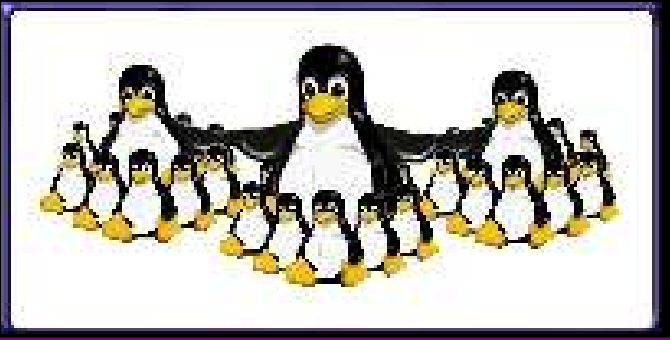
HPC with openMosix

Ninan Sajeeth Philip
St. Thomas College
Kozhencheri

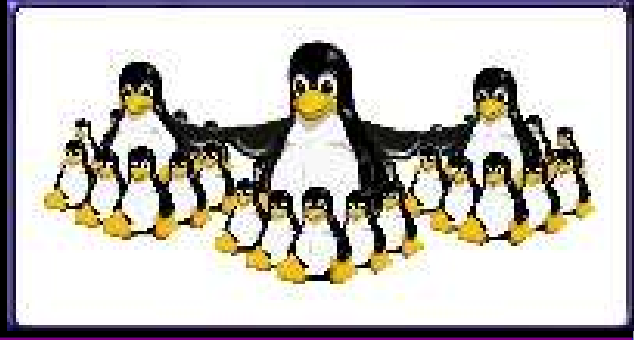
nsp@stthom.ernet.in

Acknowledgements

- ***This document uses slides and image clippings available on the web and in books on HPC. Credit is due to their original designers!***



Overview



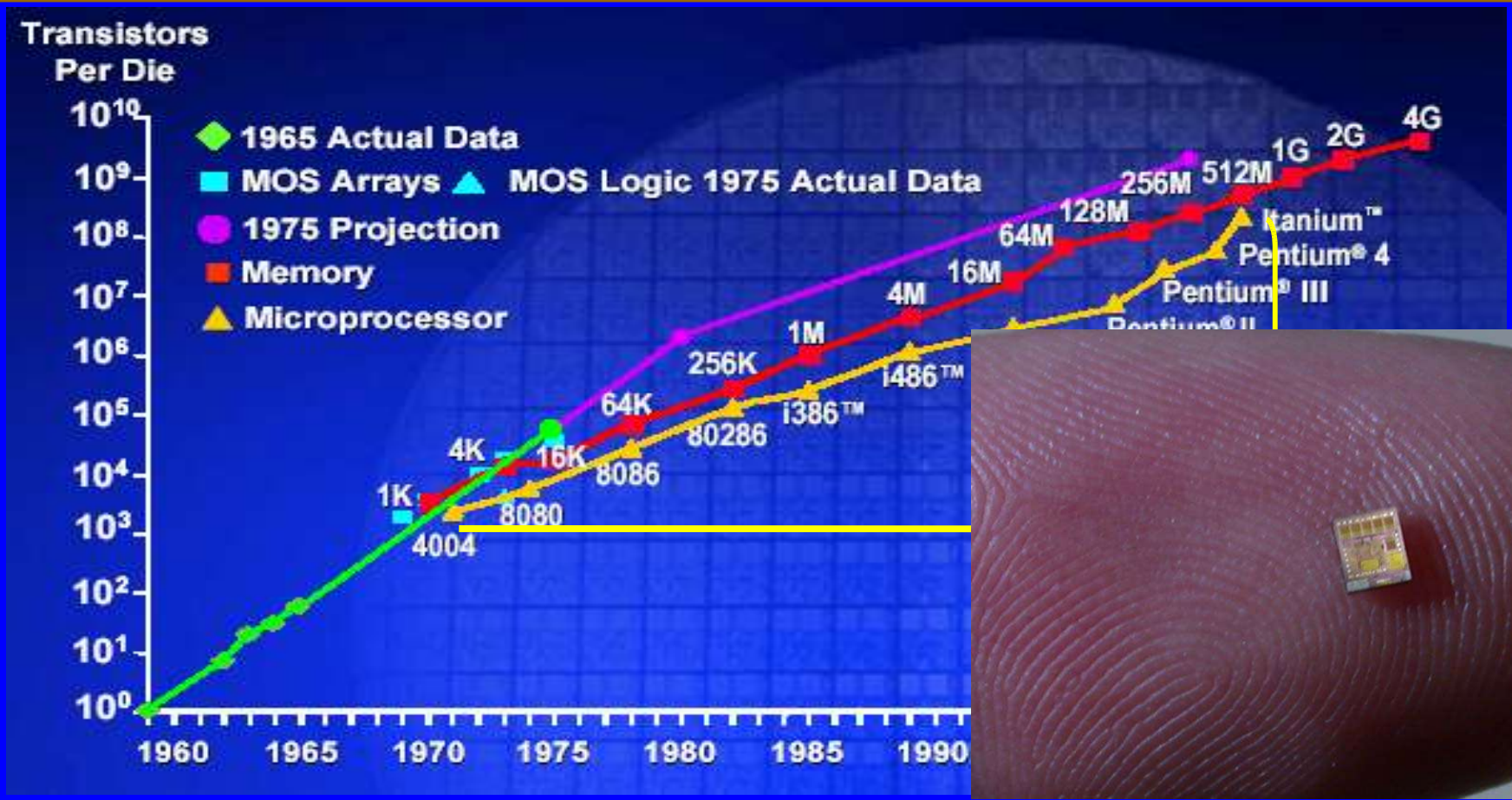
- Mosix to openMosix
- Why openMosix?
- Design Concepts
- Advantages
- Limitations

The Scenario

- We have MPI and it's pretty cool, then why we need another solution?
- Well, MPI is a specification for cluster communication and is not a solution.
- Two types of bottlenecks exists in HPC - hardware and software (OS) level.

nsp@stthom.ernet.in

Hardware limitations for HPC



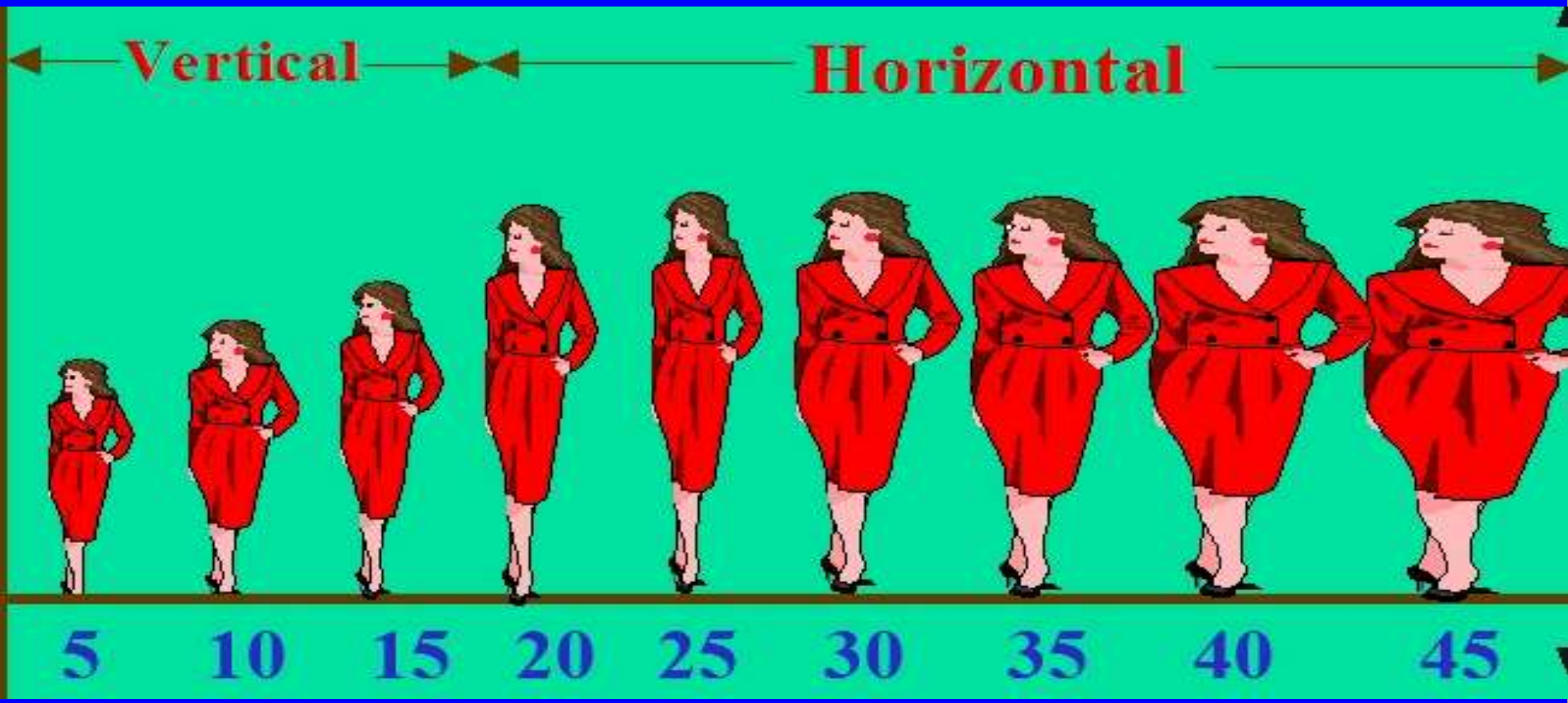
nsp@stthom.ernet.in

The Scenario

- *We are approaching the speed and size limits of the electronics*
- *Major share of possible optimization remains with software part - OS level*

nsp@stthom.ernet.in

Hardware limitations for HPC



nsp@stthom.ernet.in

How Clusters Work?

Conventional supercomputers achieve their speed using extremely optimized hardware that operates at very high speed. Then, how do the clusters out-perform them?

Simple, they cheat.

