UPPSALA
UNIVERSITET

Distributed Computing under Linux

Lars Lindbom and E. Niclas Jonsson

Division of Pharmacokinetics and Drug Therapy
Department of Pharmaceutical Biosciences
Uppsala University

Overview

- What is Distributed Computing?
  - Enabling distributed computation
- What are the potential benefits of DC in population PK/PD?
  - Example
    - Bootstrap
- Example of a small cluster
- Conclusions

What is Distributed Computing?

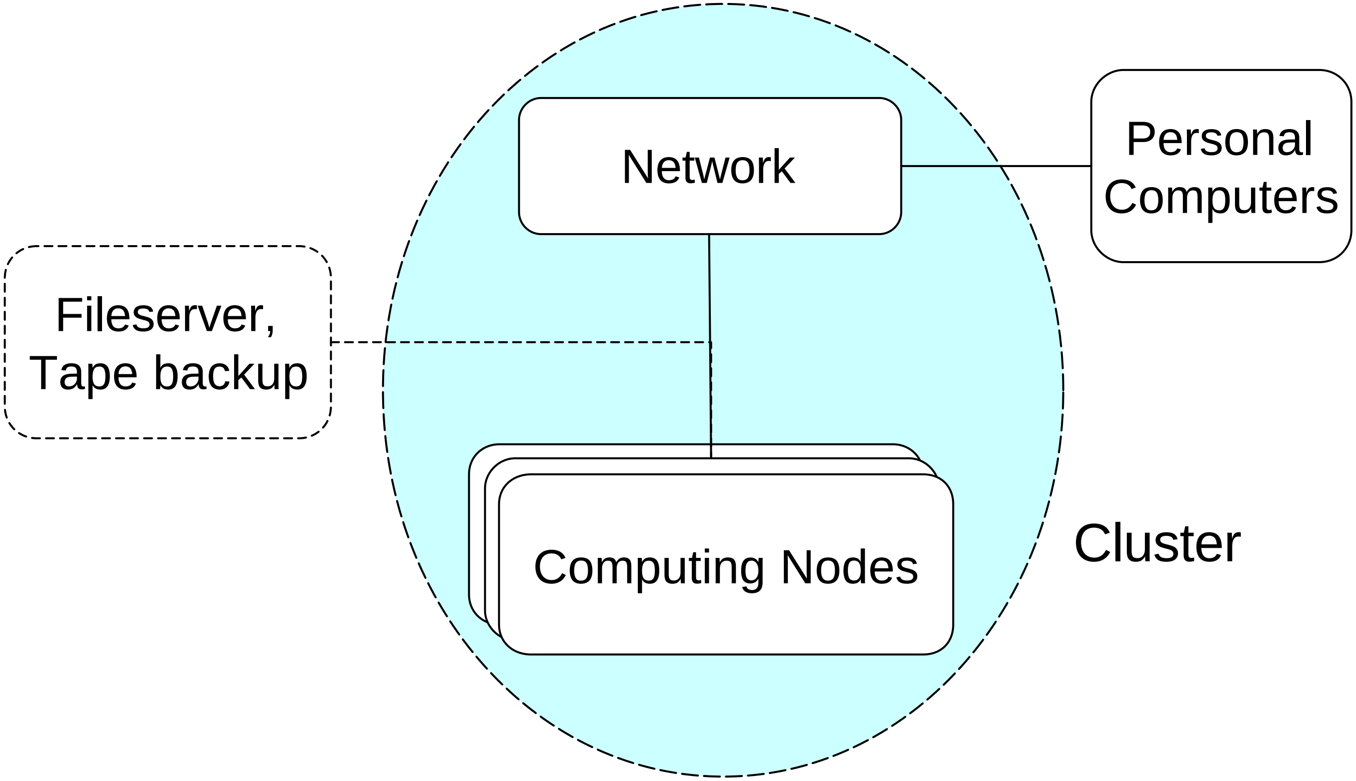
- It is using two workstations to solve one task and gaining 100% in speed
- It is using 1000 processors (CPUs) to solve a task that otherwise would have been (practically) impossible to solve.
- It is using available resources more effeciently

