batchtools: Tools for R to work on batch systems

9 November 2016

Summary

The R [@R] package batchtools is the successor of the BatchJobs package [@batchjobs_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 R's parallel's multicore backend [@R] or snow's socket mode [@snow].
- Execution on loosely connected machines using SSH (with 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 possible user customization are important features as configuration on HPC clusters is often heavily tailored towards very specific requirements or special hardware. Hence, the interaction with the different schedulers is templated 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 [@data_table] 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. BatchJobs uses a SQLite database for this synchronization, sometimes resulting in locking problems when very many processes try to access the database at the same time. Jobs can be chunked (i.e., merged into one

technical cluster job) to be executed sequentially or in parallel using multiple cores of the computational node in order to reduce the overhead induced by job management and starting/stopping `R`. All in all, the provided tools allow 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 [@batchtools_2015] which `batchtools` now also supersedes: After defining problems and algorithms, both can be parametrized with arbitrary parameters to define jobs which are then in a second step submitted to one of the parallel backends.

`batchtools` can also be controlled through the `parallelMap` package, and is offered as a backend in that package, which mainly focuses on enabling R users to easily write flexible parallel loops / mapping operations in their own R packages.

todo: dirk fehlt?

References