While the supercomputer is optimized in hardware, the cluster is so in software. The cluster breaks down a problem in a special way so that it can distribute all the little pieces to its constituents. That way the overall problem gets solved very efficiently.

- A Brief Introduction To Commodity Clustering

Ryan Kaulakis

IMSc -Jan 2005

nsp@stthom.ernet.in

What is MOSIX?

- ***MOSIX is a software solution to minimise OS level bottlenecks - originally designed to improve performance of MPI and PVM on cluster systems***

http://www.mosix.org

Not open source

Free for personal and academic use

nsp@stthom.ernet.in

MOSIX

More Technically speaking:

- *MOSIX is a Single System Image (SSI) cluster that allows Automated Load Balancing across nodes through preemptive process migrations.*

Why Mosix?

- *Because you are not sure about the load on each node in the cluster - you are not the single user of the facility.*

- *You know how to stop unnecessary services and reduce overheads on the cluster, but you have no control over OS limitations.*

Why Mosix?

- *You want a fully optimized Cluster OS with no unwanted features - want to write your own "init" and service routines.*
- *You want to use all machines, not just the latest ones - do not want to contribute to the "electronic waste accumulation business"*

A Personal Note

I decided to build the first cluster when I noticed that I had accumulated six outdated PCs on the shelf in my study that were replaced by newer machines that attracted me with their newer features….

nsp@stthom.ernet.in

A Personal Note

- *It was a wastage of money*
- *A wastage of space*
- *Wastage of resources*

I thus built the first openMosix Cluster with those outdated PCs plus a PIV in June 2002.

MOSIX public version

Subject: Announcing MO6 for BSD/OS 3.0
Oren Laadan (orenl@cs.huji.ac.il)
Tue, 9 Sep 1997 19:50:12 +0300 (IDT)

Hi:
We are pleased to announce the availability of MO6 Version 3.0 Release 1.04 (beta-4) – compatible with BSD/OS 3.0, patch level K300-001 through M300-029. .....

MOSIX

In early 1999, Mosix M06 Beta was released for Linux kernel 2.2.1

- *MOSIX is the brainchild of Amnon Barak*

Prof. Amnon Barak
Department of Computer Science
The Hebrew University
Jerusalem 91904, Israel

E-mail: amnon@cs.huji.ac.il

MOSIX -Design Objectives

Turn a network of Linux computers into a High Performance Cluster computer.

Hide cluster complexity to users - Single System Image

Use resources as efficiently as possible -

dynamic load balancing

nsp@stthom.ernet.in

Like What?

Mosix makes a network of machines behave like a single machine with many processors and lots of memory (a Virtual SMP with linear scalability).

nsp@stthom.ernet.in

MOSIX -Design Concepts

Mosix cluster approach:

Cluster is NOT isolated machines + OS + middleware.

Cluster is a single machine RUNNING Cluster OS on nodes.

Allow automatic work distribution among cluster nodes.

Have Granularity at process level: "Fork and Forget"

nsp@stthom.ernet.in

openMosix- Architecture

- ***Farm of x86 processors***
- ***Supports both UP (uniprocessor) and SMP (symmetric multiprocessor) nodes.***
- ***Supports all LAN Interconnects from Ethernet cards to Myrinet.***

openMosix

- openMosix is the open source implementation of MOSIX.
- Dr. Moshe Bar is the project leader.

LJ: What was your goal in starting the openMosix project?

MB: I started openMosix because, as a prior project manager of Mosix, I could not agree to going non-GPL. I also didn't agree with how the project was managed by the co-manager at the time.
LJ on Tue, 2003-07-15 23:00

openMosix Availability

http://openmosix.sourceforge.net/

Precompiled kernel

or

Kernel source code

or

OS patch

nsp@stthom.ernet.in

openMosix Availability

CD based openMosix distributions are available for free download and they allow diskless nodes to form a cluster.

E.g.: clusterknoppix, CHAOS, plumpOS ….

Two tier technology

Information gathering and dissemination

using probabilistic dissemination algorithms.

This helps each node to have sufficient knowledge about available resources in other nodes, without polling.

Preemptive process migration ***that can migrate any process, anywhere, anytime - transparently***

- ***Supervised by* adaptive algorithms *that respond to global resource availability***
- ***Transparent to applications no change to user interface***

Information gathering and dissemination

Means:* oM supports scalable configurations *Same overhead for 16 nodes or 2056 nodes

Decentralised control and autonomy

Each node makes its own control decisions independently.

- *No master-slave relationships*
- *Each node is capable of operating as an independent system*
- *Nodes may join or leave the farm with minimal disruption*

nsp@stthom.ernet.in

Information gathering and dissemination

How?

In every unit of time (1 second) each node gathers and disseminates information about:

- ***CPU(s) speed, load and utilization***
- ***Free memory***
- ***Free proc-table/file-table slots***

Info sent to a randomly selected node

- ***Scalable - more nodes better scattering***

Process migration by adaptive resource management algorithms

Load balancing: ***reduce variance between pairs of nodes to improve the overall performance***

Memory ushering: ***migrate processes from a node that nearly exhausted its free memory and prevent paging***

Parallel File I/O: ***migrate the process to the file-server and allow direct I/O rather than choking the network.***

nsp@stthom.ernet.in

How Migration Works?

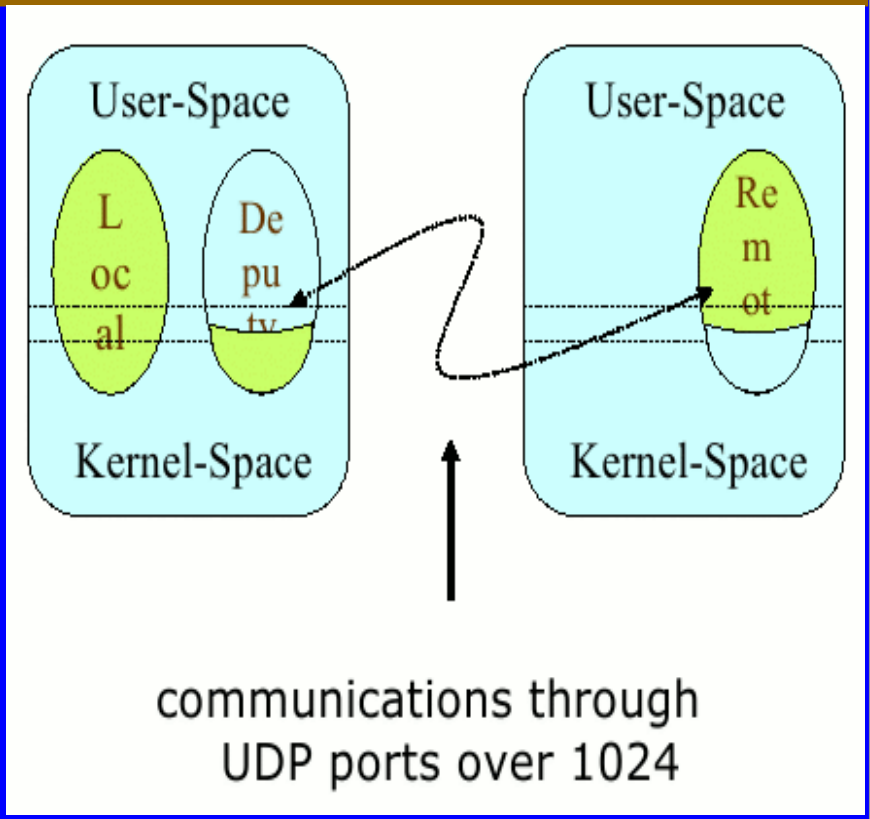
Process are divided into two pieces:

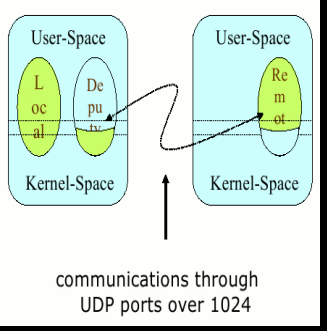
A deputy component:

that is kept in the UHN (Unique Home Node) & contains the kernel context (description of used resources + kernel stack + task structure).

A remote component:

that contains the user context (code+stack+data+memory map + registers) that gets migrated.

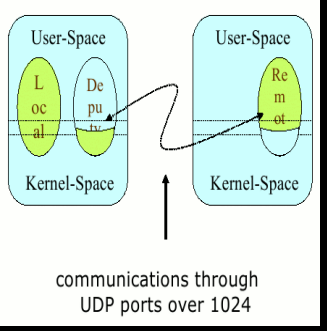




How Migration Works?

Migration time has a fixed component + a linear component that is proportional to the number of pages -> only necessary memory pages are transferred (mm+dirty pages+page tables)

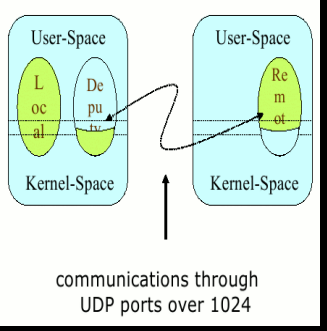
Site-dependent system calls are executed on the home node (synchronous communication). Other system calls are executed directly on the remote node.



How Migration Works?

For optimal resource usage, processes are migrated on the basis of a Big Mac Index Economic Algorithm that minimizes a GCF (global cost function).

- *Every resource usage on a remote node encumbers a "cost" (CPU, Memory, I/O). These costs are unified in the GCF.*
- *Processes are migrated to the node where they have the lowest cost: the most "economically" convenient distribution is selected.*
- *Each process is allowed to adaptively migrate as many times as required to realise the most"economic" configuration.*
- *To smooth fluctuations during load balancing and achieve scalability every node broadcasts its status to a random subset of nodes.*



How Migration Works?

The UHN asynchronously informs the remote about signals and process wake-up events (network data arrival).

Network links like sockets stay at the UHN while migratable sockets are on the way out.

If home node dies, the process dies.
If remote node dies, the process.