Enabling Distributed Computation

Hardware

Operating System

Methods (Software)



Enabling distributed computation

There is more to distributed computing than connecting computers to a common network

Hardware

Operating System

- Making the computing nodes talk to each other and cooperate

Methods (Software)

- Identifying the parts of a scientific problem that can be parallelized

What are the potential benefits of DC in population PK/PD?

- We rely on computers to fit models to data.
  - Preferably, this task should be written for the potential use of multiple processors
  - No software support this today

- There are other tasks within a population analysis involving multiple model fits where one fit not necessarily depend on a previous.
  - Model building
  - Model validation

Model Building

Automated Stepwise Covariate Model Building

Model Selection using Genetic Algorithms

Jackknife

Log-Likelihood Profiling

Cross Model Validation

Bootstrap

Model Validation

Case Deletion Diagnostics

Cross Validation

Posterior Predictive Check

J.S. Urban Hjorth, Computer Intensive Statistical Methods, (Chapman & Hall, New York, 1994)

Computer intensive methods

- Often based on repetition of nearly identical tasks that can be performed independently of each other
- Often ”embarrassingly parallel”

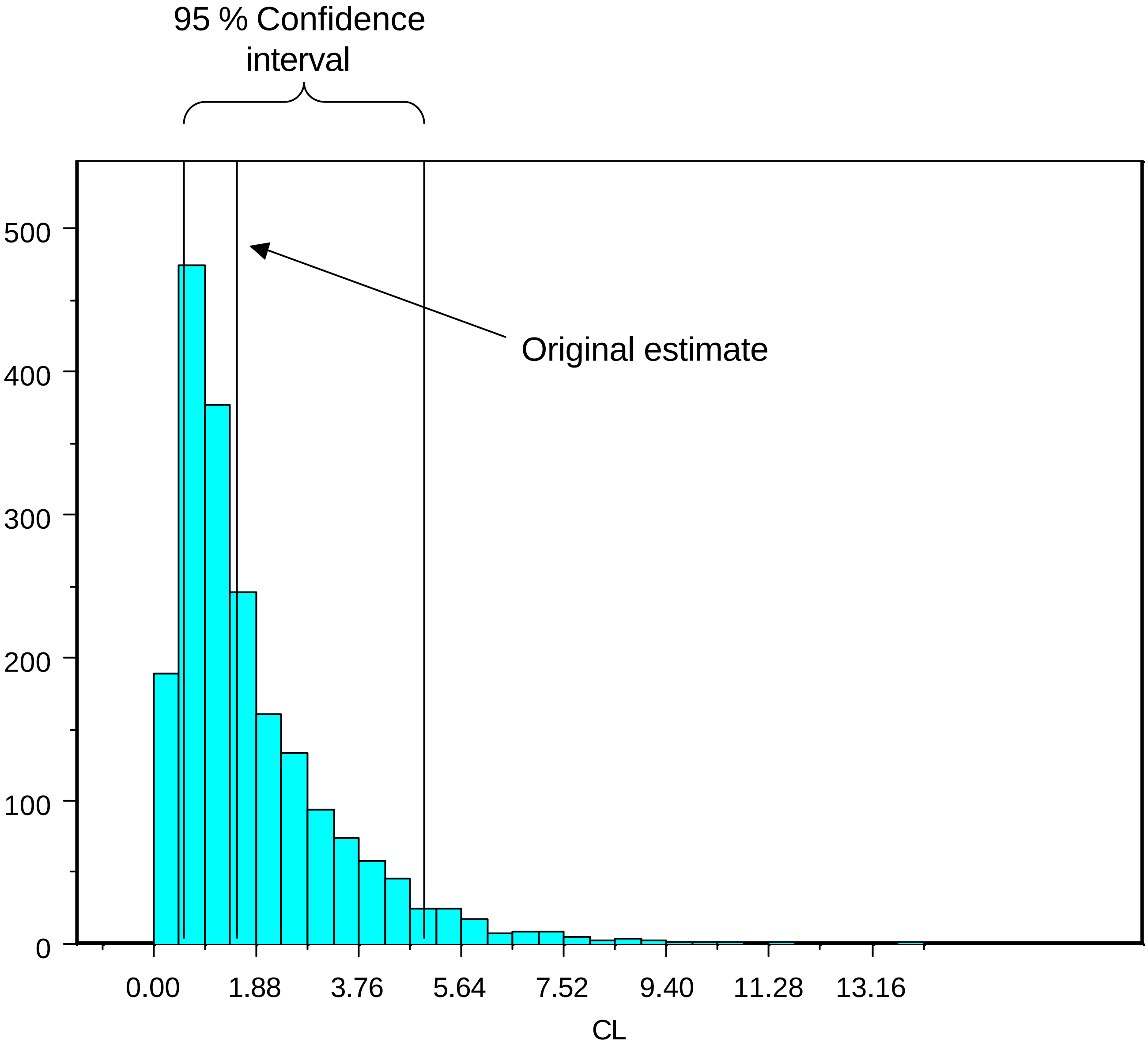
Example; The Bootstrap

- Assume that we have a model fit to a data set
- Assume further that we suspect that the true confidence interval around our estimate of clearance is non-symmetric

Bootstrap procedure:

- Draw 2000 new data sets with replacement from our data set
- Refit the model for the 2000 bootstrap samples
- Compute the confidence intervals for clearance from the distribution of the bootstrap estimates

