It's Magic!

SourceMage GNU/Linux
as a
High Performance Computing Cluster OS

Zoran Constantinescu & Jörg Cassens
Norwegian University of
Science and Technology (NTNU)

overview

- HPC - High Performance Computing
- Beowulf Clusters
- Linux
  - distributions: binary & source based
  - SourceMage GNU/Linux
- ClustIS project at NTNU
  - technical details
  - cluster software
  - parallel, distributed, scientific libraries
  - application areas

HPC * what?

- method of solving large computational problems, sharing the workload between multiple computers
- is used to provide solutions to problems that
  - require significant computational power
  - need to access/process very large amounts of data
  - get more accurate and reliable results
  - tackle computing problems which were previously either technically or economically not realisable.
- more *computing power* and *hardware resources* than is normally available on one desktop

HPC * where?

- typical application areas
  - engineering and scientific numerical simulations
    - Computational Fluid Dynamics (CFD)
    - Molecular Dynamics (MD)
  - predictive modelling and simulations
    - financial modelling
    - weather forecasting
  - optimization problems
  - information processing
    - data mining
    - data visualization

HPC requirements

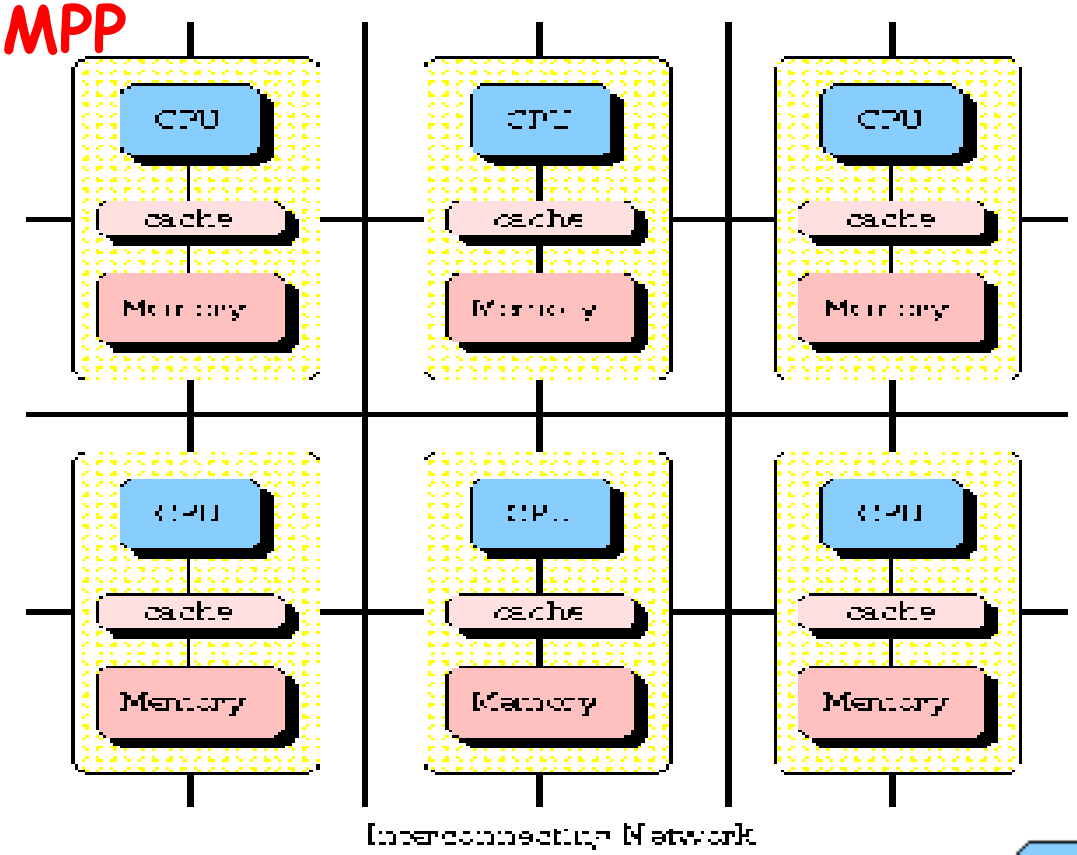
- Requirements are very much depending on application area
- Common designators:
  - Computing power (CPU)
  - Memory
  - Disk space
  - Interconnect
- “Enough” from all of these for the application
  - Data Mining might not need fast interconnect
  - Simulations might not need large disks

HPC * how?

