calculations are nearly always performed in double precision.

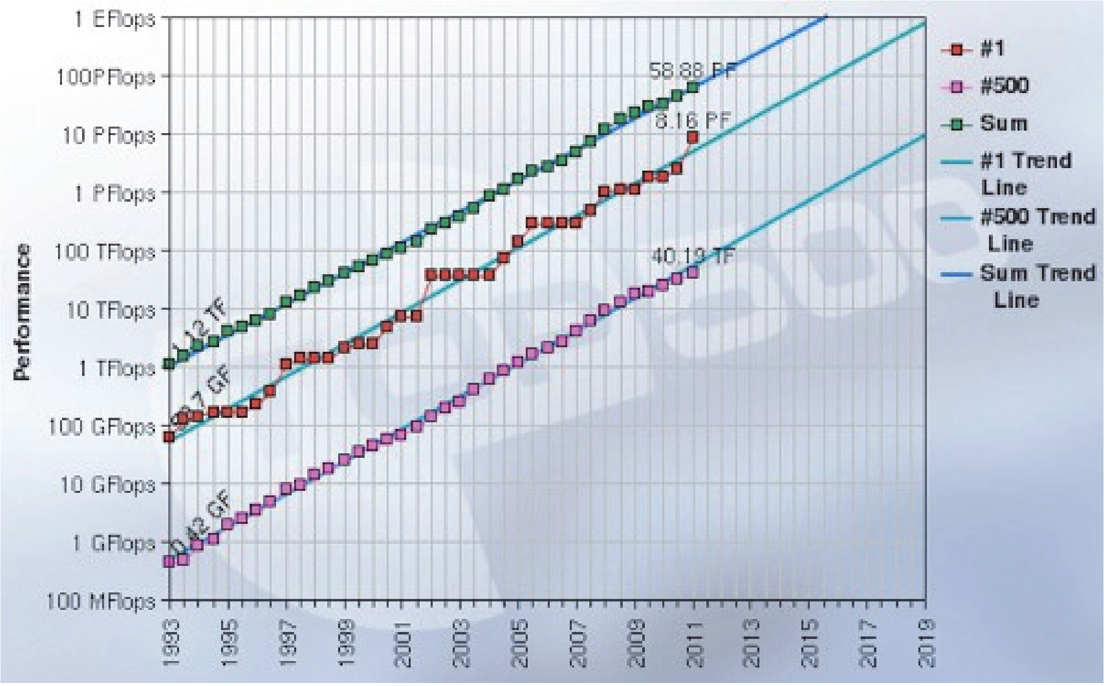
Power efficiency (GFLOPS supplied per watt consumed) has become a major issue around 2004.

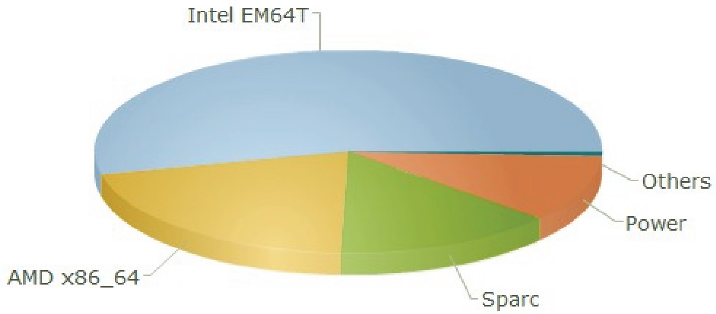
The annual electricity and cooling bill of the Oxford Supercomputing Centre runs into six digits.

