The targets R package: a dynamic Make-like function-oriented pipeline toolkit for reproducibility and high-performance computing

William Michael Landau

1 Eli Lilly and Company

Summary

The targets R package (Landau, 2020) is a pipeline toolkit for computationally intense reproducible research. It reduces the time and effort required to develop a data analysis project and maintain a trustworthy set of results. targets uses static code analysis to detect dependency relationships among interconnected computational tasks and construct a directed acyclic graph (DAG), which researchers can visualize in order to understand and communicate the structure of a complicated workflow. To run the pipeline at scale, targets leverages implicit parallel computing and optional cloud storage. In subsequent runs, targets, skips tasks that are already synchronized with their upstream dependencies, which not only reduces the runtime of rapidly developing workflows, but also provides tangible evidence of reproducibility.

In high-performance computing scenarios, targets uses its DAG to discern which targets can run concurrently and which targets are still waiting for other upstream targets to finish processing. As soon as a target’s dependency requirements are met, the target is deployed to the next available parallel worker. Internally, targets leverages the clustermq package (Schubert, 2019) for persistent workers and the future package (Bengtsson, 2020) for transient workers. Both clustermq and future are powerful and versatile frameworks capable of submitting R workloads not only to multiple cores on a single machine, but also to popular resource managers on shared computing clusters.

targets is the successor to drake (Landau, 2018), which in turn originated from remake (FitzJohn, 2017), an R package modeled after GNU Make (Stallman, 1998). Unlike Make, targets and drake and remake focus on the R language, encourage an idiomatic function-oriented style of programming, and abstract each target as an R object. Relative to remake and drake, targets is friendlier and more efficient, surpassing the permanent architectural limitations of both predecessors. The data storage system of targets is lighter and more transparent, which helps users diagnose issues, move projects to different file systems, work with multiple contributors, and leverage seamless Metaflow-like cloud storage integration (Ge & Goyal, n.d.). In addition, targets supports stronger user-side guardrails, more introspective dependency graph visualizations, parallel efficient dynamic branching, and an interface more amenable to metaprogramming and third-party extensions.

References