- advances in computing technology
- parallel processing - use lots of CPUs in parallel
- Shared Memory Systems
  - Symmetric Multi-Processing - SMP
  - Parallel Vector Processing - PVP
- Distributed Memory Systems
  - Massively Parallel Processing - MPP
  - Cluster Computing
- Distributed Shared Memory Systems - DSM
  - Non-Uniform Memory Access - NUMA

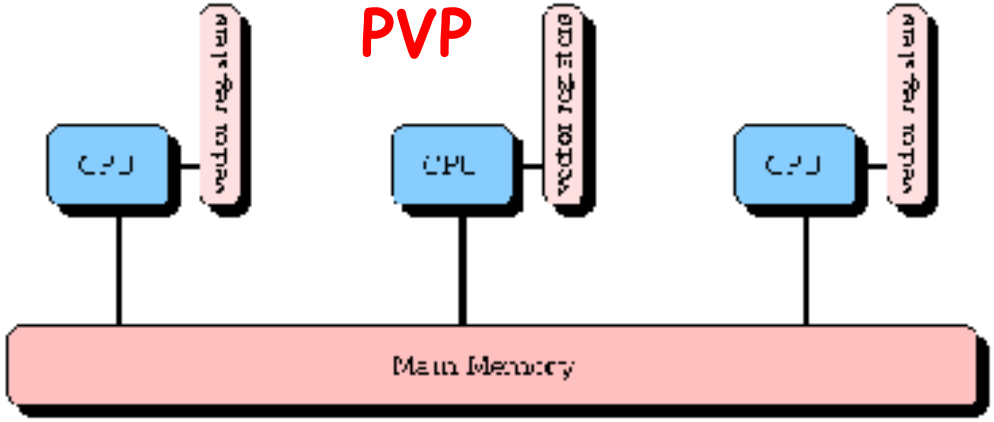
HPC * types

MPP



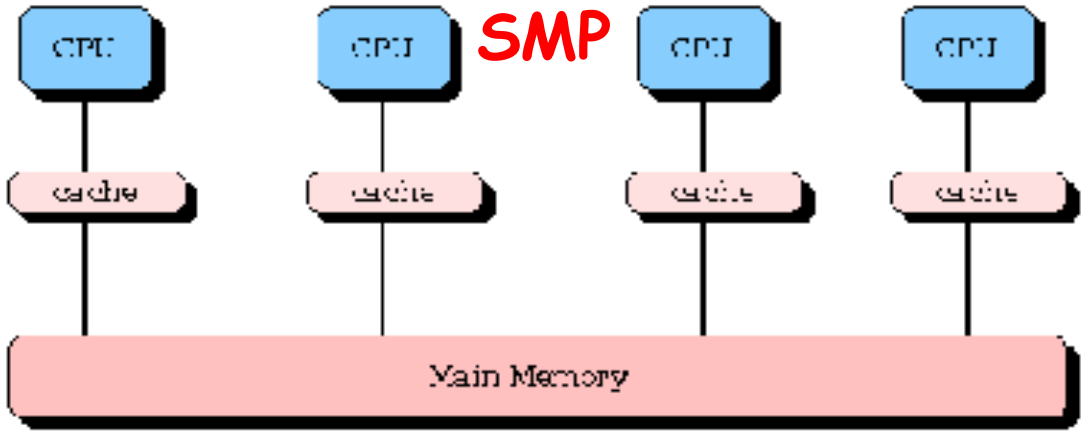
Cray T3E, IBM SP/2, Hitachi SR2000

PVP



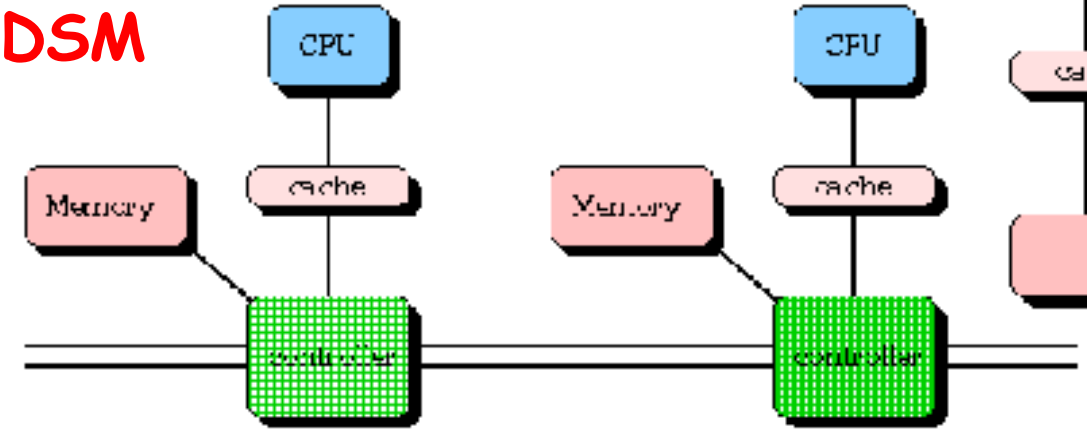
Cray T90, NEC SX-4

SMP



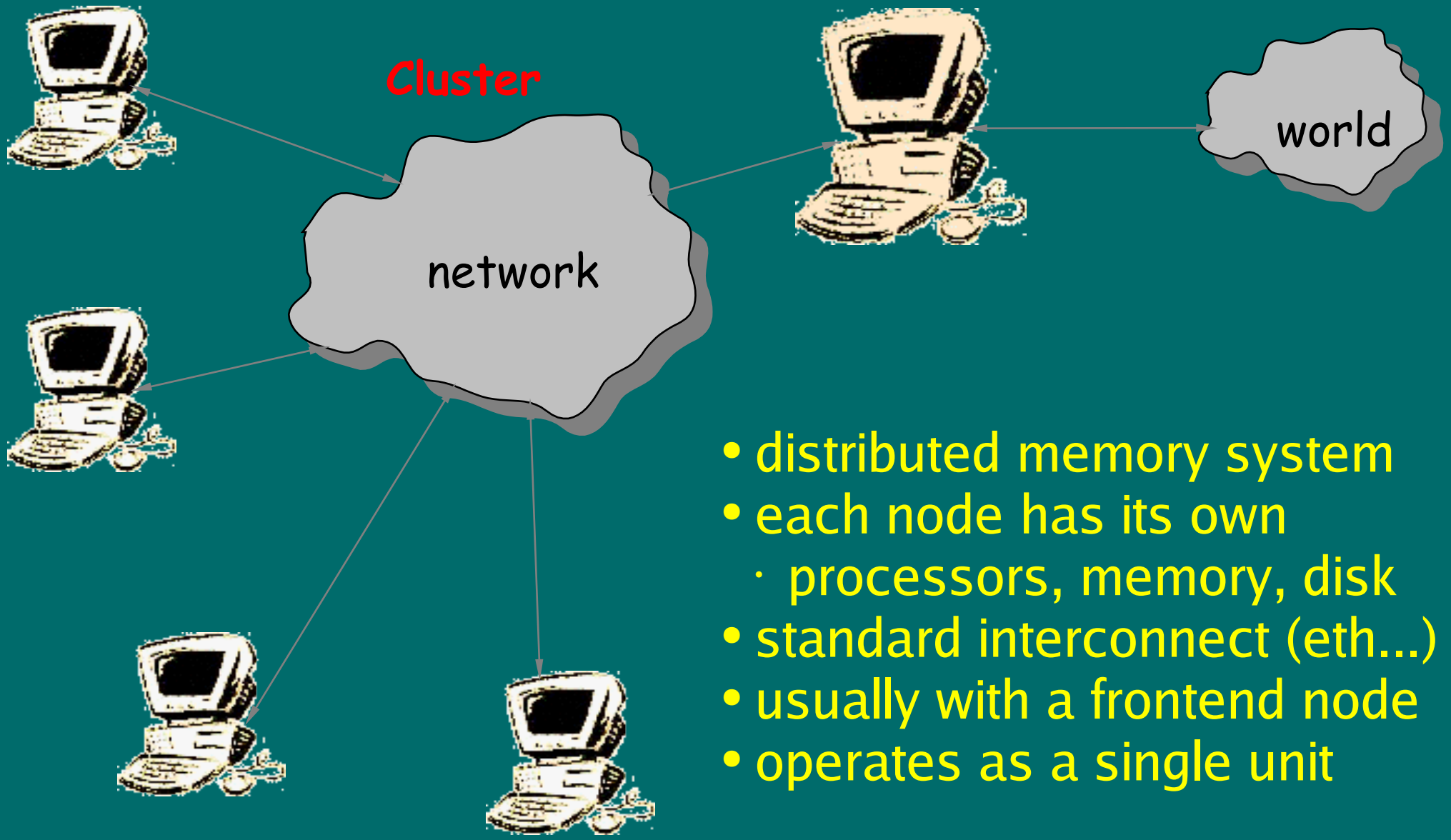
Sun Enterprise Server
DEC Alpha Servers
SGI Challenge