R. T. B. Efron, An introduction to the bootstrap, (Chapman & Hall, New York, 1993)

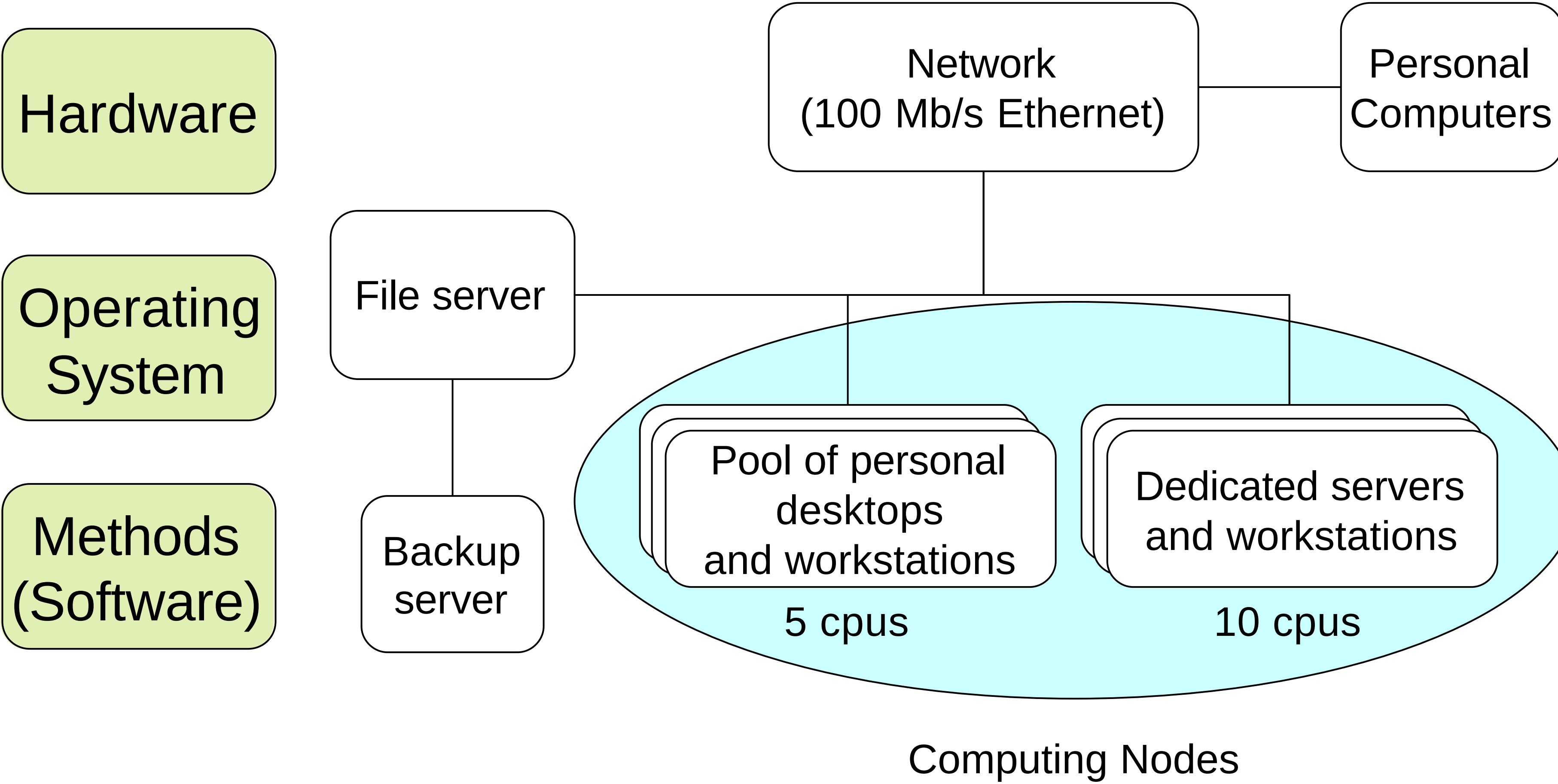

95 % Confidence
interval
Original estimate
500
400
300
200
100
0
0.00
1.88
3.76
5.64
7.52
9.40
11.28
13.16
CL

Example of a small cluster, prerequisites

The cluster should be

- Easy to set up
- Cheap (low total cost for hardware, software and administration)
- ~~Independent of operating system~~
- ~~Independent of processor architecture~~
- As far as possible independent of third party software

Example of a small cluster



Hardware

Operating System

Methods (Software)

Network (100 Mb/s Ethernet)