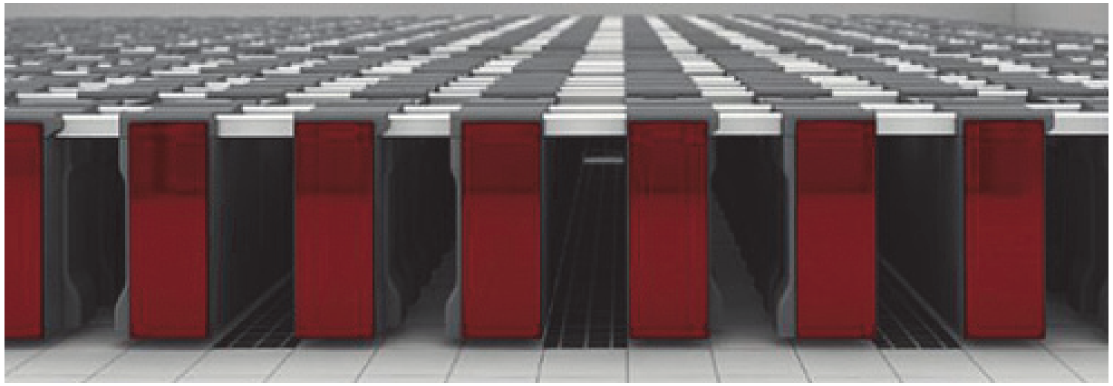


CPU	Year	TDP
P-MMX 233	1995	17
P-2 450 MHz	1998	27
P-3 1.1 GHz	2001	40
P-4 HT 672	2004	115
Core i7-980X	2009	130

Time evolution of computing power







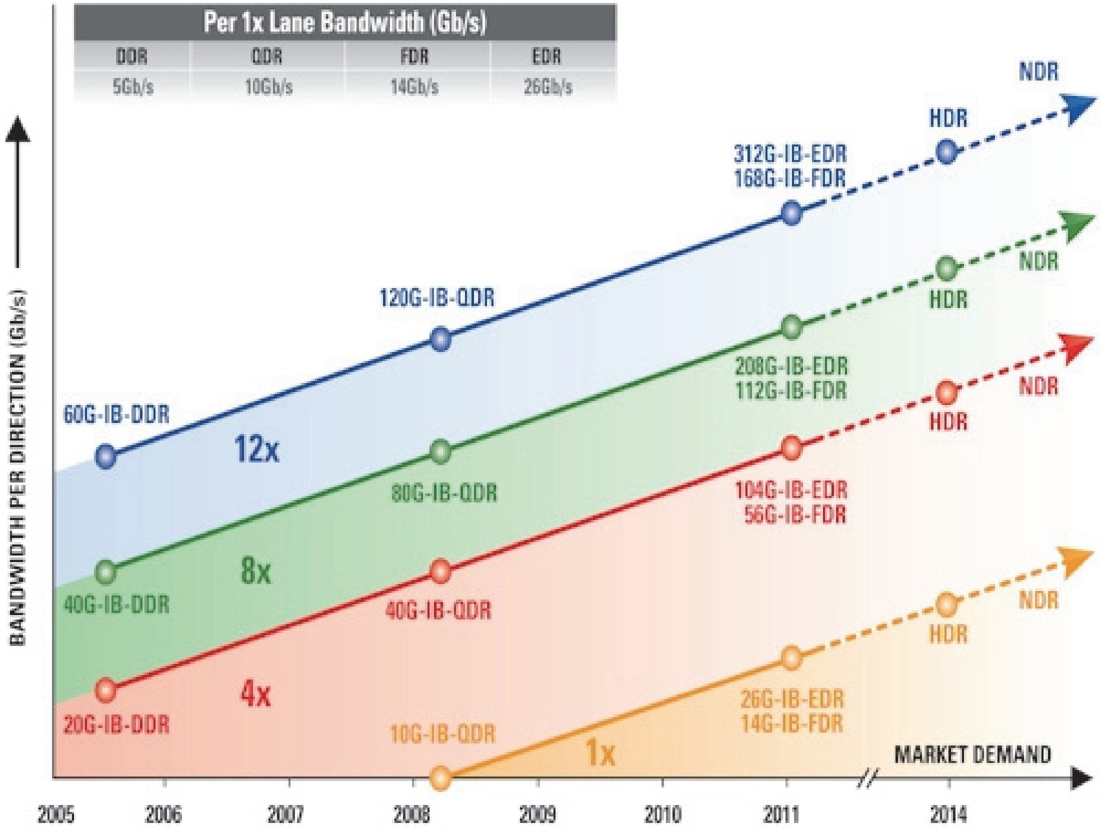
The amount of CPU power available per dollar doubles roughly every 18 months, and has been for the last 40 years.