DFSA

Direct File System Access means:

- *To have a distributed FS*
- *It allows the migrated process to do IO "locally" in the migrated (remote) node.*
- *The resource sharing algorithm will move the process to the data!*

Distributed FS compatible with DFSA include:
oMFS, GFS, QFS (Qlusters), PVFS2

Cluster Configurations

Single-pool: all the servers and workstations are used as a single cluster: each machine is a part of the cluster and can migrate processes to each other.

Server-pool: servers form a cluster with isolated cluster nodes attached to it: servers will accept and migrate jobs to other servers or to the local nodes attached to them.

Adaptive-pool: servers are shared while workstations join or leave the cluster, depending on daytime.

How does it compare?

PVM and MPI software run more efficiently on the openMosix kernel than on Cluster unaware OS kernels.

OS Level Bottlenecks are minimized

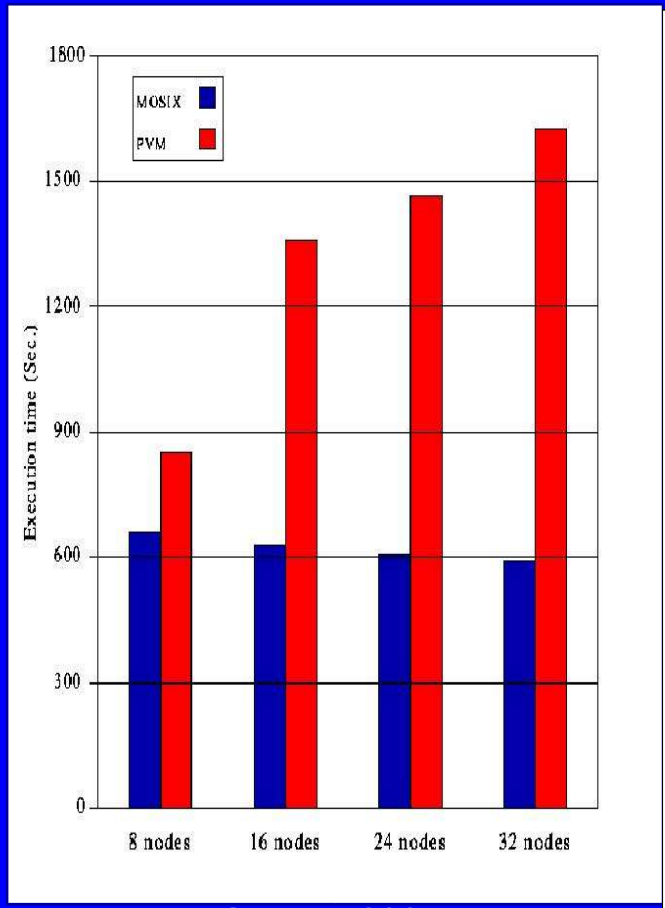
Process migration (MOSIX) **vs.** *static allocation (PVM/MPI)*

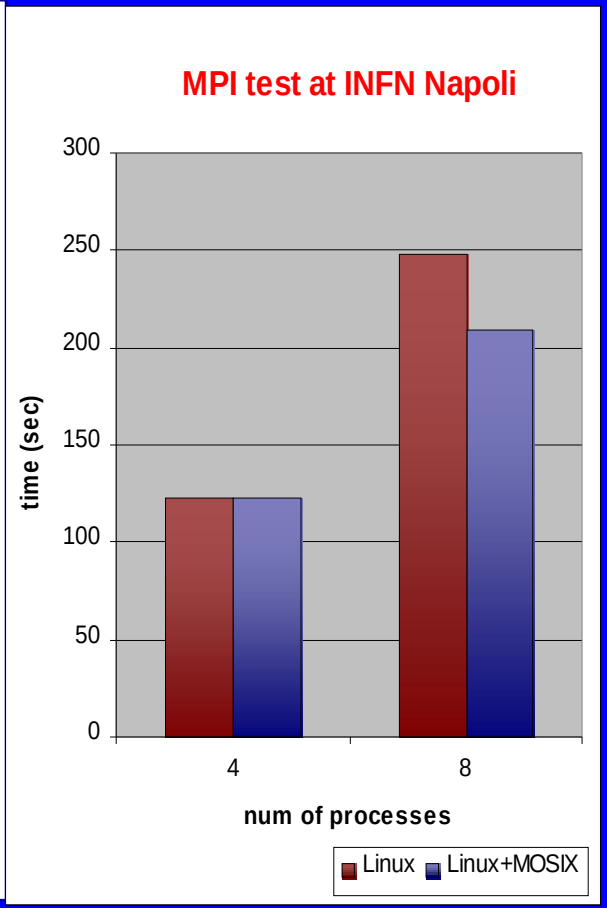
Fixed number of processes per node

Random process size with average 8MB

Note the performance **(un)**scalability !

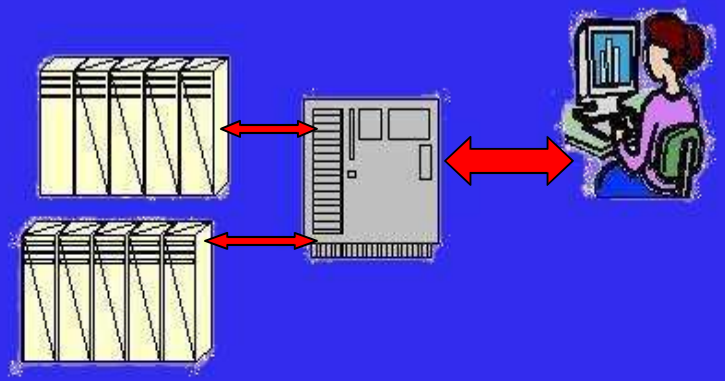
-Adopted from "Clustering with openMosix" by Maurizio Davini, Department of Physics and INFN Pisa



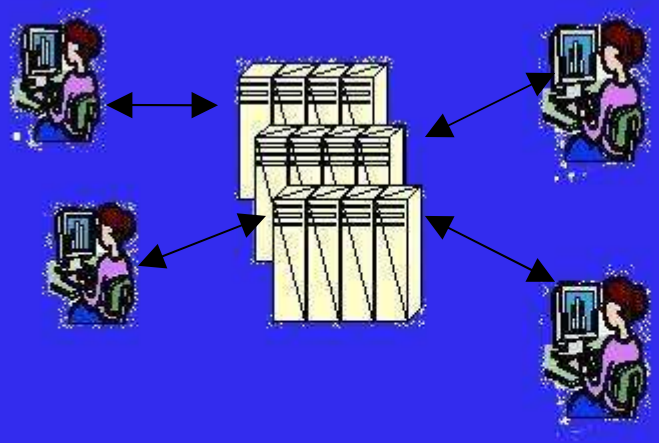


nsp@stthom.ernet.in

openMosix Features



Beowulf Cluster

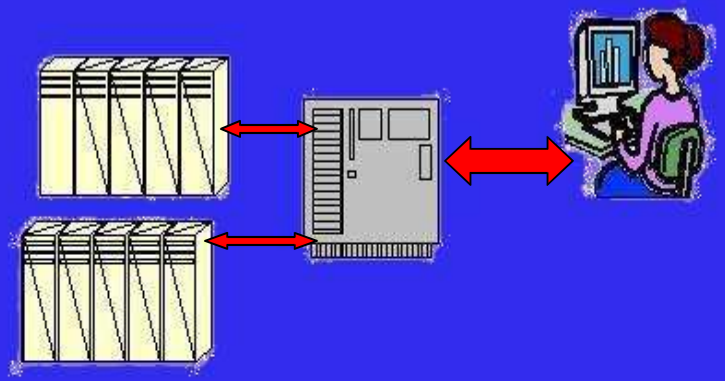


openMosix Cluster

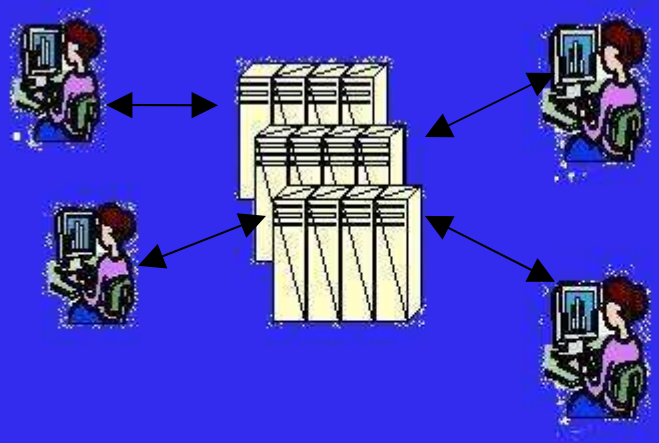
Allows dynamic addition & removal of nodes

nsp@stthom.ernet.in

openMosix Features



Beowulf Cluster

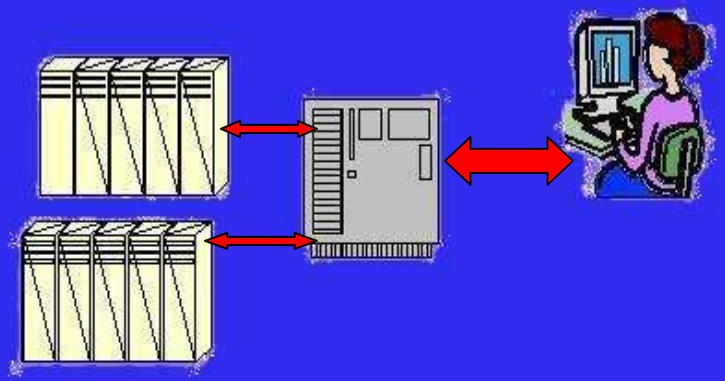


openMosix Cluster

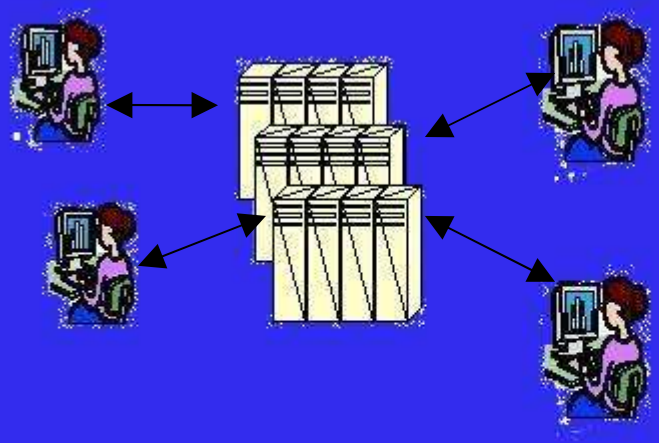
Preemptive process migration offer optimal load balancing across the nodes.