DSM



Cray/SGI Origin 2000

HPC clusters



- distributed memory system
- each node has its own
  - processors, memory, disk
- standard interconnect (eth...)
- usually with a frontend node
- operates as a single unit

parallel computers vs. clusters

- pluses
  - better interconnect between processors (high-bandwidth, low-latency)
  - service and support
  - long experience
- minuses
  - limited scalability
  - just a few vendors
  - more expensive

- pluses
  - cheaper
  - well known h/w standards
  - no single h/w vendor
  - very flexible
  - use open source s/w
- minuses
  - space
  - service & support
  - not that easy to install
  - hard to optimize

clusters in Top500

DETAILS **TOP500 List for June 2003** EXPLANATION OF THE FIELDS

Rank	Manufacturer Computer/Procs	Rmax Rpeak	Installation Site Country/Year
1	**NEC** Earth-Simulator/ 5120	**35860.00** 40960.00	Earth Simulator Center Japan /2002
2	**Hewlett-Packard** ASCI Q - AlphaServer SC ES45/1.25 GHz/ 8192	**13880.00** 20480.00	Los Alamos National Laboratory USA/2002
3	**Linux Networx** MCR Linux Cluster Xeon 2.4 GHz - Quadrics/ 2304	**7634.00** 11060.00	Lawrence Livermore National Laboratory USA/2002
4	**IBM** ASCI White, SP Power3 375 MHz/ 8192	**7304.00** 12288.00	Lawrence Livermore National Laboratory USA/2000
5	**IBM** SP Power3 375 MHz 16 way/ 6656	**7304.00** 9984.00	NERSC/LBNL USA/2002
6	**IBM** xSeries Cluster Xeon 2.4 GHz - Quadrics/ 1920	**6586.00** 9216.00	Lawrence Liv Laboratory USA/2003
7	**Fujitsu** PRIMEPOWER HPC2500 (1.3 GHz) 2304	**5406.00** 11980.00	National Aero Japan Japan/2002
8	**Hewlett-Packard** rx2600 Itanium2 1 GHz Cluster - Quadrics/ 1540	**4881.00** 6160.00	Pacific North Laboratory USA/2003

TOP500 SUPERCOMPUTER SITES

	Count	Share
MPP	211	42.2 %
Cluster	149	29.8 %
Constellations	139	27.8 %
SMP	1	0.2 %
Total	**500**	**100 %**

Beowulf clusters

- commodity-off-the-shelf (COTS) hardware
  - nodes are usually standard desktop PCs
  - good performance CPUs
  - network: Fast/Giga Ethernet, Myrinet, SCI
  - disk: EIDE, SCSI
- free/open source software
  - Linux as operating system (or FreeBSD)
  - parallel communication software (MPI, PVM)
  - parallel numerical software
- for massive computational requirements

driving technologies

- cheap hardware
  - widely available, very good performance
- easy to port existing applications (UNIX)
- do-it-yourself cluster computing - *complete control*
  - (almost) anybody can build such a cluster
  - fits well for many academic and research environs.
- computer science education (Linux, parallel prg)
- free/open source software
  - operating systems - Linux, FreeBSD
  - a lot of high-quality parallel sotware out there

advantages

- learning to built and run a Beowulf cluster
  - *is an investment*
- learning the peculiarities of a specific vendor
  - *enslaves you to that vendor*

- Beowulf clusters are <u>not</u> supercomputers
  - one can build a Beowulf that is *big enough* to attract the interest of supercomputer users

cheap hardware

- CPUs (32bit & 64bit)
  - Intel P4 @3.2GHz, AMD Athlon XP3200+ @2.2GHz
  - Intel Itanium, AMD Opteron
  - RISC (IBM/Motorola PowerPC)
- memory - high speed DDR RAM
- fast disks (EIDE, Ultra Fast SCSI)
- (standard) high-speed Ethernet network
  - 100 MBps, 1 GBps, 10 GBps

cheap software

- free/open source software
  - free as in *speech* and in *beer*
- free commercial software
  - free as in *beer*
- GNU/Linux OS
- GNU software
  - compilers
  - utilities
- applications at sourceforge.net
- numerical libraries at netlib.org

other clusters

- load balancing clusters
  - large web/ftp servers, streaming media servers
  - large data bases or search engines (e.g. Google)
  - terminal servers (e.g. Linux Virtual Server)
- high availability clusters
  - for reliability and availability of resources
  - always on: 24 hours a day, 7 days a week
  - heartbeat, automatic failover
- computational clusters
  - mainly for computational intensive problems