Numerous predictions to the effect that this scaling (known as “Moore’s Law” after a co-founder of Intel Corporation) would stop at some point have not been fulfilled.

At the time of writing the scaling remains exponential.

Network interconnects

Infiniband roadmap as of 2010

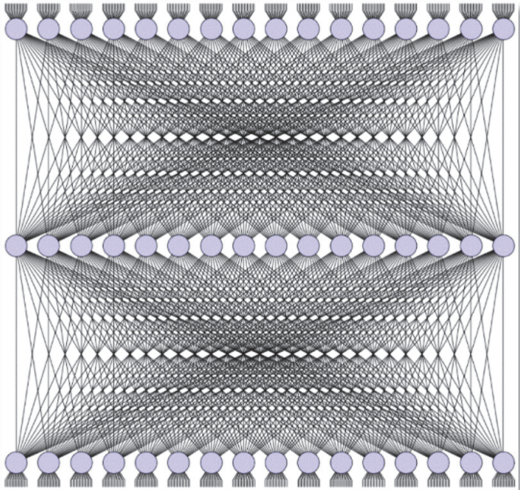


Name and Year	Bandwidth
10BASE Ethernet (1990)	10 Mbps
100BASE Ethernet (1995)	100 Mbps
1000BASE Ethernet (1999)	1 Gbps
Myrinet-10G (2005)	10 Gbps
10GBASE Ethernet (2006)	10 Gbit/s
Infiniband 12X QDR (2009)	96 Gbit/s
100GBASE Ethernet (2010)	100 Gbit/s

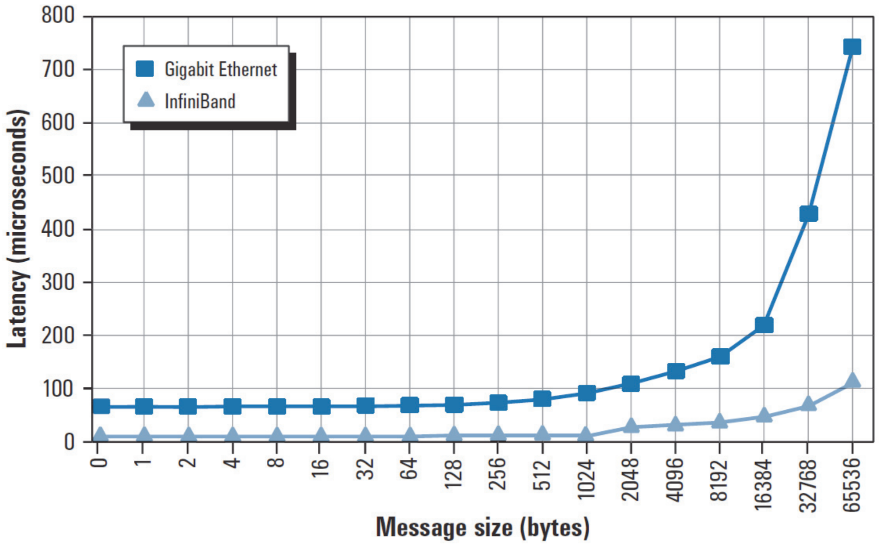
The protocols may be implemented over copper cable or optical fibre. Noise and impedance losses mean that at larger bitrates make the maximum length of the cable gets shorter. Latencies are typically in the 100 ns range.

