# Quantian as an environment for distributed statistical computing

Dirk Eddelbuettel

Debian Project
edd@debian.org

*DSC 2005 – Directions in Statistical Computing 2005*
University of Washington, Seattle, August 13-14, 2005

# Outline

# What is Quantian?

## A live-dvd for numbers geeks

- Quantian is a directly bootable and self-configuring Linux sytem that runs from a compressed dvd image.
- Quantian offers zero-configuration cluster computing using openMosix, including 'openMosix Terminalserver' via PXE.
- Quantian contains over 6gb of software, including an additional 4gb of 'quantitative' software: scientific, numerical, statistical, engineering, ...
- Quantian also contains editors, programming languages, complete latex support, two 'office' suites, networking tools and more.

# Quantian lineage

## Knoppix, clusterKnoppix, Debian

- Quantian is based on clusterKnoppix, which extends Knoppix with an openMosix-enabled kernel and applications, kernel modules and security patches.
- ClusterKnoppix extends Knoppix, an impressive 'linux on a cdrom' system which puts 2.1gb of software onto a cdrom along with auto-detection and configuration (but Knopppix followed Quantian and switched to 4gb dvds).
- Knoppix is based on Debian, a Linux distribution containing over 8000 source packages available for 12 architectures (such as i386, alpha, ia64, amd64, sparc or s390) produced by hundreds of individuals.

# Timeline

## As provided by the releases

- 0.1 (March 2003): Initial version at DSC 2003.
- 0.2 (May 2003): Now based on Knoppix 3.2.
- 0.3 (June 2003): Switched to using clusterKnoppix which added openMosix clustering support.
- 0.3.9.* (Sep. 2003): Updated clusterKnoppix.
- 0.4.9.* (Oct. 2003 to Mar. 2004): Based on Knoppix 3.3.
- 0.5.9.* (June to Sep. 2004): Based on Knoppix 3.4, first ’kitchen sink’ versions > 1gb for bootable DVDs.
- 0.6.9.* (Oct. to Dec. 2004): Based on Knoppix 3.6, size increased to 2.0gb.

# Motivation

## Major modes of use

- **Computing clusters** to speed up embarrassingly parallel tasks.
- **Computer labs** by enabling temporary use of a computing environment booted off a dvd, and/or netbooting.
- **Students / co-workers** as distributing DVDs enables work in identical environments with minimal administration.
- **Convenience** of not having to chase down new software releases, and to configure and installing it.
- **Easier installation** of a 'normal' workstation by booting off Quantian, and / or installing to hard disk getting a head start with 6gb of configured software.

# What is included?

## Broken down by field

- **Statistics:** GNU R (plus essentially all of CRAN and BioConductor; Ggobi, ESS), Xlispstat, Gretl, PSPP.
- **Bioinformatics:** BioConductor, BioPython, BioPerl and tools like emboss and blast2.
- **Mathematics:** Six computer algebra systems, matrix languages Octave (with add-on packages), Yorick and Scilab, TeXmacs front-end.
- **Physics:** CERN tools (Cernlib, Geant, PAW/PAW++), Scientific / Numeric Python, GNU GSL libraries.
- **Visualization and graphics:** OpenDX, Mayavi, Ggobi, Gnuplot, Grace, Gri, plotutils, xfig.

# What is included?

## Broken down by application area

- **Programming languages:** C, C++, Fortran, Java, Perl, Python, PHP, Ruby, Lua, Tcl, Awk, A+.
- **Editors:** XEmacs, Vim, jed, joe, kate, nedit, zile.
- **Scientific Publishing:** Extended LaTeX support with several frontends (xemacs, kile, lyx) and extensions.
- **Office software:** OpenOffice.org, KOffice, Gnumeric, and tools like the Gimp.
- **Finance:** Software from the Rmetrics project and the QuantLib libraries.
- **Networking:** ethereal, portmap, netcat, ethercap, bittorent, nmap, squid plus wireless tools and drivers.
- **General tools:** Apache, MySQL, PHP, and more.

# How to use many computers

## Conceptual overview

- 'sneaker net': physically (or virtually via ssh) running from machine to machine, launching jobs and collecting results.
- 'Beowulf' clusters using 'MPI/PVM/...' require explicitly parallel code (though there are some R wrappers, more below).
- openMosix forms a 'single system image' computer and does require explicitly parallel code.
- Other approaches such as Condor or OSCAR which we won't cover here.

# Setup for PVM and MPI

## Should go into next Quantian revision

- PVM and MPI 'do not know' they are running inside.
- They want to talk to other hosts by ssh.
- PVM/MPI require distinct hostnames for all machines.
- Setup for ssh, LAM and PVM:

```
$ cp -ax /root/.ssh ~knoppix
$ chown -R knoppix.knoppix ~knoppix/.ssh
$ ifconfig                 # note $IP
$ hostname Quantian$IP
$ vi /etc/hosts              # define local hosts
$ scp /etc/hosts to_all_local_hosts
$ vi /tmp/clusterhosts              # add them
$ lamboot /tmp/clusterhosts
$ echo "conf" | pvm /tmp/clusterhosts
```

# Distributed computing: Beowulf

- Beowulf clusters use message-passing interfaces such as LAM/MPI or PVM to communicate across nodes.
- This may require a sizable amount of new programming and explicitly parallel coding. 'Hard'
- Quantian includes several Beowulf tools and libraries:
  - LAM MPI libraries and run-time;
  - Mpich MPI libraries and run-time;
  - Pvm (Parallel Virtual Machine) libraries and run-time;
  - Sprng (Scalable Parallel Random Number Generator);
  - as well as documentation as examples for these.
- Contrast: openMosix takes existing programs and moves them around nodes in the cluster to achieve optimal load across all nodes in the cluster – no alteration to algorithms, or new programming.