Linux in HPC

- why Linux?
  - it's free & source code is availability
  - maturity and robustness
  - very good performance
  - 'best' technical support
  - tools and applications
  - wide user acceptance
- other alternatives...
  - FreeBSD, Windows, Solaris

Linux distros

- binary
  - certain number of precompiled s/w packages
- source based
  - compiled at install time from latest source code

	binary	source
bytes to download	less	more
time to compile	short	long
install time	short	long
latest software versions	no	yes
compilation logs	no	yes
optimized binaries	maybe	yes
architecture specific binaries	maybe	yes

binary distribution

- limited support for clusters
  - by providing some of the tools and libraries
- ready solutions from cluster system suppliers
  - based on existing distros (RedHat, SuSE,YellowDog)
  - focus is more on stability
  - older versions of software
  - good for engineering domains
- research systems need more 'bleeding edge'
  - not easy to combine a binary distro with newer s/w (complex dependencies, library problems)

source based distribution

- packages are compiled at install time
- from the latest original source code distributions
- no more outdated packages
- compiled for the architecture and optimizations
- installed, tracked, and archived for easy removal
- admins have more control over the system
- enhanced performance and customization
- more time needed to install :-(
- latest versions have 'less' bugs ;)

SourceMage GNU/Linux

- simple yet powerful src.based distribution
- *sorcery*: the package management system
  - both command line and menu driven commands
  - powerful bash scripts
  - easy to automate operations
- *grimoire*: the package descriptions
  - contains 'spells' with instructions for downloading, compiling, configuring and installing each package
  - most spells install automagically
  - some require user interaction (e.g. kernel config)

ClustIS @NTNU

- NTNU
  - Norwegian University of Science and Techn.
  - largest technical university in Norway
- ClustIS
  - Beowulf type computational cluster from the Division of Intelligent Systems (D/I/S)
- why needed?
  - doctoral research (simulations)
  - HPC/parallel computing teaching
  - increase penguin population at NTNU

DIIS
Division of Intelligent Systems

ClustIS pics

ClustIS website

Location: http://clustis/

Feedback: zrm jmt

ClustIS - Intelligent Systems Cluster

Welcome to the computational cluster ClustIS of the Division of Intelligent Systems (DIS) and the Division of Complex Computing Systems (KDS).

News

25.03.2003	We have set up the ClustIS Wiki FAQ. If you have a problem related to ClustIS, this is the first place to go...
11.02.2002	No news is usually good news... but today we can celebrate the first anniversary of the DIS cluster! Which is good news. Happy birthday, ClustIS!
29.11.2002	Some of the nodes went down because of a power outage. ClustIS is not using an UPS, so problems with power supply can affect the cluster. We don't think it is important to have the nodes available 24/7, but we are considering adding an UPS at for the master node.
24.10.2002	We gave another talk about ClustIS in the internal seminar series of the AI and Learning group at DIS. Slides can be found here.
14.10.2002	Returned some of the temporary nodes, master was upgraded to a dual Athlon, added a new storage node...

Archived older news.

Documentation

(Access to some of the documentation is limited.)

- ClustIS Wiki FAQ (important!)
- Sorcerer GNU/Linux documentation
- 3com switch reference guide
- Apache documentation; mod_ssl manual
- Squid configuration manual