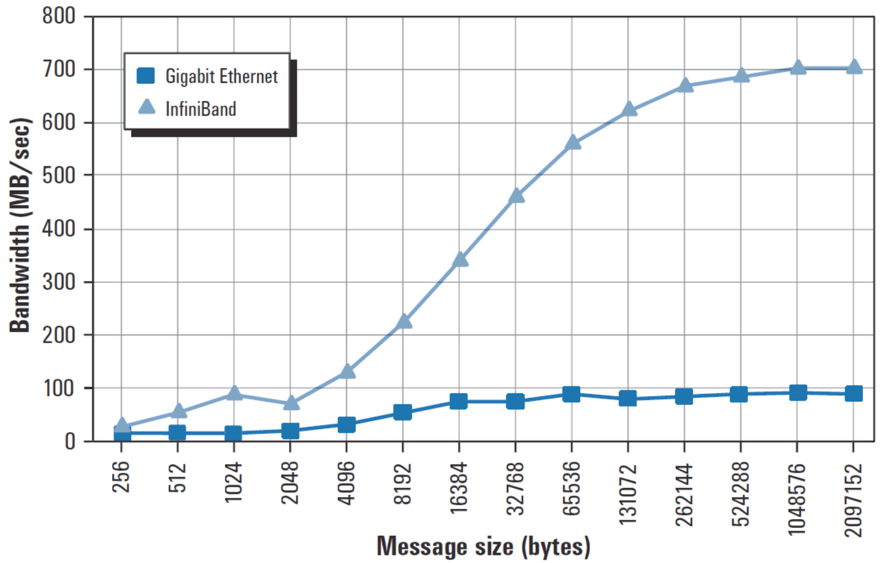
Cluster interconnect bandwidth and latency

256-port Myrinet switch.



Myrinet network topology.

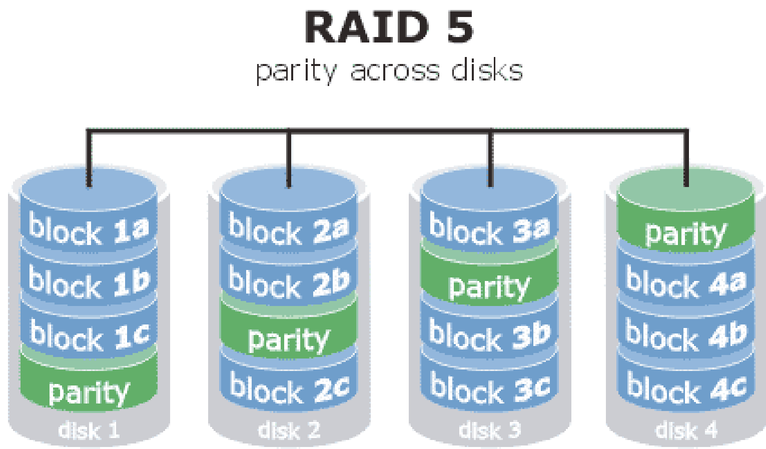


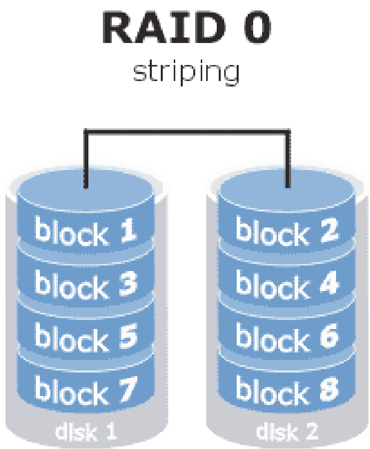


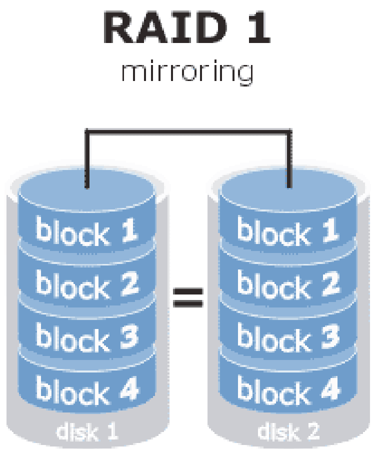
Note the microsecond latencies – 1 microsecond is equivalent to about 2k clocks on a Nehalem CPU. The bandwidth is also far below the RAM or cache bandwidth. For this reason, memory intensive calculations with a lot of data dependencies are often difficult to run on cluster architectures.