nsp@stthom.ernet.in

openMosix Features



Beowulf Cluster



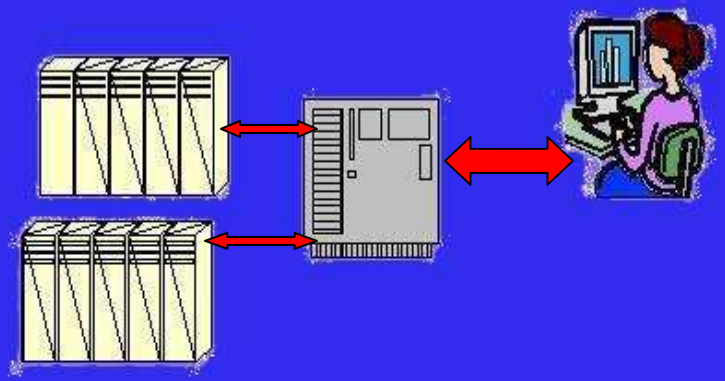
openMosix Cluster

B: ***MPI Code performance is the performance of the slowest node***

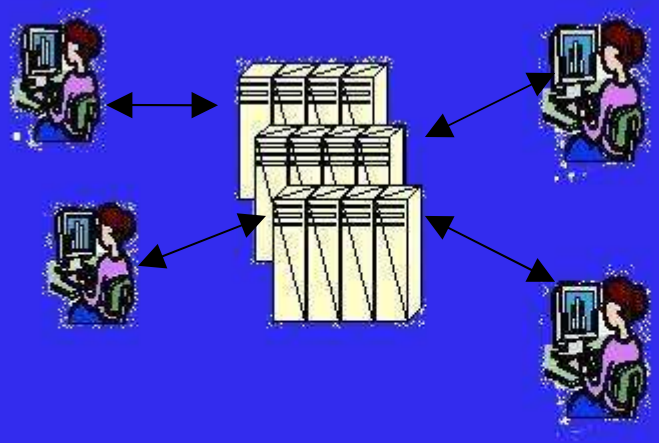
oM: ***MPI code performance is the performance of the fastest node***

nsp@stthom.ernet.in

openMosix Features



Beowulf Cluster

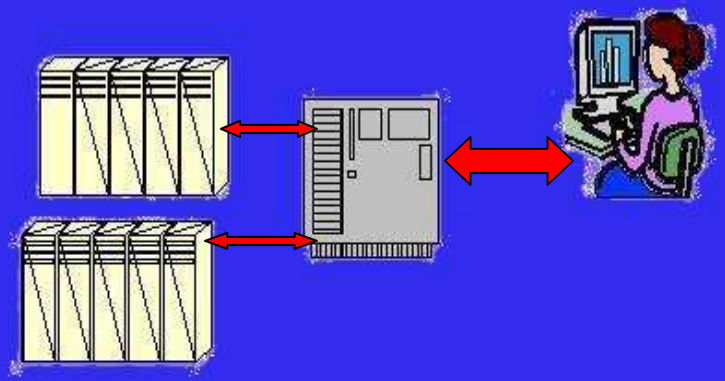


openMosix Cluster

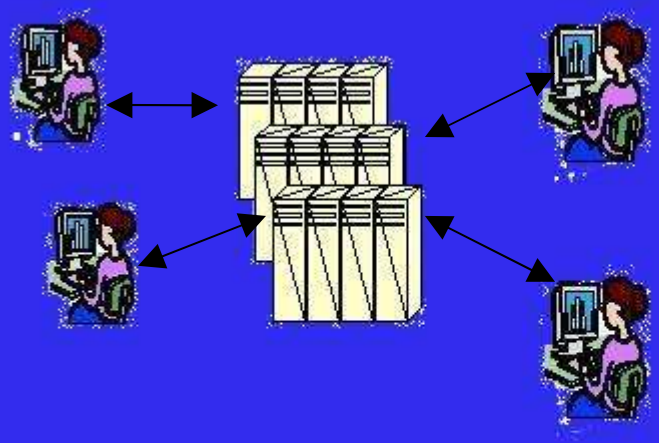
Adaptive resource allocation scheme allows use of heterogeneous nodes in the cluster.

nsp@stthom.ernet.in

openMosix Features



Beowulf Cluster

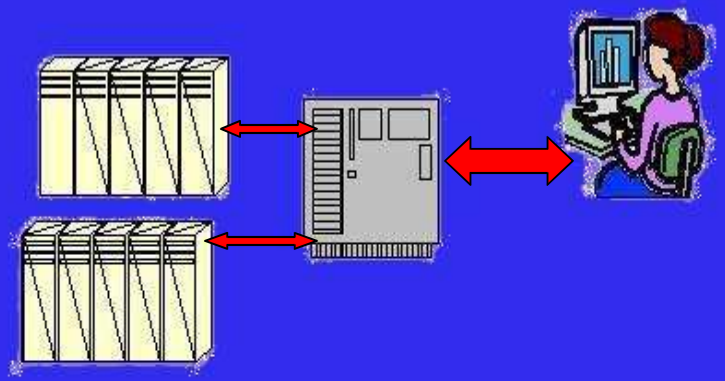


openMosix Cluster

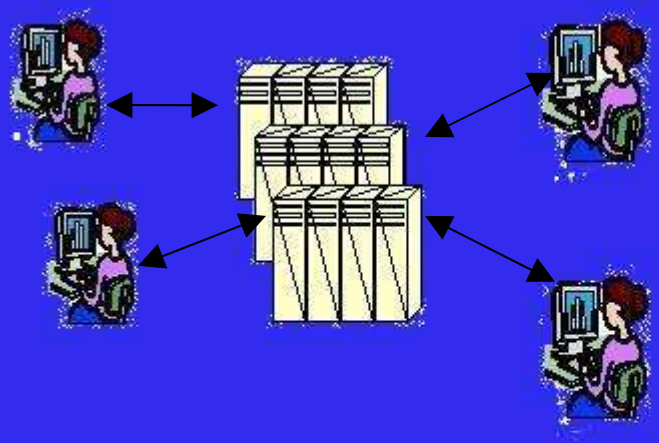
No need to modify, compile or link user code with any special software library

nsp@stthom.ernet.in

openMosix Features



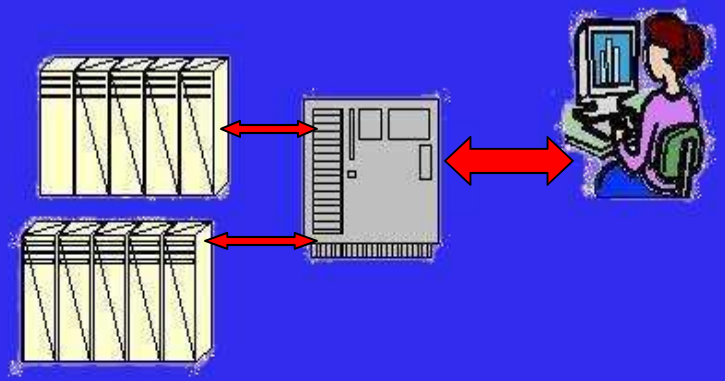
Beowulf Cluster



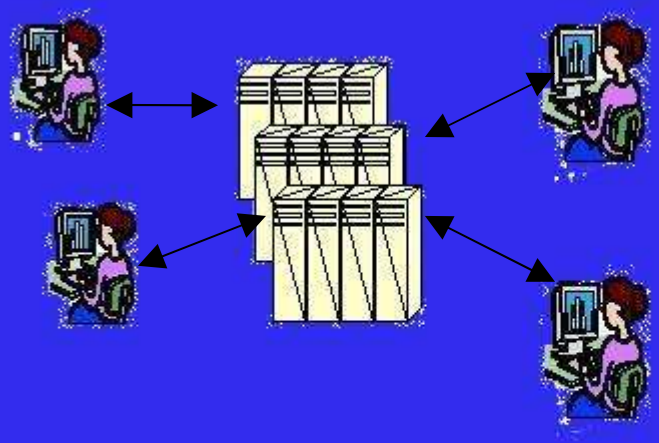
openMosix Cluster

Offers optimal performance for CPU bound code

openMosix Features



Beowulf Cluster

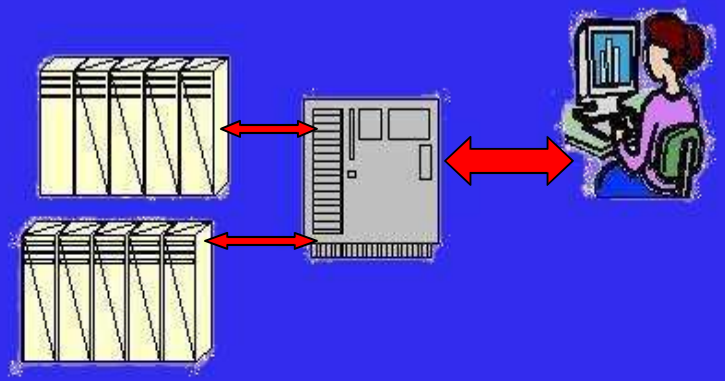


openMosix Cluster

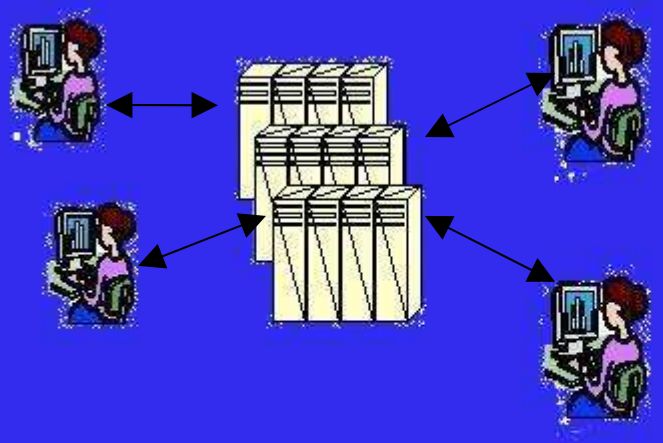
Uses DFSA for optimal buffered file resource sharing

