A Cloud-Based Cluster Computing Platform for Pharmacometric Applications

Timothy Bergsma, William Knebel, Daniel Polhamus, Jeffrey T. Hane, Marc R. Gastonguay
Metrum Research Group, LLC

METRUM RESEARCH GROUP

Objectives

Cloud computing creates a new opportunity to improve computational flexibility for pharmacometrics. Methodologically, the **scale** of the system can be closely matched to the problem. Administratively, creation of a quality-assured production system can be **scripted**, for easy re-deployment or adaptation.

We sought to design a scalable, scripted phamacometrics platform with good feature support for two widely-received pharmacometric applications: NONMEM® [1] and OpenBUGS [2]. We envisioned that deployment by means of version-controlled scripts would improve **quality assurance** while reducing **maintenance** effort. Additionally, we intended that exploitation of scale options would provide superior computational **power** where required, while minimizing **cost** where not.

Methods

OpsCode Chef [3] software was used to develop a machine image for Amazon Web Services [4]. Chef allowed fully scripted hardware abstraction for systematic version control of the image during development. The image itself was deployed simultaneously as a cluster of virtual machines using StarCluster [5] control software residing on the user's workstation (Figure 1). The cluster was then attached to a company server.

On the image, the following were pre-installed:

- NMQual 8.1.3 [6],
- Intel Fortran compiler 12.0.4 [7],
- NONMEM 7.2.0, Enterprise license (using Intel Fortran compiler),
- OpenBUGS 3.2.1,
- R 2.15.1 [8],
- qapply 0.04 [9],
- Sun Grid Engine 6.2 [10],
- and RStudio Server 0.96.316 [11].

During deployment, the user had the opportunity to specify both the speed and number of machines in the cluster, given problem-specific demands for performance and parallelization.

Post-deployment, the R packages

- metrumrg 5.24 (or greater) [12] and
- R2OpenBUGS 3.2-1.4 [13]

were installed to provide NONMEM® and OpenBUGS connectivity, respectively. In a typical workflow (Figure 2), the user browses to RStudio Server, authenticates, loads appropriate packages, and invokes NONMEM® or OpenBUGS. Pharmacometric applications were invoked interactively or by means of scripted commands, per package documentation. Features in the R packages metrumrg and qapply adequately supported parallelization by means of Sun Grid Engine for NONMEM® and OpenBUGS, respectively. Parallelization was supported across runs, and within runs – by individual (NONMEM®) or by Markov Chain (OpenBUGS). Machines were added or removed from the active cluster as desired.

References

[1] Beal S, Sheiner LB, Boeckmann A, Bauer RJ. (2009). *NONMEM User's Guides.* Icon Development Solutions, Ellicott City, MD, USA.

[2] Lunn D, Spiegelhalter D, Thomas A, Best N (2009). *The BUGS project: Evolution, critique, and future directions.* Statistics in Medicine 28: 3049-3067.

[3] http://www.opscode.com/chef

[4] Fusaro VA, Patil P, Gafni E, Wall DP, Tonellato PJ (2011). *Biomedical Cloud Computing With Amazon Web Services.* PLoS Comput Biol 7(8): e1002147. http://www.ploscompbiol.org/article/info%3Adoi%2F10.1371%2Fjournal.pcbi.1002147

[5] http://web.mit.edu/stardev/cluster

[6] http://nmqual.googlecode.com

[7] Knebel W, Bergsma TT, Dagdigian C, Hane J, Gastonguay MR (2010). *Utilizing NONMEM 7 and the Intel Fortran Compiler in a Distributed Computing Environment.* Population Approach Group Europe. http://www.page-meeting.org/pdf_assets/6315-PageNM7IFCposter.pdf

[8] R Core Team (2012). *R: a language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0. http://www.R-project.org

[9] Dan Polhamus (2012). *qapply: (Open) Grid Engine Parallel lapply.* R package version 0.04. http://qapply.googlecode.com

[10] Gentzsch W (2001). *Sun Grid Engine: Towards Creating a Compute Power Grid.* Proceedings of the 1st International Symposium on Cluster Computing and the Grid p. 35. IEEE Computer Society Washington, DC, USA. http://dl.acm.org/citation.cfm?id=792378

[11] http://www.rstudio.com/ide/download/server

[12] Bergsma TT, Knebel W, Fisher J, Gillespie WR, Riggs MM, Gibiansky L, Gastonguay MR (2013). *Facilitating pharmacometric workflow with the metrumrg package for R.* Computer Methods and Programs in Biomedicine 109: 77-85. http://www.sciencedirect.com/science/article/pii/S0169260712001915

