openMosix migShm

migShm experiences
shared memory migration on openMosix
Live free() or die()

Kris Buytaert
kb@ipng.be

LinuxKongress, Saarbrucken , 14-16 October 2003

Intro

migShm is a new development in openMosix developed by Maask group

This talk is about my experiences with migShm

Welcome

agenda

1. openMosix
2. Issues
3. Shared Memory
4. Maask migShm
5. Environment
6. Building
7. Testing
8. Apache
9. Blast
10. mySQL and other
11. Conclusions
12. Thanks
13. Questions ?

1. openMosix

1.1. openMosix

openMosix

- Kernel Patch + userland tools
- Automatic Migration of processes to other nodes
- Statistics define to which node a process can migrate
- not everything migrates, there are limitations

1.2. New Features in openMosix

- Shared Memory Migration
- CheckPointing
- ClusterMask
- Autodiscovery
- General openMosix Deamon
- Load-Limit

2. Issues

Not all applications migrate to remote nodes due to different reasons

- Shared Memory
- Hardware restrictions
- Threading issues
- ...

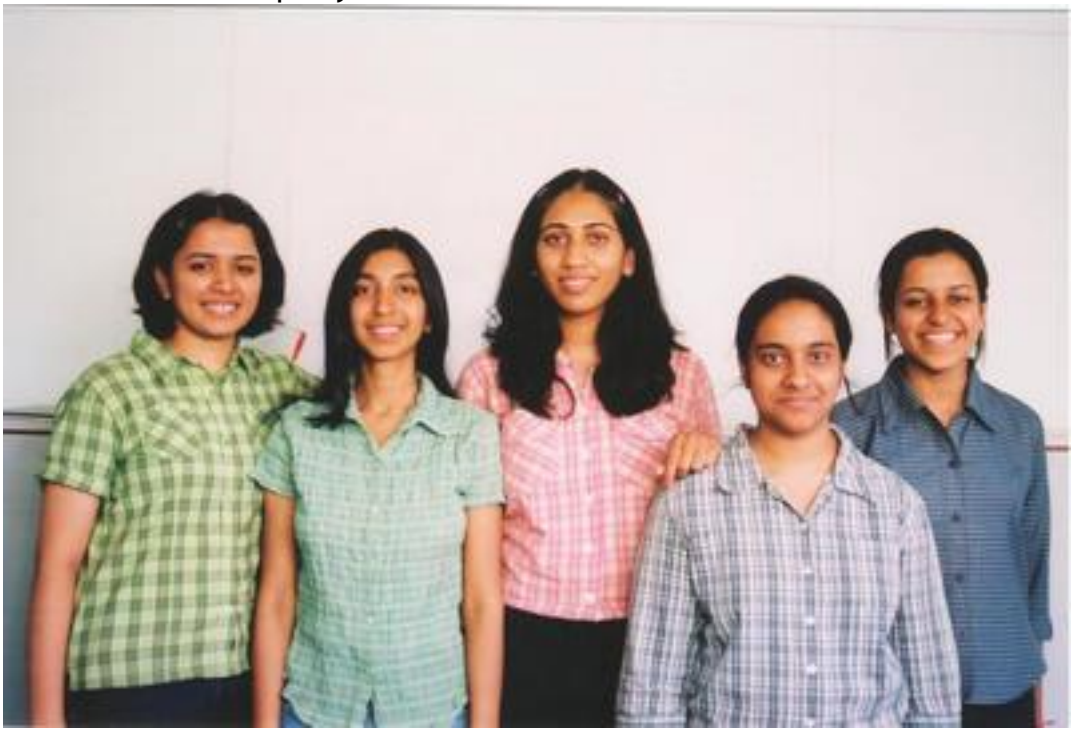
3. Shared Memory

3.1. The Problem

- A process uses shared memory and has to be migrated.
- Normally openMosix destroys the memory map and recreates them on the remote node
- Other apps are using the same region
- 2 shared memory processes on the same node,
- migrate both processes with their memory to the same remote node
- exponential memory use growth

3.2. Maask

Maya, Anu, Asmita, Snehal, Krushna, the developers of Migshm, are students of Cummins College of Engineering, University of Pune, India. This is their first project



3.3. Modular approach

- Migration of shared memory processes
- Communication module
- Consistency module
- Access Logs and migration decisions
- Migration of shared memory
- Thread Migration

3.4. System Calls

In order to manage shared memory one needs a number of system calls such as shmget(), shmat(), shmdt() and shmctl(). Each of these function calls has to be made transparently available at the remote when such a shared memory process tries to migrate, these calls then have to access the local memory map.