Personal Computers

File server

Backup server

Pool of personal desktops and workstations

5 cpus

Dedicated servers and workstations

10 cpus

Computing Nodes

Cluster enabled operating system – Linux and openMosix

Hardware

Operating System

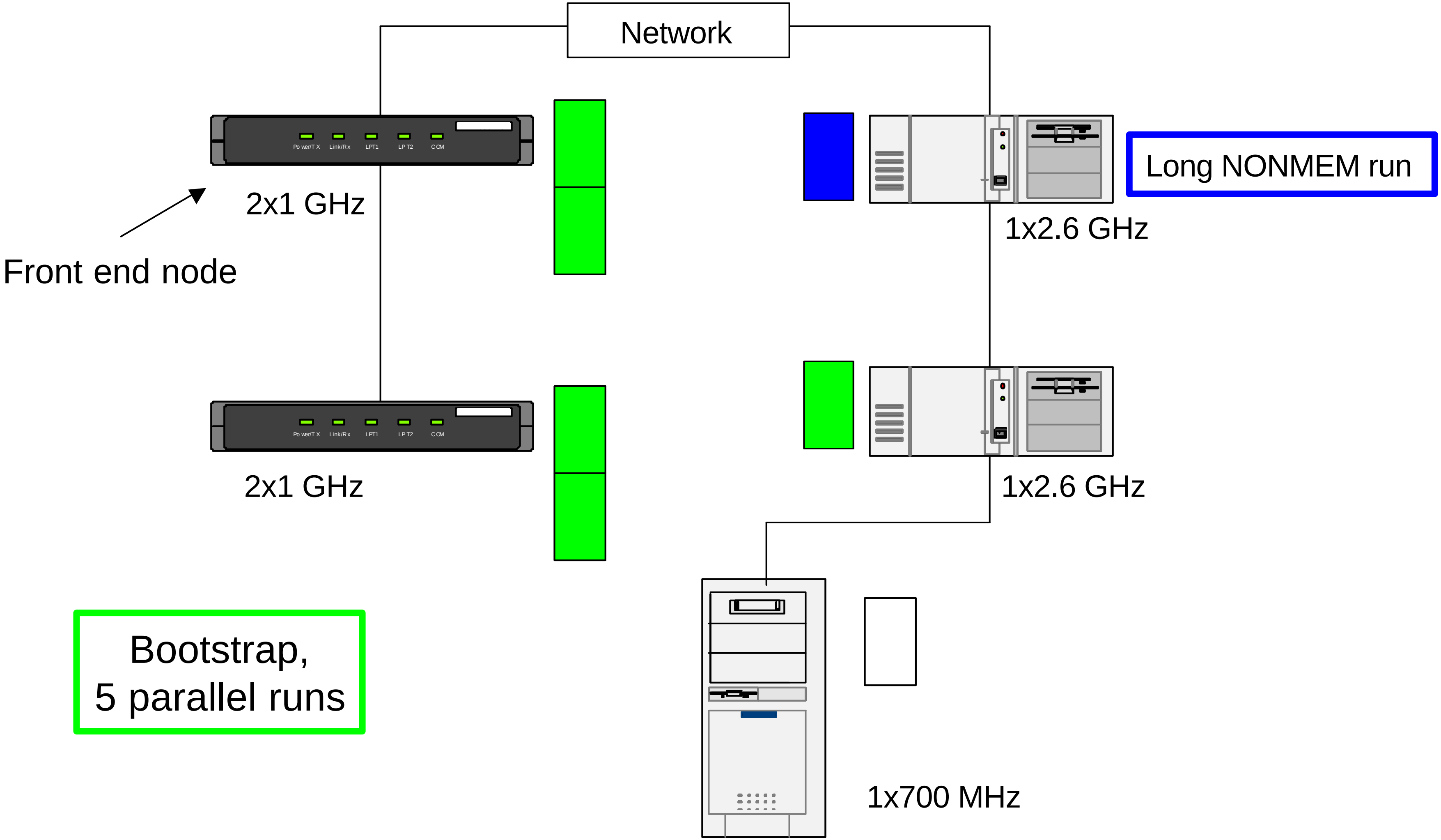
Methods (Software)