ClustIS status

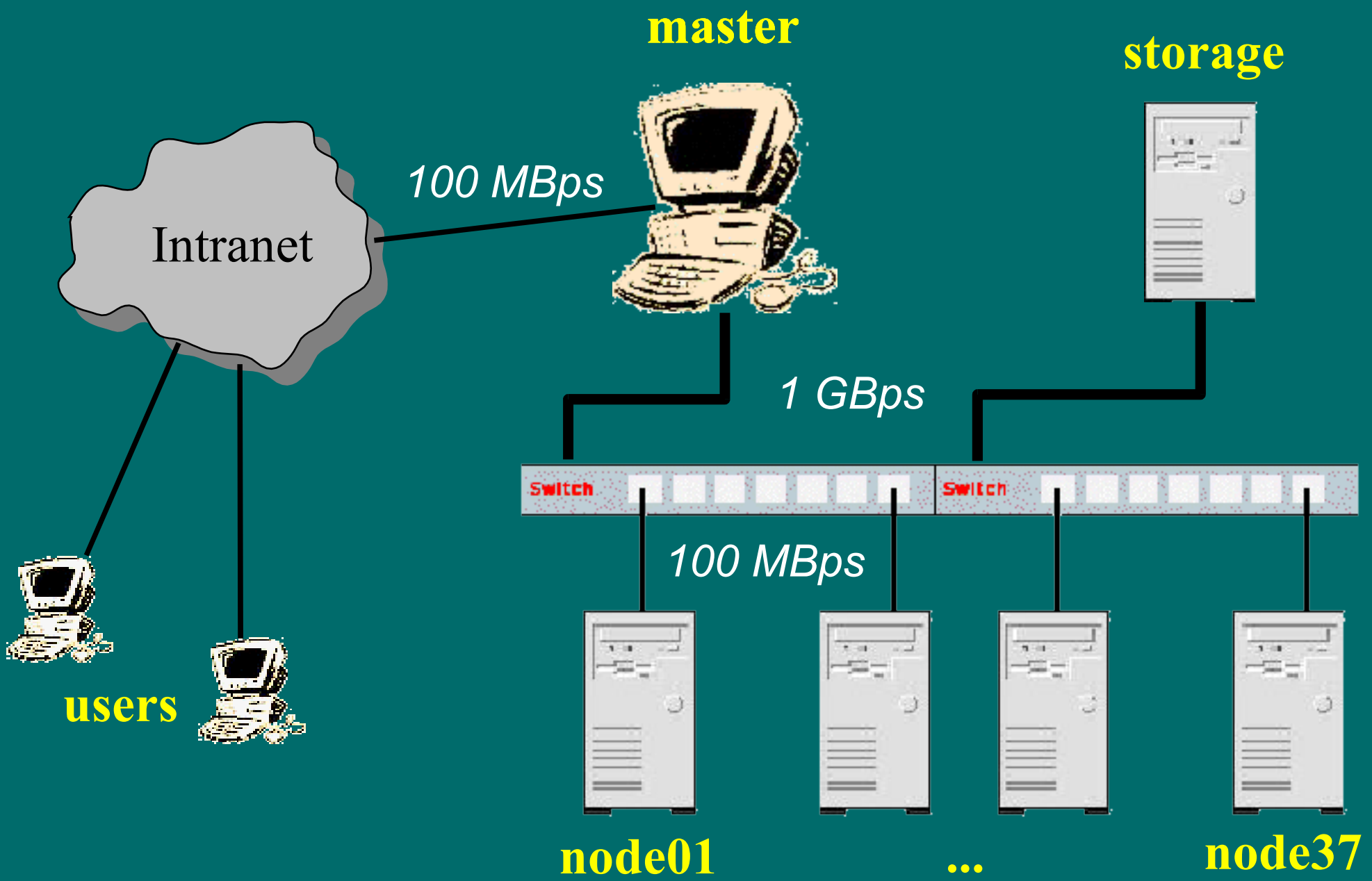
ClustIS history

- started with 1+28 nodes in feb. 2002
  - cheap desktop PCs
- problems with the cooling (16 sq.m no-win.room)
- problems with the electrical power consumption
- added 8 mode nodes in may 2002
  - rack mounted PCs
- acquired a new dual CPU master node
- old master became storage node
- reboot only when power shortage or kernel upgr.

ClustIS hardware

- quite dynamic - between 39 and 43 nodes ;-)
- typical desktop PCs with 1-2 GByte RAM
- 1 master node *(dual MP2100+ @1.66 GHz)*
  - frontend node, applications, compilers, nfs server
- 1 storage node
  - nfs file server
- 28 comp. nodes *(XP1700+ @1.46 GHz)*
  - *40 GB IDE disk, 100 MBps net, no floppy/cdrom*
- 8 comp. nodes *(MP1600+ @1.4 GHz)*
  - *18 GB SCSI disk, 2x100 MBps net, floppy+cdrom*