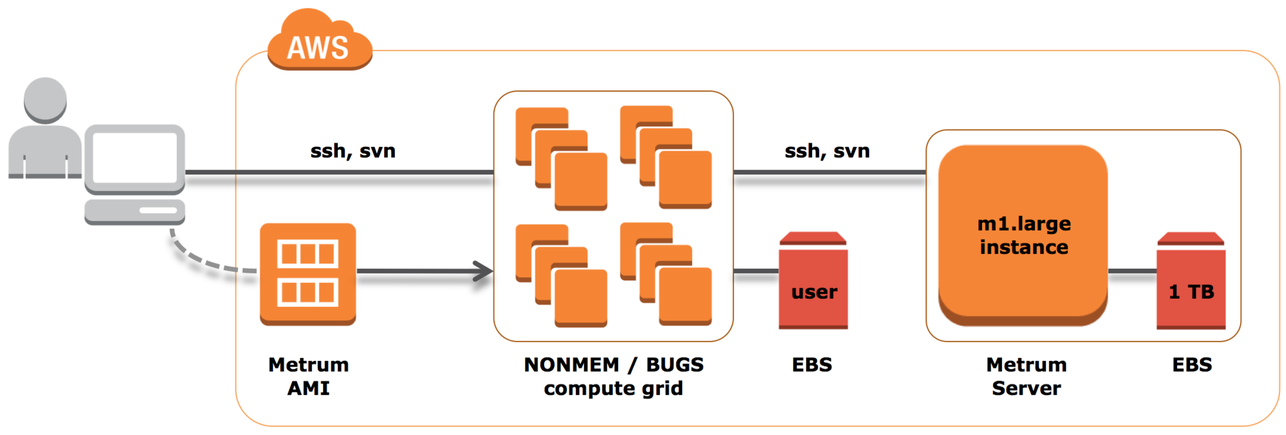
[13] Sturtz S, Ligges U, Gelman A (2005). *R2WinBUGS: A Package for Running WinBUGS from R.* Journal of Statistical Software 12(3): 1-16. http://cran.r-project.org/web/packages/R2OpenBUGS

[14] http://subversion.apache.org/

Figure 1

Cloud configuration.

The Amazon machine image (AMI) and the company server are created directly from version-controlled scripts. Per details from the user, one or more instances of the AMI are deployed as a cluster hosting a compute grid. The user's persistent elastic block storage (EBS) is attached, and encrypted connections are formed among user, cluster, and server – which holds Subversion [14] project repositories. Modeling inputs and outputs are moved between cluster and server using Subversion operations. The user can also connect directly to the server (not shown).



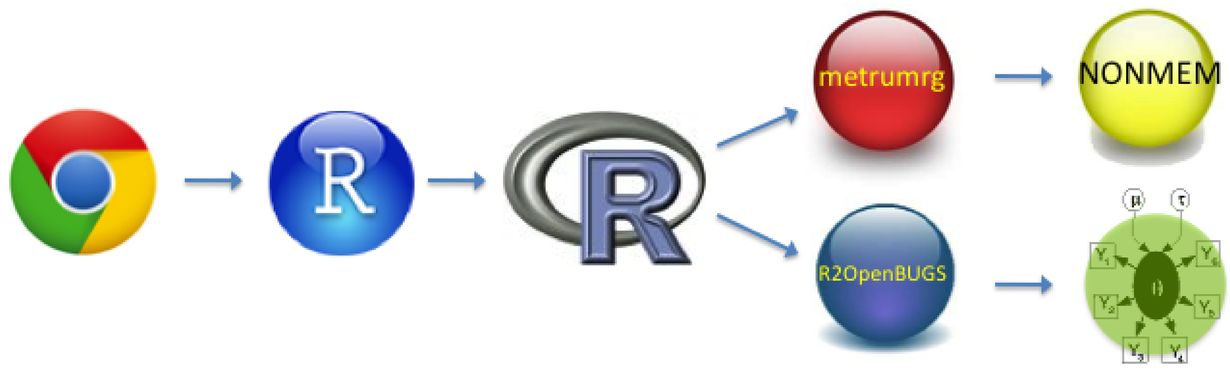
Results

The resulting platform exceeded our expectations for a production modeling and simulation environment.

- Deployment **scripts** made it easy to create qualified production clusters on demand, both for individual scientists and for clients. Wait-times for colleague-shared machines were eliminated.
- Cluster **scale** was well-matched to project needs in terms of speed and size. Modelers noted that simultaneous evaluation of several models encouraged model complexity and comparison. Gains were realized for within-run parallelization as well. Run times for representative models under NONMEM® 7.2 dropped substantially with even modest parallelization relative to single-core execution (Figure 3).
- Use of version-controlled machine images reduced **maintenance** burden while enhancing **quality assurance**: the image represented a well-defined management target that could be systematically characterized and readily reproduced with a complete audit trail. Images could be implemented as either fixed persistent or elastic cloud computing (EC2) instances and could – in the event of a service outage – be efficiently cloned in a different service region.
- Operational **cost** was reduced relative to conventional hardware. Fees were based only on computational time and block storage used. Cluster sizes were achieved that would have been cost-prohibitive under a full-ownership hardware model.
- Extensive use of open-source components made this solution easy to adopt. The following are freely available: RStudio, R, metrumrg, R2OpenBUGS, OpenBUGS, NMQual, Sun Grid Engine, Chef, and StarCluster.

Figure 2

Sofware configuration. The user points a web browser at an instance of RStudio Server on the master node of the cluster. RStudio hosts the R environment. The packages metrumrg and R2OpenBUGS are loaded, serving as connectors to NONMEM® and OpenBUGS, respectively.



Conclusion

A scripted, cloud-based cluster computing solution enabled scalable use of pharmacometric applications in a multi-user environment. Full feature support, ease of maintenance, and superior quality assurance were accommodated. Operating costs were less than conventional, and the design made extensive use of off-the-shelf components. As implemented, this platform practically eliminated computation time from the critical path of completion for typical pharmacometric projects.

Figure 3

Run times for two contrasting models with varying levels of parallelization under NONMEM 7.2.0. To illustrate effects of parallelization on run times, two models were executed with varying numbers of cores. Both models were run using NONMEM ® 7.2.0 with PREDPP subroutine ADVAN6. The first was a dual linear/nonlinear absorption model for 1000 subjects; the second was a PKPD model for 70 subjects. 1, 8, 16, 24, 48, and (for 1000 subjects) 96 cores were attempted. Even modest amounts of parallelization greatly reduced run times. Most of the benefit was achieved with far fewer cores than the number of subjects.