# Cluster computing: openMosix

- Easiest way to distribute computing load, esp. for ’embarrassingly parallel’ tasks, as the kernel schedules tasks across the cluster.
- Since release 0.3, Quantian contains a kernel with the openMosix patch as well as a set of openMosix utilities.
- As a result, “instant cluster computing” is possible based on a single dvd or iso image:
  1. boot one master instance from the dvd or hard disk,
  2. enable ’openmosixterminalserver’ from the menu,
  3. boot 1, 2, . . . ’slave’ nodes via PXE protocol (available in most recent computers) from master, and
  4. enjoy openMosix on the cluster.
- Big advantage: Identical software configuration, library versions, ... throughout the cluster.

# Cluster computing: openMosix (cont.)

- clusterKnoppix autoconfigures and autodiscovers the nodes. and enables (root) ssh access between them.
- openMosix is ideal for stand-alone programs such as 'old fashioned' C++ or Fortran apps that 'just run'.
- In general, any program without shared memory, or threads, will migrate though I/O may bring jobs back to the front node.
- Ian Latter's CHAOS projects addresses some of the security aspects by overlaying a VPN allowing for private clusters on top of public networks.
- Mix-and-match with clusterKnoppix or CHAOS is easy, any identical kernel and openMosix version can join.

# R Examples: Snow

- Tierney et al. introduced the 'Simple Network of Workstations' (SNOW) for R.
- Snow takes care of all communications, and the user concentrates on higher-level abstractions.
- Snow can use sockets, pvm or mpi to communicate, and includes support for two parallel RNG streams.
- Snow employs the existing CRAN packages rmpi, rpvm, rsprng/rlecuyer.
- Snow provides a host of functions clusterSplit, clusterCall, clusterApply, clusterApplyLB, clusterEvalQ, clusterExport, parLapply parRapply, parCapply, parApply, parMM, parSapply.

# R Examples: Snow (cont.)

- Snow example of a simple bootstrap provided by Luke Tierney.
- Example uses code from the boot package with functions and datasets from Davison & Hinkley (1997)
- Basic (non-parallel) bootstrap code:

```
library(boot)
data(nuclear)
[...]
nuke.boot <-
  boot(nuke.data, nuke.fun, R=nbBootstraps,
       m=1, fit.pred=new.fit, x.pred=new.data)
```

where `nuke.boot` is the returned bootstrap object.

# R Examples: Snow (cont.)

- This can be generalized fairly easily to work in parallel:

```
library(rsprng)
library(snow)
[...]
cl <- makeCluster(nbClusters, "MPI")
clusterSetupSPRNG(cl)
[...]
clusterEvalQ(cl, z<-library(boot))
[...]
cl.nuke.boot <-
  clusterCall(cl,boot,nuke.data, nuke.fun,
      R=round(nbBootstraps/nbClusters),
      m=1, fit.pred=new.fit, x.pred=new.data)))
```

  where `cl.nuke.boot` is a list containing the per-node returned bootstrap objects.
- Requires a little bit of extra effort to splice the list of per-node results together.

# R Examples: SnowFT

- Ševčíková and Rossini have extended Snow to allow for fault-tolerance, recovery and improved replicability in the SnowFT package.
- SnowFT provides a high-level function performParallel().
- However, SnowFT supports only PVM and not LAM/MPICH.
- The code in her example1.R does not migrate under openMosix (presumably due to I/O in the SnowFT handler) whereas the Snow example migrates well.
- However, explicitly launching PVM works.

# R Examples: papply

- Currie recently introduced papply, a parallel version of the apply function which also uses RMPI to farm out tasks. Similar to Snow, it offers a high-level abstraction complete with cluster initialization if required.
- Two simple examples are provided on the help page for papply. The shorter one is simply

```
numberLists <- lists(1:10, 4:40, 2:27)
results <- papply(numberLists, sum)
results
```

  which illustrates the elegant generalization of apply.
- Similarly, we can create arbitrary lists and functions to operate on them.

# R Examples: biopara and taskPR

- Lazar and Schoenfeld introduced biopara, a self-contained (i.e. no PVM/MPI) system for parallel code in R. It is potentially cross-platform as pure socket communications are employed. It provides a function pboot() for parallel bootstraps.
- The ASPECT Project has an initiative called Parallel R which contains a wrapper to Scalapack (RScalaPack) as well as task-R (taskPR). taskPR supports a 'parellel engine' to which expressions are submitted, and from which results can be retrieved. LAM is used as the communications mechanism.
- A detailed look at these newer contributions is beyond the scope of this talk.

# Summary

- Modern statistical computing applications (e.g. MCMC, bootstrap, boosting, . . .) require simulations.
- Also, sensitivity analysis often requires re-running similar code with slight parameter variations.
- Such ’embarassingly parallel’ problems profit greatly from a cluster: $M$ parallel runs on $N$ nodes.
- Several approaches for cluster computing are available directly from R.
- Quantian provides these approaches out of the box.

# Summary

## More information, links, . . .

http://dirk.eddelbuettel.com/quantian

http://www.quantian.org