

Project Progress Report

Distributive Memory benchmarking with openMosix Clustering

Prepared for: Professor Kaugars, CS 454, Spring 2006

Prepared by: Lucas Holt & Swaroop Atre

March 28, 2006

Distributive Memory benchmarking with openMosix Clustering



Overview

Our goal with the project is to prove the usefulness of openMosix with desktop environments including home use and possible other situations like computer labs. Many home users will find the install process painful. The difficulty to setup the environment must be weighed with the cost of acquiring newer hardware which supports SMP or SMP like activity and/or the cost of purchasing memory. The combination of shared memory and running processes on multiple machines would be extremely useful in research and other areas.

Installation hurdles

Using newer hardware is quite difficult do to the limitation of the 2.4.30 linux kernel. One of our planned test machines does not boot properly on a 2.4 kernel and thus can not be used. Gentoo distributions dating as far back as the 2004.3 release default to a 2.6 kernel. The 2.4 kernel is very picky about which version of gcc is used among other things. gcc 2.9x releases seem to be required for a reliable build.

The gentoo instructions for installing openMosix look quite simple. <http://www.gentoo.org/doc/en/openmosix-howto.xml>

The simplicity of the directions is quite misleading. Emerging the openMosix related files is not that simple. The kernel is not bootable using a 2005.1 or 2006.0 cd without several steps. udev is not supported. We do not yet have a cluster in operation. I will comment further below on this.

Tasks (from February report)

- Setup a host with Linux and install an openMosix kernel. Verify the host is stable and functional.
- Setup and configure additional linux hosts with a similar configuration and identical kernel version.
- Create a series of tests we can run to prove the functionality of the cluster.
- Run the tests on the cluster and compare them to the results running individually without openMosix distributing the load. Also compare to an SMP machine without openMosix active (but still in the kernel) as a baseline.
- Formalize a series of benchmarks and determine the environmental needs and attempt to create a consistent environment for the tests. Run the tests as many times as possible in the time available.
- Draw conclusions from the results and present them.

Progress (from February report)

1. A Linux machine was setup with the latest Gentoo 2005.1 release. Portage includes the openMosix kernel patches and source to the 2.4.30 kernel. The 2.6.x release is masked do to lack of user-land tools for it.

2. The openMosix kernel was installed and booted on the host.
3. Currently scrounging for more hardware to do a cluster. At least two more machines are available.
4. Currently researching possible tests we can run on the cluster. The lame encoder will be included in testing.
5. Expect to have at least 3 computers simultaneously setup for openMosix testing, benchmarks run individually and together on the hosts by the project deadline. Ideally would like to run at least 10 different tests to verify results focusing on memory and CPU intensive operations. The results should prove interesting both in terms of scheduling and SMP operation.

Current Progress

1. Two machines have Gentoo 2005.1 Release installed. The openMosix kernel is bootable on one machine and does not boot on the second machine do to hardware compatibility issues. We are in the process of installing openMosix on two more machines.
2. The userland tools are installed and functioning on host1.
3. Several benchmarks have been picked. We plan on using RAMspeed 3.3 and povrey 3.6 for benchmarking the memory performance on the hosts individually and while in cluster mode. In addition, lame and x11 environments will be benchmarked. (gnome and kde with apps running) I'm working on a number of c programs to do simple cpu bound and memory bound tests as well. Install apache 2 and determine if it will run properly under openMosix with some of the httpd processes spanned across physical machines. I also wish to test mpeg video encoding time if possible using mplayer or xine's mpeg output abilities.
4. ramsmg and povrey have been run on host1 to get a baseline without the cluster up under the kernel. I'll include some of the results.
5. Physical SMP test might not be possible as that is the machine that has some hardware issues with the 2.4.x kernel.

More on Hardware

host1: Intel Pentium 4 1.4 ghz w/ 512mb ram (old CAE lab machine)

This machine was setup in one of the CAE side labs by Swaroop. Both of us have run initial benchmarks and worked on the setup. (2005.1 gentoo, 2.4.30 kernel)

host2: Intel Xeon 2.0 ghz (2 cpus) w/ 1GB ram (my home desktop)

System is having trouble booting the 2.4 kernel. I'm working on the problem.

host3: AMD Sempron 2300+ w/ 512mb ram (second home machine)

I'm currently installing gentoo 2006.0 on it as I write this. The motherboard chipset and video card were out during the 2.4 kernel series (nVidia nForce2 and Geforce fx5200) Should work fine. I'll bring the machine into the lab for testing. if host2 works, I'll also run a separate cluster on my gigabit network here and provide the results.)

host4: CAE Laptop? TBD. Swaroop hopes to borrow a CAE laptop and setup the machine as a client using a wired network connection. He's currently working on this.

RAMspeed

RAMspeed (ramsmg) is a memory and cache benchmark tool. We will benchmark the hosts individually and then the cluster. The network will be 100 base T using host1 and if possible gigabit if I can get two systems up at home.

<http://www.alasir.com/ramspeed/index.html>

Swaroop has install the SMP version on our first test machine. Here are the initial benchmark results using floating point:

openMosixN1 ramsmp-3.3.0 # ./ramsm -b6 -l5 -p16

RAMspeed/SMP (GENERIC) v3.3.0 by Rhett M. Hollander and others, 2002-O5

4Gb per pass mode, 16 processes

5-benchmark FLOATmem BatchRun mode

Benchmark #1:

FL-POINT Copy: 1085.63 Mb/s

FL-POINT Scale: 951.29 Mb/s

FL-POINT Add: 1827.13 Mb/s

FL-POINT Triad: 1561.60 Mb/s

FL-POINT AVERAGE: 1356.41 Mb/s

Benchmark #2:

FL-POINT Copy: 1130.59 Mb/s

FL-POINT Scale: 1072.77 Mb/s

FL-POINT Add: 1468.07 Mb/s

FL-POINT Triad: 1694.39 Mb/s

FL-POINT AVERAGE: 1341.46 Mb/s

Benchmark #3:

FL-POINT Copy: 1050.19 Mb/s

FL-POINT Scale: 996.10 Mb/s

CS454

FL-POINT Add: 1389.89 Mb/s

FL-POINT Triad: 1646.20 Mb/s

FL-POINT AVERAGE: 1270.60 Mb/s

POVray

Povray is often used to benchmark cpu performance on various machines. There are several sites cataloging performance such as <http://www.haveland.com/index.htm?povbench/index.php>. Certain tests can also spawn multiple processes and consume the system memory which will allow us to expand our test out into the shared memory region using openMosix's shared memory patch. This will be the first test we run to determine the capabilities of openMosix. Since it spawns multiple processes, we can also load balance the test across hosts.