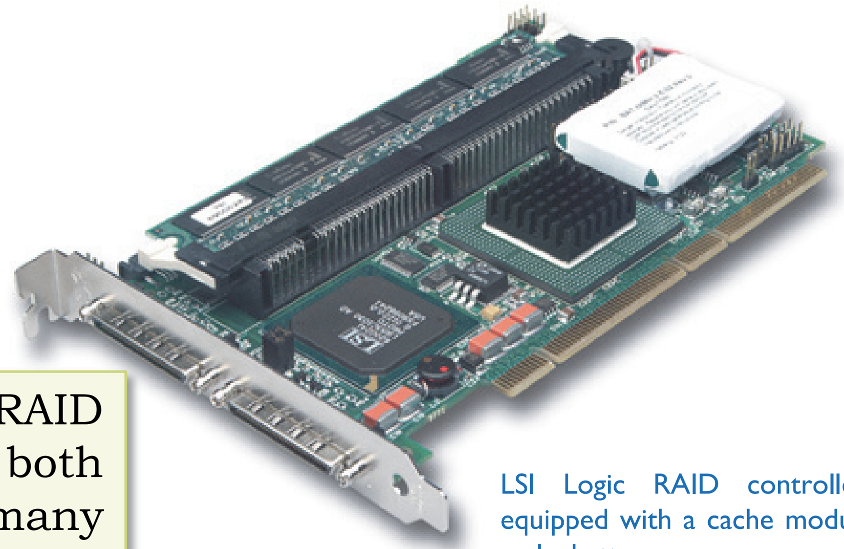
Magnetic storage and RAID arrays

The size (around 2 TB) and I/O bandwidth (around 150 MB/s) of an individual hard drive are insufficient for most supercomputing applications. A single hard drive is also lacking a fault tolerance mechanism. These issues are addressed using RAID (redundant array of independent disks).









LSI Logic RAID controller, equipped with a cache module and a battery.

RAID0 does not provide any fault tolerance. RAID 1 is fault-tolerant, but is not faster. RAID5 is both fault-tolerant and faster (particularly with many disks) than a single hard drive.

Filesystem performance hierarchy

Name	Latency	Bandwidth	Typical usage
RAM disk	nanoseconds	10 GB/s	A segment of RAM that looks like a file system to the OS, not often used.
scratch / swap	microseconds to milliseconds	0.1-1 GB/s	Dedicated filesystem, usually very fast, for the storage of intermediate calculation results.
local filesystem	milliseconds	0.1 GB/s	Stores operating system files, executables, system logs, etc.
network filesystem	milliseconds	0.1 GB/s	User home directories located on some external hardware that is accessed via a network.
backup filesystem	seconds	0.1 GB/s	Infrequently accessed incremental backup areas.

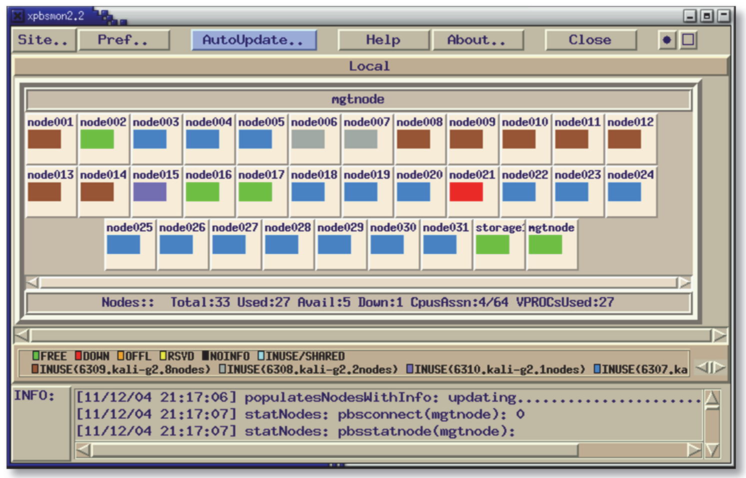
Warning: never use local and networked filesystems for scratch storage. You run the risk of crippling the entire cluster and making lots of people really, really angry.