nsp@stthom.ernet.in

openMosix Features



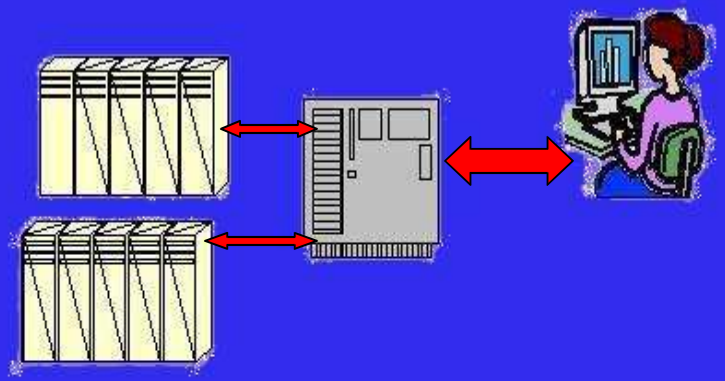
Beowulf Cluster



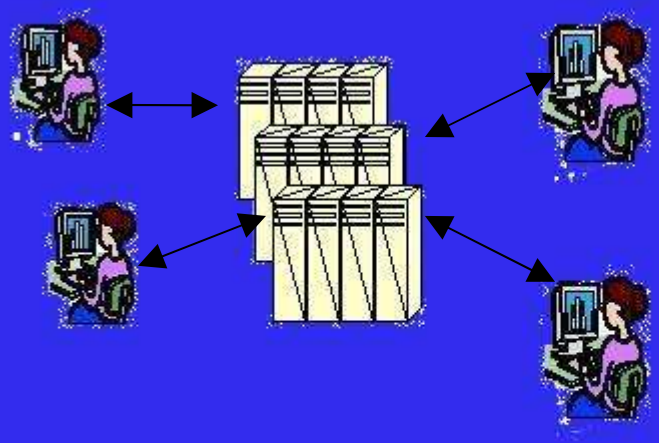
openMosix Cluster

Ideal for multi user, time shared systems

openMosix Features



Beowulf Cluster

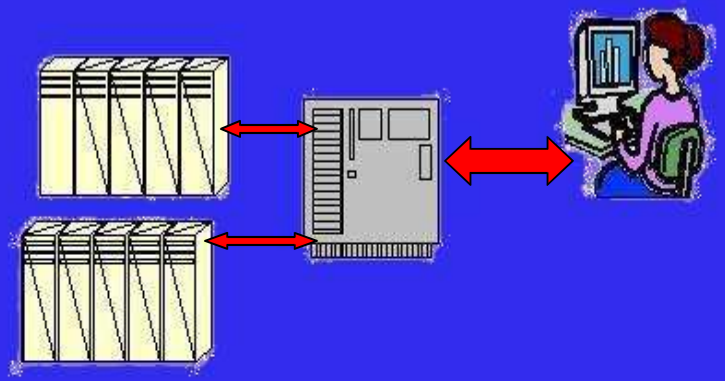


openMosix Cluster

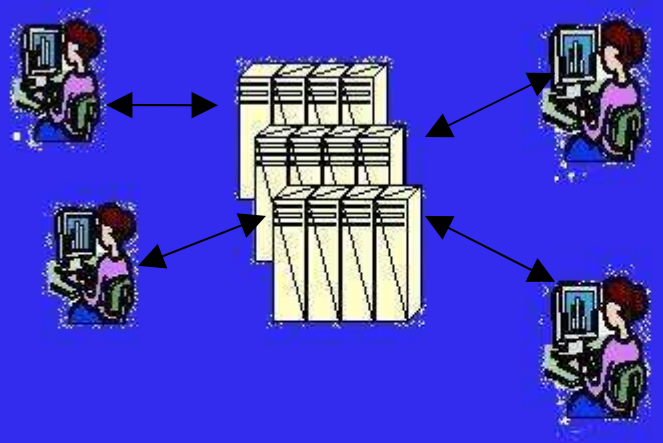
Large compilations - make -j 10 modules

nsp@stthom.ernet.in

openMosix Features



Beowulf Cluster

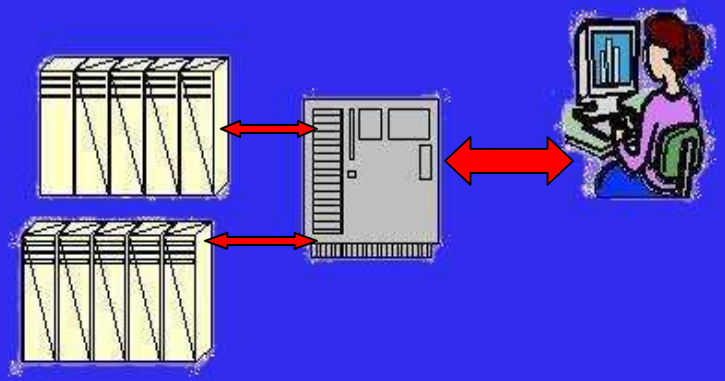


openMosix Cluster

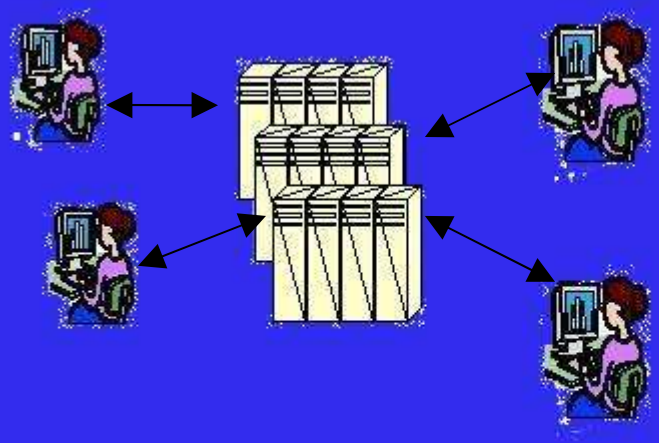
Optimal use of computing facility

nsp@stthom.ernet.in

openMosix Features



Beowulf Cluster



openMosix Cluster

For building farms with different speed nodes and/or memory sizes

When NOT openMosix?

- *I/O* ***bound applications -migration becomes a disadvantage***
- ***Shared Memory applications -work in progress***
- ***Explicit dependency to a particular hardware in a node***

Software Side

- MOSIX Linux kernel 2.2.19
  - 80 new files (40,000 lines)
  - 109 modified files (7,000 lines changed/added)
  - About 3,000 lines are load-balancing algorithms

Software Side

- openMosix Linux kernel 2.4.17
  - 47 new files (38,500 lines)
  - 126 kernel files modified (5,200 lines changed/added)
  - 48 user-level files (12,000 lines)

Software Side

In openMosix Linux kernel 2.4.x

- *No new system-calls are used.*
- *Administration and info through /proc/hpc*
  - /proc/hpc

    /proc/hpc/admin

    /proc/hpc/info

    /proc/hpc/nodes/nnnn (per node information)

    /proc/hpc/remote/pppp (remote procedure information)

Node Configuration

A cluster of 3 nodes with IP 192.168.1.50 -52

- define /etc/openmosix.map

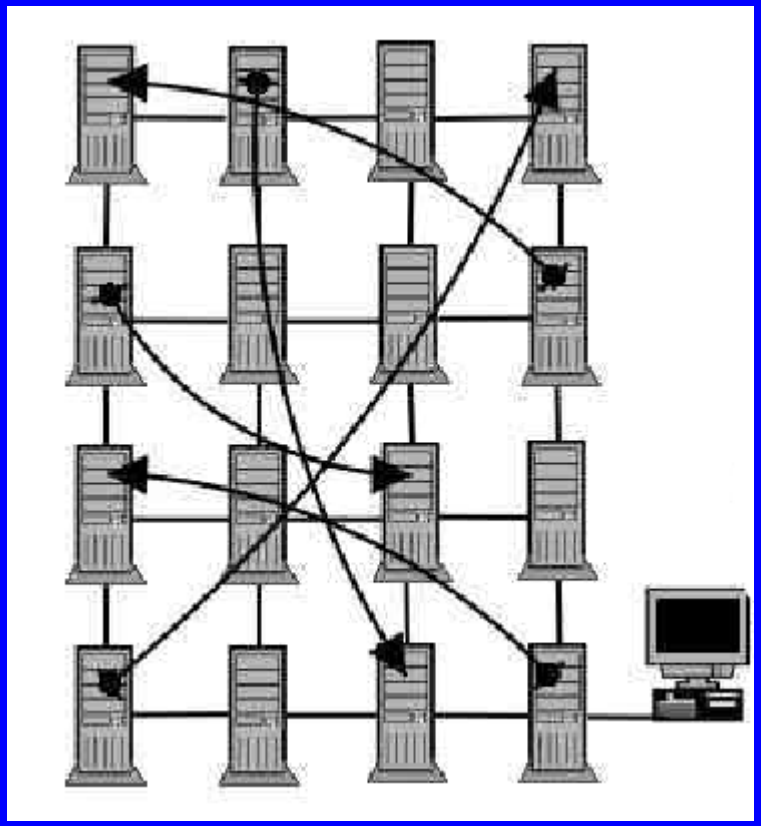
```
# MOSIX-# IP number-of-nodes
  1 192.168.1.50 3
```

OR

- define /etc/openmosix.map

```
# MOSIX-# IP number-of-nodes
  1 192.168.1.22 1
  2 192.168.1.60 1
  3 192.168.1.14 1
```