3.5. Communication Module

A Communication Module was needed in order to have communication between the deputy and the remote stub. MAASK group used the commlink for this purpose, however in order to make sure that they could guarantee communications between two processes on different nodes they also had to implement a daemon active on every node of the openMosix cluster called MigSharedMemD, which is listening on port 0x3418. A header type is defined to facilitate the communication.

3.6. Consistency

Different processes using the same shared memory regions can migrate to different nodes, on those different nodes they work on local copies of that share memory. Consistency has to be maintained between the different instances of these local copies. MAASK group opted for the Eager Release consistency model, this implements that a local copies of modified pages will only be written to the original owner when the lock on that memory segment is released. , thus not for every write. This ensures that the owner node of the shared memory always has the latest copy. Upon being read by another remote process, it page faults from the owner node to fetch the latest up to date pages.

3.7. migshmdaemon

A `mig_shm_daemon()` is running on every node , once an invalidate message is sent by the owner to the remotes, the page table entries of all remote processes for the pages that were modified are being invalidated. The next time a process tries to access a recently invalidated page , it will page fault to the owner node and get the latest copy from the owner node

The updating of the pages from the remote node to the owner node is performed by a write-back operation, hence the dirty pages are being sent back to the owner node in order to be restored to their correct position, just before release of the lock on the remote node.

3.8. Deciding on Migration

Just as in the original openMosix algorithms one has to know which processes that use shared memory are actually suitable for migration, To facilitate this a module has been written that monitors and logs the access to shared memory. The functionality of openMosix's memsorter daemon has been extended to monitor all accesses of different processes to a shared memory region. MAASK assumed that a process attaches to only one shared memory region .

3.9. Migrate memory along ?

Based on these statistics and a threshold migShm determines how strong a process is linked to a certain shared memory region. A weakly linked process is migrated without migrating the memory. For a strongly linked process whether a process with or without the memory is being migrated depends on how strong other processes are linked to it. If the selected process is the most active then memory is migrated with the process.

3.10. Migrating

When a process is found to be suitable for migration and the shared memory is migrated along with the process to a remote node, this remote node becomes the owner of the shared memory node.

4. Environment

A PIII 666 with 254Mb ram
A PII 266 with 64Mb ram
A PIII 500 with 128Mb ram

These machines were connected using a 10 Mb HUB. openmosixmigshm 2.4.21 kernels (based on the migShm-1.4 patch for openMosix 2.4.21-1

http://howto.ipng.be/openMosix-Migshm-rpm/

5. Building

As migShm is a patch to openMosix, which is a patch to the vanilla Linux kernel, building the migShm kernel requires us first to patch the vanilla Linux kernelversion

```
patch -Np1 < openMosix-2.4.21-1
```

then the migShm patch.

```
patch -p1< patch-Migshm-2.4.21.diff
```

Enabling options :

```
Shared memory migration support (Experimental) (CONFIG_SHM) [N/y/?] (NEW
  flush() support for consistency (CONFIG_SHM_FLUSH) [N/y/?] (NEW) y
  Kernel Debug messages (CONFIG_SHM_DEBUG) [N/y/?] (NEW) y
```

6. Testing

6.1. The Tests

- Stability
- openMosix Functionality
- migShm Functionality

- openMosix Stress Test
- migShm Stress Test

6.2. Results

- Don't run test while running X
- Minor issues on openMosix stresstest
- Run migShm tests as root
- Conlusion: programs using shared memory migrate.

7. Apache

- Maask group published results on 1.3.20
- Reran tests on 10Mbit
- Processes migrate but no gain in performance
- Apache 2.0.40 does not migrate correctly

- According to available bandwith, migShm does or does not increase performance

8. Blast

Blast is one of the most popular applications in the Bio-informatics world, lots of people already tried to run Blast on openMosix with mixed success. A normal blast version uses shared memory, however a patched version of Blast exists that doest not use shared memory.

migShm enables the migration of Blast on an openMosix cluster.

9. Others ?

- migShm does not solve al the problems yet.
- Applications that use Threads, such as mySQL, Mathlab, still don't migrate.
- Postgress uses shared memory but not the system semaphores for locking it. Therefore it does not migrate.

10. Conclusions

- migShm enables the migration of certain shared memory applications
- migShm does not enable pthread migration
- Issues with Xfree, not stable yet
- We need more people testing

11. Thanks

- Moshe Bar
- Maask Group
- Mirko Caserta
- Matthias Rechenburg

12. Questions

Questions ?