Scaling of elementary operations

Operation	CPU Cost / FLOPs	Memory cost	Sparse cost	Parallel execution
Matrix-matrix multiplication	$O(n^3)$	$O(nnz)$	LOWER	EASY
Matrix-vector multiplication	$O(n^2)$	$O(nnz)$	LOWER	EASY
exp(matrix)-vector multiplication	$O(n^2)$	$O(nnz)$	LOWER	EASY
Matrix exponentiation	$O(n^3)$	$O(nnz)$	LOWER	EASY
Most matrix factorizations	$O(n^3)$	$O(n^3)$	SAME	HARD

Matrix factorizations (LU, Householder, SVD, diagonalization, *etc.*) are very expensive, do not benefit from sparse storage and are difficult to parallelize.

Schedulers



The job of the scheduler is to allocate jobs to specific nodes and keep track of their progress as well as to perform various administrative tasks (job migration, scratch storage clean-up, billing, *etc.*). OSC systems use PBS scheduler.

Operating systems

The Top500 list of supercomputers is dominated by Linux clusters:

Operating System	Count	Share %	Rmax Sum (GF)	Rpeak Sum (GF)	Processor Sum
Linux	405	81.00 %	22370029	36176292	3186754
Super-UX	1	0.20 %	122400	131072	1280
AIX	19	3.80 %	1305305	1700704	100176
Cell OS	1	0.20 %	35480	38836	3650
SuSE Linux Enterprise Server 9	4	0.80 %	258017	393332	59504
CNK/SLES 9	15	3.00 %	3048504	3697868	1146880
SUSE Linux	1	0.20 %	274800	308283	26304
Redhat Linux	4	0.80 %	361590	446020	48800
RedHat Enterprise 4	3	0.60 %	109580	151341	14736
UNICOS/SUSE Linux	1	0.20 %	35200	42598	8192
SUSE Linux Enterprise Server 10	4	0.80 %	157080	192640	20952
SLES10 + SGI ProPack 5	14	2.80 %	1255157	1416601	126720
UNICOS/lc	1	0.20 %	174083	208435	22656
CNL	11	2.20 %	1298028	1662396	182033
Windows HPC 2008	5	1.00 %	412590	509350	59072
RedHat Enterprise 5	2	0.40 %	129120	139795	11928
CentOS	7	1.40 %	948610	1102684	96720
Open Solaris	2	0.40 %	139110	152247	15104
Totals	**500**	**100%**	**32434683.70**	**48470495.53**	**5131461**

The choice of language and platform

<u>Linux / C++</u>	
Coding speed:	very slow
Runtime speed:	very fast
TTI (Spinach):	unrealistic
Entry barrier:	unrealistic
Maintenance:	difficult
License costs:	modest to none

<u>Windows / Matlab</u>	
Coding speed:	very fast
Runtime speed:	fast
TTI (Spinach):	six months
Entry barrier:	none
Maintenance:	easy
License costs:	£1,000+

<u>FEC of brain time</u>: between £30 and £100 per hour, depending on qualification.

<u>FEC of CPU time</u>: £0.06 per core-hour, improves exponentially with time.

Major sins and virtues

Things you should not be doing

1. Never run on the head node of the cluster – the head node is busy scheduling the entire cluster, it may only be used as a gateway and for code compilation.
2. Never use the home filesystem for scratch storage – this would overload the service network and slow down the entire system. The dedicated scratch storage must always be used for storing intermediate results.
3. Never bypass the scheduler – direct node logins are for debugging purposes only, they should not be used for calculations.

Good practices

1. Mind the latencies – sequential or streaming access to any resource (disk, memory, CPU, GPU) is orders of magnitude faster than random access.
2. Always exit the software package that you do not use – this would release the license back into the license pool. This is particularly important for Matlab.
3. Always use platform-specific libraries – for many matrix operations, Intel's MKL or AMD's ACML are orders of magnitude faster than hand-coded *for* loops.
4. Specify the resources you are going to use – the scheduler defaults are pessimistic and may slow down the progression of your job in the queue.