Performance monitoring



Available tools:

- *mosmon*
- *mps*
- *mtop*
- *openMosix view*
- *openMosix analyser*
- *openMosix migmon*
- *3dmosmon*

System Administration

setpe - openMosix node configuration
mosctl - to control process migration
mtop,mps - unix-like openMosix-ized commands
mosmon - text-based load monitoring

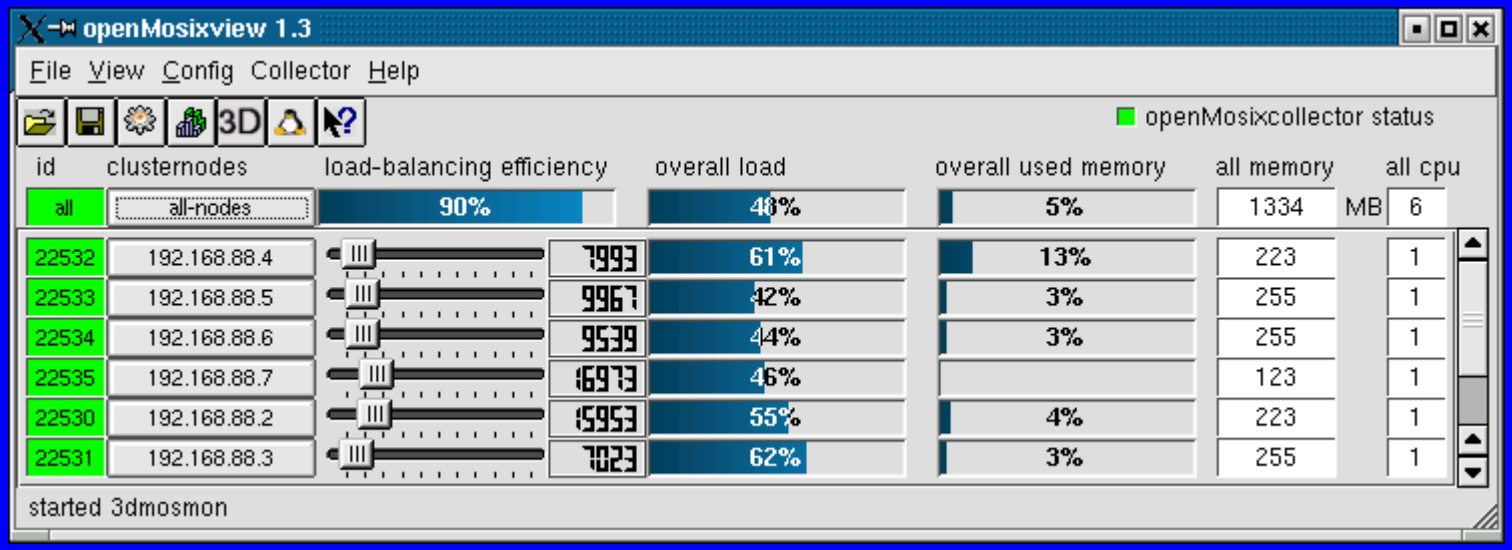
/proc/[PID]/cantmove: reason why a process can't migrate.
/proc/[PID]/goto: to which node the process should migrate
/proc/[PID]/lock: if a process is locked to its home node
/proc/[PID]/nmigs: how many times the process migrated
/proc/[PID]/where: where the process is currently being computed

System Administration

/proc/[PID]/migrate: same as goto remote processes
/proc/hpc/remote/from: the home node of the process
/proc/hpc/remote/identity: additional informations about the process
/proc/hpc/remote/statm: memory statistic of the process
/proc/hpc/remote/stats: cpu statistics of the process

nsp@stthom.ernet.in

openMosixview

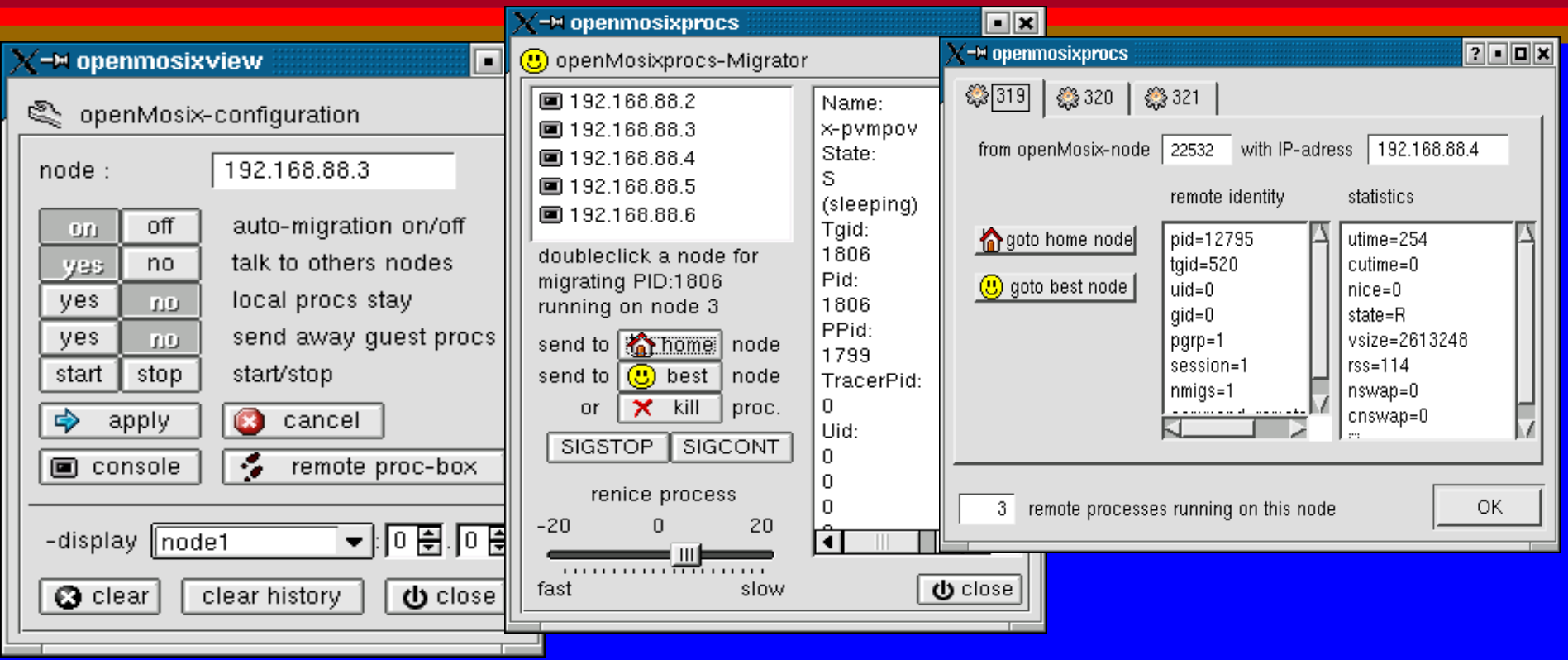


All parts are accessable from the main application window.

"Priority-sliders" for each node simplifying the manual and automatic load-balancing.

nsp@stthom.ernet.in

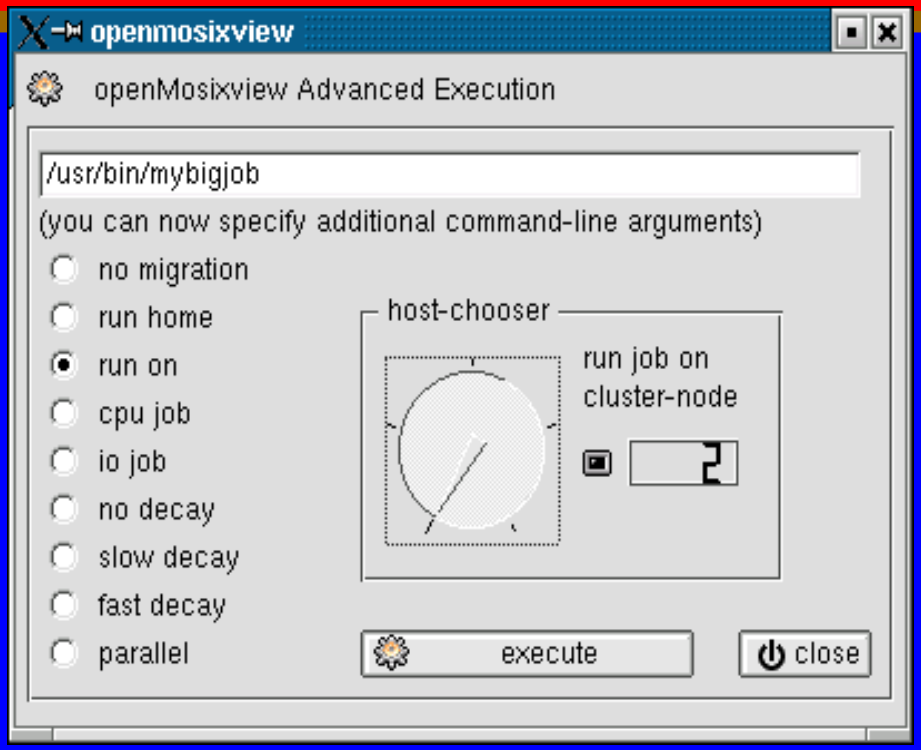
OpenMosixview



The most common openMosix-commands are executable by a few mouse-clicks.