ClustIS architecture



ClustIS details

- bootup procedure
  - the nodes network-boot using the PXE protocol
  - the master (dhcp+tftp server)
    - assigns the corresponding IP addresses to each node
    - delivers the pxelinux netloader
    - delivers the right kernel for booting (SMP, SCSI support)
  - if dhcp server is down, nodes will boot from disk
- two bootup modes:
  - install and disk mode
  - the mode is selected by changing the kernel arguments when booting the node

ClustIS details

- install mode
  - node will mount an nfs-root file system from master
  - formats local disk with system/swap/work partitions
  - downloads a tar.gz image of the operating system
  - does local configuration based on IP (node#, etc.)
  - reboots in disk mode
- disk mode
  - normal operating mode
  - after booting the kernel, it's using the operating system from the local disk (from install)

ClustIS details

- the tar.gz image of the local operating system
  - created on the master (~130 MBytes)
  - contains an archive of a full Linux installation
  - is identical for all the nodes
  - but we can have different versions
- created by making a customized Linux install in a chroot environment on the master
- the nodes have a limited number of packages
  - no compiler, no Xwindows, no big applications
  - only base system, libraries, some applications

ClustIS software

- scheduling software for fair share of resources
  - OpenPBS
- standard software
  - GNU tools, Java, Matlab
- parallel and distributed software
  - MPI, PVM, Qadpz
- scientific libraries
  - BLAS, Lapack, PetSc
- other user installed software
  - Weka

scheduling policy

- OpenPBS
- the first 8 nodes are reserved for interactive use (for testing, compiling, short jobs)
  - rely on fairness of users
- other nodes controlled by batch scheduling system
  - different queues with different priorities (our friends highest, students lowest ;-)
- Not easy to use

other software

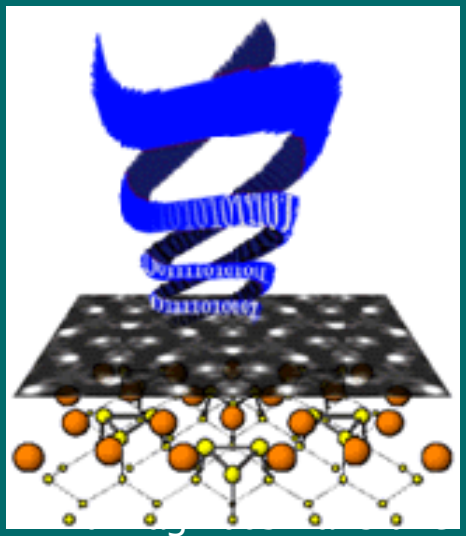
- what users put in /opt ;)
- users can get write access to /opt-tree
  (default for research fellows, students ask)
- Install software with
  - description
  - how-to
  - .profiles
- high flexibility, low security
  (not suitable for every type of cluster)

ClustIS applications

- different types of projects
  - genetic algorithms
  - protein folding
  - machine learning
  - numerical simulations
  - artificial immune systems
  - massive multiplayer game simulations
  - parallel rendering & visualization
  - teaching parallel computing

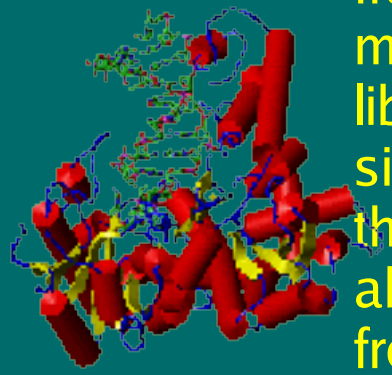
applications1

- genetic algorithms
  - develop control systems for simulated robots
  - repeated evaluation, selection and reconstruction of automatically generated solutions
  - 40 nodes in parallel for 30 minutes on average
  - 150 iterations with evaluation of 500 solutions
  - each simulation 2 runs of 1000 timesteps
  - 150 million images giving the robot a view of the world



applications2

- protein fold recognition
  - Comparative modeling methods can consistently produce reliable structural models for protein sequences with more than 25% sequence identity to proteins with known structure. However, there is a good chance that also sequences with lower sequence identity have their structural components represented in structural databases. To this end, we present a novel fragment-based method using sets of structurally similar local fragments of proteins. The approach differs from other fragment-based methods that use only single backbone fragments. Instead, we use a library of groups containing sets of sequence fragments with geometrically similar local structures and extract sequence related properties to assign these specific geometrical conformations to target sequences. We test the ability of the approach to recognize correct SCOP folds for 273 sequences from the 49 most popular folds. 49% of these sequences have the correct fold as their top prediction, while 82% have the correct fold in one of the top five predictions. Moreover, the approach shows no performance reduction on a subset of sequence targets with less than 10% sequence identity to any protein used to build the library.