Open source operating system Linux

- Red Hat Linux 7.3

Open source Single System Image Clustering add-on to Linux

- openMosix for kernel 2.4.19



Network

2x1 GHz

Front end node

2x1 GHz

1x2.6 GHz

Long NONMEM run

1x2.6 GHz

1x700 MHz

Bootstrap,
5 parallel runs

```
110 processes: 107 sleeping, 3 running, 0 zombie, 0 stopped
CPU states: 1145.0% user,  5.8% system, 944.2% nice,  0.0% idle
Mem:  514128K av, 453064K used,  61064K free,      0K shrd,   2808K buff
Swap: 1052216K av,    324K used, 1051892K free                77664K cached

  PID USER      PRI  NI  SIZE  RSS SHARE STAT N# %CPU %MEM   TIME COMMAND
 1639 grant      11   0  1236 1236     4 S    13 99.9  0.2  77:00 nonmem
 2034 anja       19  19  1788 1788   488 S N  13 99.9  0.3   2:02 nonmem
 2035 anja       19  19  1800 1800   500 S N   3 99.9  0.3   2:05 nonmem
 2036 anja       19  19  1796 1796   496 S N  17 99.9  0.3   2:06 nonmem
 2043 anja       19  19  1796 1796   492 S N   5 99.9  0.3   2:09 nonmem
 2045 anja       19  19  1792 1792   496 S N   4 99.9  0.3   1:58 nonmem
 2046 anja       19  19  1796 1796   496 R N   0 99.9  0.3   1:38 nonmem
 2047 anja       19  19  1788 1788   492 S N   5 99.9  0.3   2:04 nonmem
 2050 anja       20  19  1788 1788   492 S N   4 98.8  0.3   2:07 nonmem
 1715 grant      17   0  1772 1772   460 S    19 98.6  0.3  59:51 nonmem
 2049 anja       19  19  1796 1796   496 S N  18 50.3  0.3   1:29 nonmem
 2044 anja       19  19  1808 1808   504 S N  18 50.2  0.3   1:50 nonmem
 2048 anja       19  19  1616 1616   320 R N   0 45.6  0.3   0:53 nonmem
 2051 root        9   0  2108 2108  1740 S     0  0.3  0.4   0:00 sshd
 2116 lasse       9   0  1040 1040   824 R     0  0.3  0.2   0:00 mtop
 2054 lasse       9   0  4212 4212  3396 S     0  0.1  0.8   0:00 gnome-terminal
    1 root        8   0   476  476   420 S     0  0.0  0.0   0:08 init
    2 root        8   0     0    0     0 SW    0  0.0  0.0   0:00 keventd
    3 root       19  19     0    0     0 SWN   0  0.0  0.0   0:00 ksoftirqd_CPU0
    4 root       19  19     0    0     0 SWN   0  0.0  0.0   0:00 ksoftirqd_CPU1
```

Benefits - Single NONMEM jobs on an openMosix cluster

Hardware

Operating System

Methods (Software)

~40 users during the past three years

At each day, 5-7 people are running jobs on the cluster

- Better usage of a heterogeneous computer pool
- The longest runs get the fastest CPUs
- Easier administration of one front end node than of many computational servers

Perl-speaks-NONMEM (PsN)

Hardware

Operating System

Methods (Software)

Computer intensive methods using NONMEM shares many common tasks:

- Opening, reading, changing, writing to and saving NONMEM-files (model files, data files and output files)
- Running NONMEM in a controlled fashion, registering the termination and analyzing the result.

PsN is intended to provide a common programming library for method development using NONMEM

Perl-speaks-NONMEM

Perl

- Has effective handling of text files
- Has good support for invoking system calls within scripts.
- Supports parallel execution
- Is platform independent

Perl-speaks-NONMEM

PsN is object oriented

- Object classes have been created, using the NONMEM files as basis
  - Model, data and output classes.
- The classes include methods for many tasks, e.g.
  - Extracting parameter estimates or termination status
  - Splitting or resampling of data
  - Changing initial estimates, etc

PsN based methods

Examples of methods developed using PsN

- Automated covariate model building
- Bootstrap
- Log-likelihood Profiling
- Case Deletion diagnostics

The current version of PsN is 2.0 and it can be obtained for free.

Own experiences and thoughts

Hardware

Operating System

Methods (Software)

The demand for system administration of a small cluster is 1/3 - 1/2 of one full time employee

Very short runs (<10 sec) do not benefit from a distributed environment – overhead of data transfer between nodes is too high

Other solutions for distributed computing exist – expanding area

OpenMosix scales well in this application to at least 20 CPUs

Conclusions

- Affordable, fairly simple solutions for distributed computing within the population PK/PD area exist

- A distributed computing setup is (presently) a pre-requisite for the use of computer intensive methods in population PK/PD

References

openMOSIX Homepage, http://openmosix.sourceforge.net

Red Hat Linux Homepage, http://www.redhat.com

PsN, Lars.Lindbom@farmbio.uu.se