nsp@stthom.ernet.in

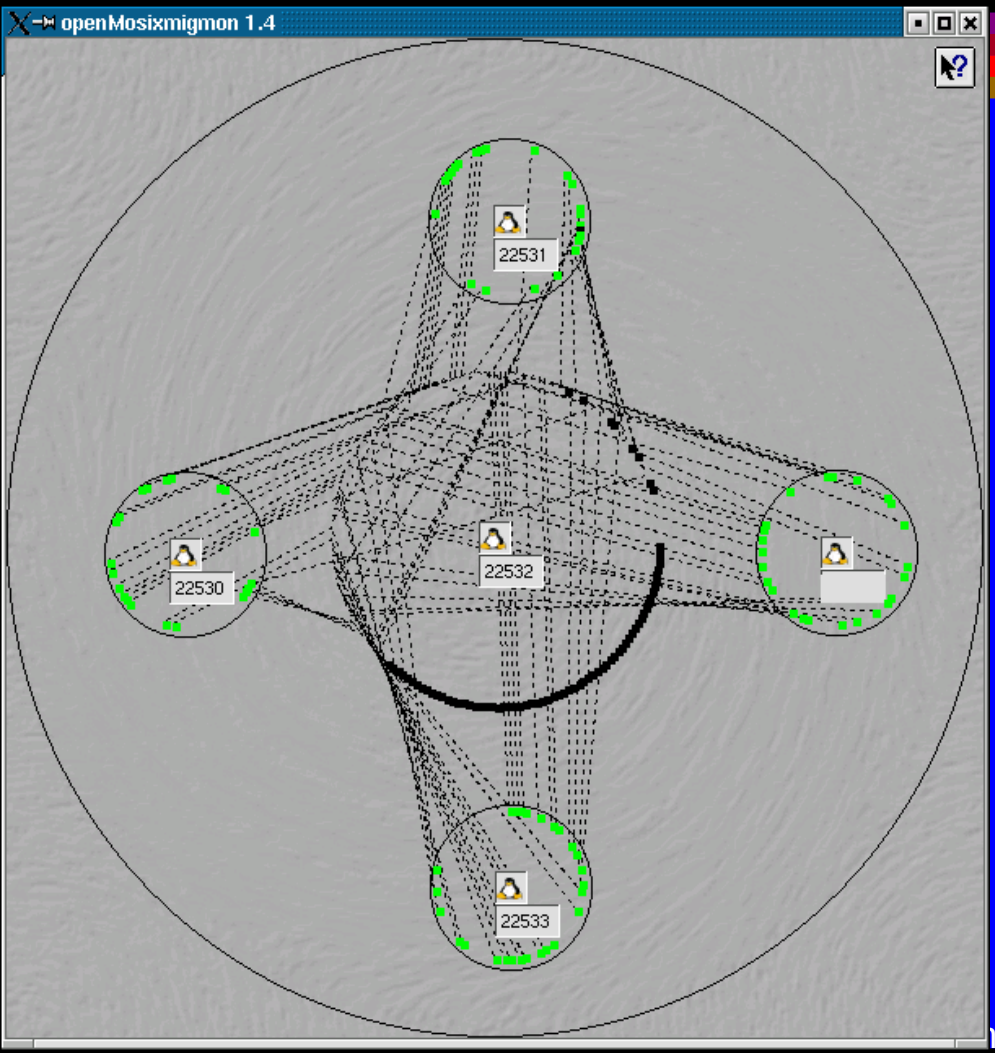
openMosixview



An advanced execution dialog helps to start applications on the cluster.

nsp@stthom.ernet.in

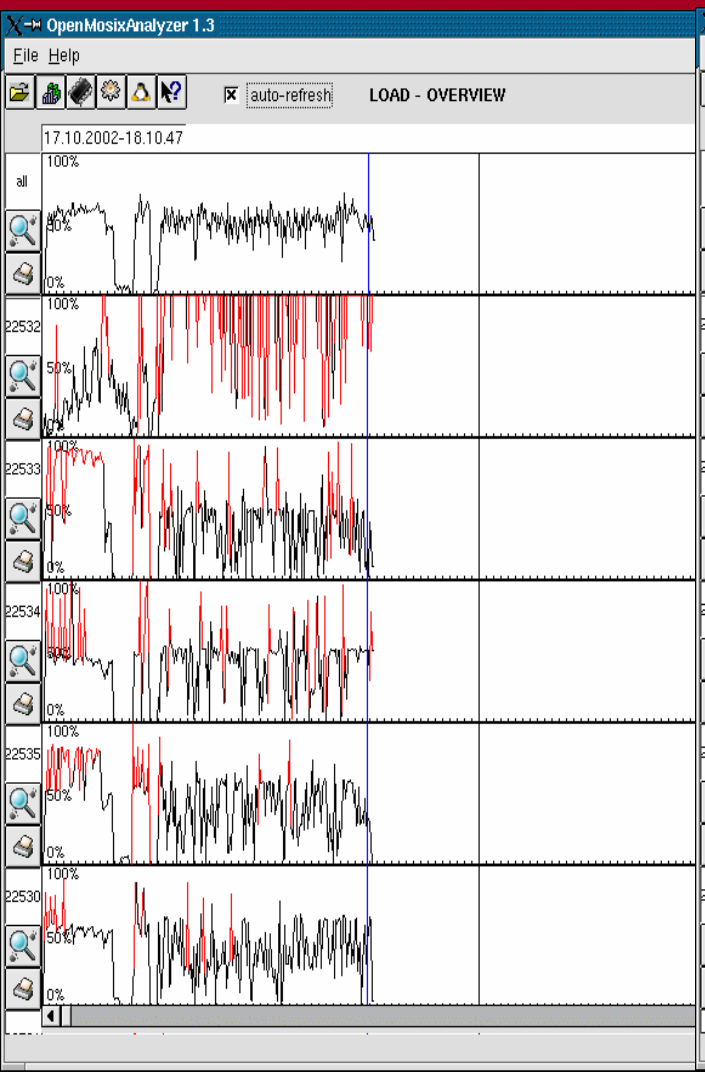
OpenMosixview

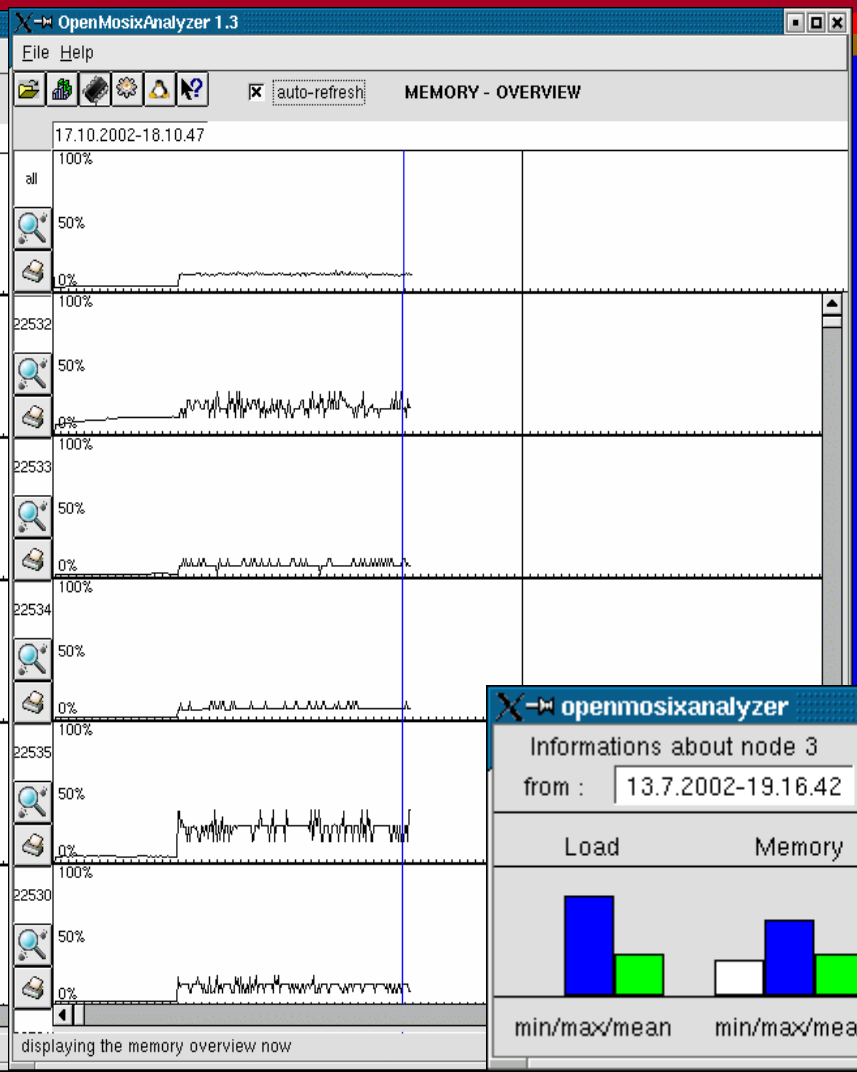


The migation monitor show how well the processes migrate across the nodes.

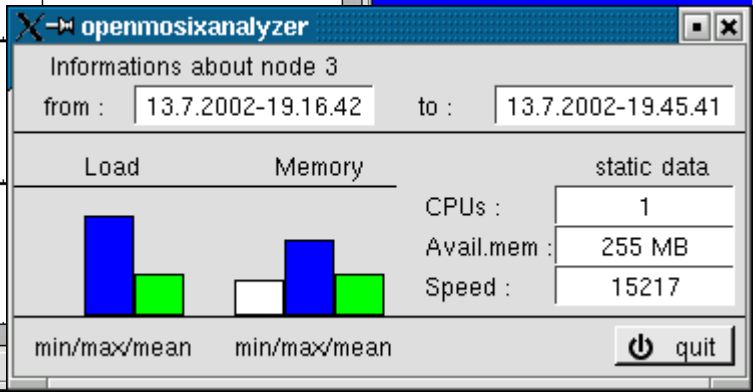
Manual control possible – just drag and drop

