batchtools: Tools for R to work on batch systems

Michel Lang[1], Bernd Bischl[2], and Dirk Surmann[1]

[1]TU Dortmund University
[2]LMU Munich

9 November 2016

Paper DOI: http://dx.doi.org/10.21105/joss.00135
Software Repository: https://github.com/mllg/batchtools
Software Archive: http://dx.doi.org/10.5281/zenodo.293835

Summary

The `R` (R Core Team 2016) package `batchtools` is the successor of the `BatchJobs` package (Bischl et al. 2015). It provides an implementation of a Map-like operation to define and asynchronously execute jobs on a variety of parallel backends:

- Local (blocking) execution in the current `R` session or in an externally spawned `R` process (intended for debugging and prototyping)
- Local (non-blocking) parallel execution using `parallel`'s multicore backend (R Core Team 2016) or `snow`'s socket mode (Tierney et al. 2016).
- Execution on loosely connected machines using SSH (including basic resource usage control).
- Docker Swarm
- IBM Spectrum LSF
- OpenLava
- Univa Grid Engine (formerly Oracle Grind Engine and Sun Grid Engine)
- Slurm Workload Manager
- TORQUE/PBS Resource Manager

Extensibility and user customization are important features as configuration on high-performance computing clusters is often heavily tailored towards very specific requirements or special hardware. Hence, the interaction with the schedulers uses a template engine for improved flexibility. Furthermore, custom functions can be hooked into the package to be called at certain events. As a last resort, many utility functions simplify the implementation of a custom cluster backend from scratch.

The communication between the master `R` session and the computational nodes is kept as simple as possible and runs completely on the file system which greatly simplifies the extension to additional parallel platforms. The `data.table` package (Dowle et al. 2015) acts as an in-memory database to keep track of the computational status of all jobs. Unique job seeds ensure reproducibility across systems, log files can conveniently be searched using regular expressions and jobs can be annotated with arbitrary tags. Jobs can be chunked (i.e., merged into one technical cluster job) to be executed as one virtual job on a node (executed sequentially or using multiple local CPUs) in order to reduce the overhead induced by job management and starting/stopping `R`. All in all, the provided tools allow users to work with many thousands or even millions of jobs in an organized and efficient manner.

The `batchtools` package also comes with an abstraction mechanism to assist in conducting large-scale computer experiments, especially suited for (but not restricted to) benchmarking and exploration of algorithm performance. The mechanism is similar to `BatchExperiments` (Bischl et al. 2015) which `batchtools` now also supersedes: After defining the building blocks of most computer experiments, problems and algorithms, both can be parametrized to define jobs which are then in a second step submitted to one of the parallel backends.

Important changes to its predecessors are summarized in a vignette to help users of `BatchJobs`/`BatchExperiments` migrating their cluster configuration and aid the transition to `batchtools`.

References

Bischl, Bernd, Michel Lang, Olaf Mersmann, Jörg Rahnenführer, and Claus Weihs. 2015. "BatchJobs and BatchExperiments: Abstraction Mechanisms for Using R in Batch Environments." *Journal of Statistical Software* 64 (11): 1–25. doi:10.18637/jss.v064.i11.

Dowle, M, A Srinivasan, T Short, S Lianoglou with contributions from R Saporta, and E Antonyan. 2015. *Data.table: Extension of Data.frame.* https://CRAN.R-project.org/package=data.table.

R Core Team. 2016. *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Tierney, Luke, A. J. Rossini, Na Li, and H. Sevcikova. 2016. *Snow: Simple Network of Workstations.* https://CRAN.R-project.org/package=snow.