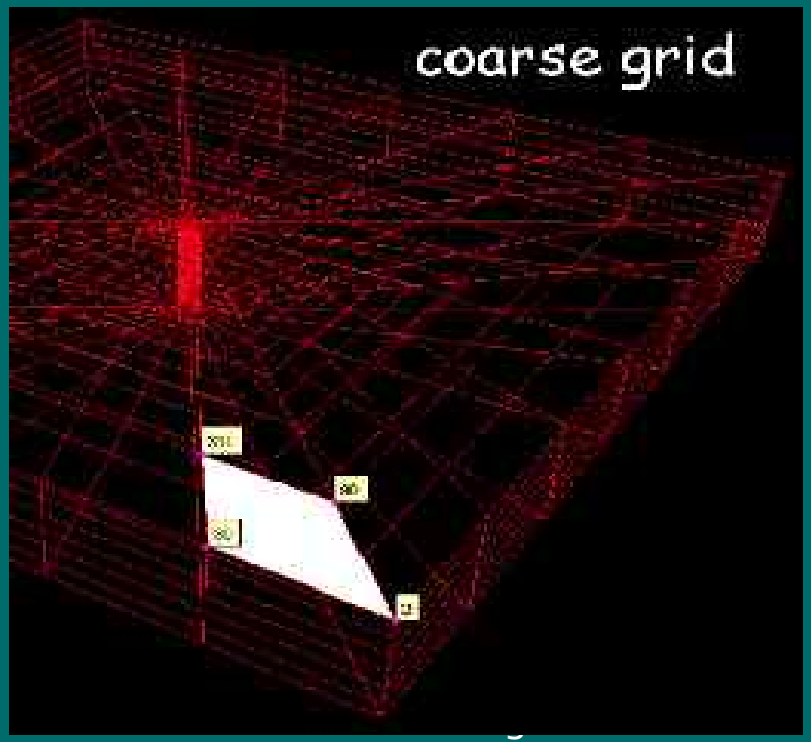
(*European Conference on Computational Biology*)

applications3

- comparing data classifier algorithms
  - modified version of the Weka tool (open source software for machine learning algorithms)
  - data pre-processing, classification, regression, clustering, association rules, and visualizationhen
  - for a new classifier: test it against as many other classifier and on as many datasets as possible
  - method used: n-fold cross validation
  - using 5-8 classifiers
    - on a PC - around 48 hours
    - on the cluster - a couple of hours

applications4

- numerical simulations
  - Computational Fluid Dynamics (CFD)
  - solver for the Navier-Stokes equations
  - using finite element methods
  - parallel version using MPI,PVM,QADPZ
  - 300 000 elements & nodes
  - simulation times:
    - 2 nodes: ~3 hours
    - 16 nodes: ~20 minutes
    - 32 nodes: ~12 minutes



applications5

- artificial immune systems simulation
  - how artificial immune systems may be used to solve real-world problems
  - e.g. to preserve security in computer networks
  - ⇒ implem. a network intrusion detection system
  - 2640 & 3360 simulations
  - training set from data of network traffic over 50 days
  - different intrusion types were simulated
  - data files generated by simulations 60 GBytes (gzip)
  - all nodes were used for a few days

applications6

- teaching
  - undergraduate course in parallel computing
    - optimization of algorithms and programs for both serial and multi-processor systems
    - optimized libraries, profiling
    - PC clusters for large computational problems
    - programming with MPI
    - parallel libraries, PetSc
  - graduate course in parallel environments and scientific computing
    - parallel programming with MPI
    - parallel image processing

applications7

- parallel rendering
  - complex 3D scenes
  - using parallel version of PovRay with PVM
- parallel visualization
  - for large scale data sets
  - 1500 x 1500 x 512 pixel X-ray scans
  - using volume rendering/ray casting with vtk
  - parallel version using MPI

applications8

- data mining in multiplayer games
  - simulation of players in multiplayer online games (e.g. Everquest, Lineage, Anarchy Online)
  - goal is to provide a highly scalable testbench for simulating player behavior
  - massively multiplayer games have >100 players
  - simulation of 160 000 players using 20 nodes
  - goal: up to 4 million simulated players (most of Norway's population ;-)

end of presentation

Q: how many ***penguins***
hide in this presentation?