openMosixview

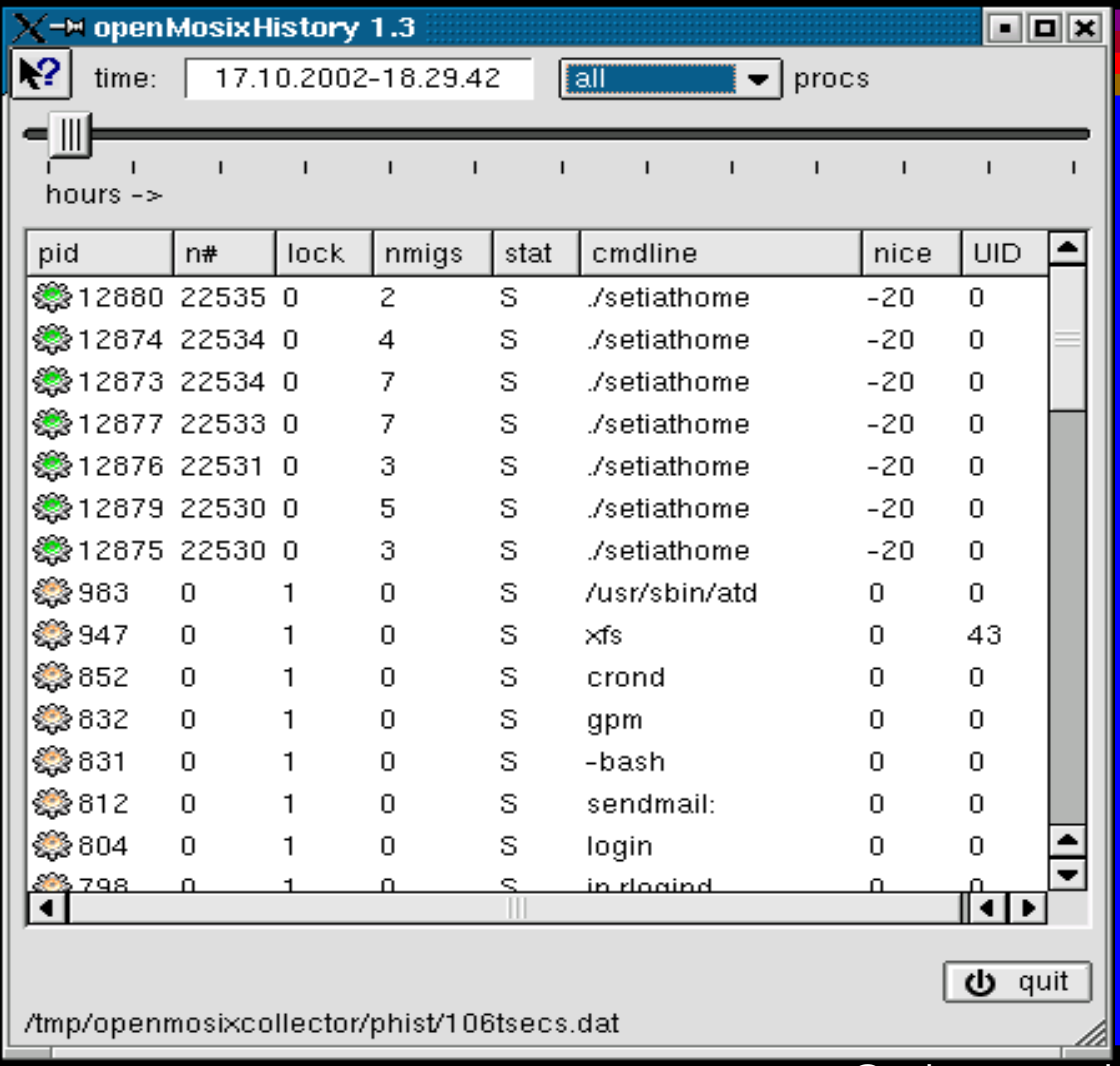




Load in each node can be monitored



openMosixview



Detailed History of processes ran and migrated is available for easy debugging.

openMosix attractions

Adding new nodes to a running openMosix cluster can be made dynamic with the auto discovery daemon.

Nodes can join in and withdraw gracefully without disturbing the processes running on the Cluster.

openMosix attractions

Add power on-demand:

The dynamic addition of nodes allow one to add or remove nodes from the cluster with ease.

It is even possible to avoid any kind of software installation on the system (if that scares you) with the use of live CD distributions of openMosix such as CHAOS, ClusterKnoppix, Quantian. etc.

openMosix attractions

Recycle all your old hardware: it will be used as best as possible

How?

Form cluster groups of machines of similar architecture. Keep a well done Linux box as front-end of each such group and other "abruptly" installed machines as back-end.

nsp@stthom.ernet.in

openMosix attractions

Applications that improve performance include:

- Matlab 5 migrates just fine. (Matlab 6 uses java threads and thus won't migrate. It is possible to enable migration with the –nojvm option)

- Octave is an opensource work-a-like to Matlab and is migratable

- MJPEG tools: Because it uses both i/o intense and cpu intense pipes, it works very well on small clusters.

openMosix attractions

Applications that improve performance include:

- bladeenc (easy rip your mp3 collection) works fine.
- Flac (lossless mp3 encoder) works fine.
- POVRAY Spread your pic-frames to multiple proccesses by a shell script or use the parallel (PVM) version.
- MPI Replacing rsh/ssh calls to nodes with mosixrun utility automatically migrates jobs to the best nodes available in the cluster. A patch for mpich version is available.

openMosix attractions

- Postfix the high performance mail transport agent migrates fine.
- SETI@home migrates fine.
- CISILIA the cluster aware passwd cracker engine performs better on openMosix.

openMosix attractions

- Grid Computing and Distributed Computing

- Sun's GridEngine for distributed and Grid Computing performs perfectly well on top of openMosix.

openMosix limitations

At present, processes using shared memory cannot migrate:

- Applications using pthreads (Java..)
- mySQL , uses pthreads.
- Apache, unless the MigShm patch is applied
- Mathematica, uses shared memory.
- SAP, uses shared memory
- Oracle, uses shared memory
- Baan, uses shared memory
- Postgres, uses shared memory,no semaphores
- Python with threading

The MAASK Team (Maya, Anu, Asmita, Snehal, Krushna) have developed a MigShm patch that is not yet added to official oM.

openMosix limitations

- Instabilty issues: sudden hard locks
- Many limitations are present (same kernel on nodes, no pthreads, X and kde migration instability..)
- No performance gain for a single process (apart from the advantage given by the migration of the other processes).

Research in HPC?

Notebook Computers, PDAs (personal digital assistants) and Palmtop Computers with stylus touch screens go wireless. Do we have the technology to have wireless Clusters?

Traffic control,

Disaster management..

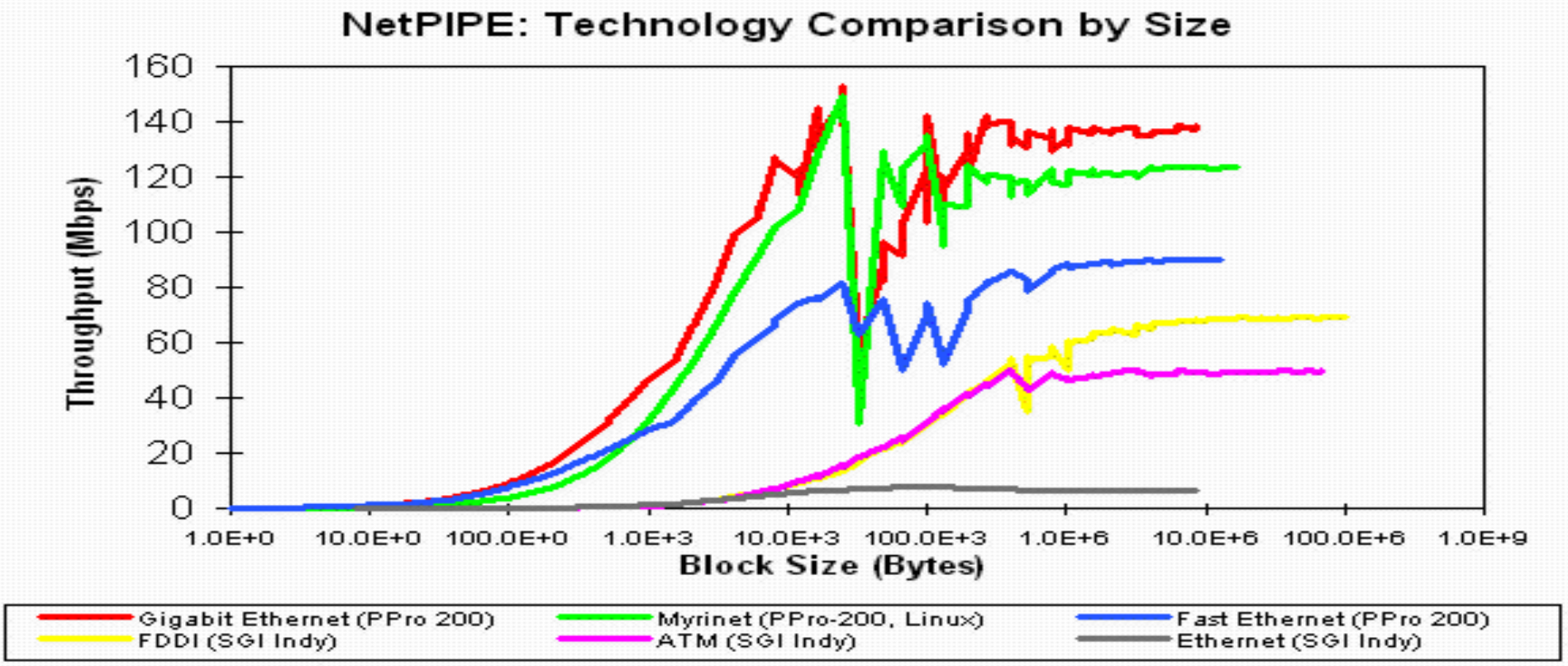
nsp@stthom.ernet.in

Research in Wireless HPC?

- Can be used to set up a high-speed Wireless-G (802.11g) network.
- Data rates up to 54Mbps - 5 times faster than Wireless-B (802.11b).
- Also interoperates with Wireless-B networks (at 11Mbps).
- Advanced wireless security with WPA, 128-bit WEP encryption, and MAC filtering.

nsp@stthom.ernet.in

Network Bandwidth



Curtsey: SCL Lab, Iowa State University

Research in Wireless HPC?

nsp@stthom.ernet.in

Santhom Cluster Project

nsp@stthom.ernet.in

Thank You

IMSc -Jan 2005
nsp@stthom.ernet.in