future.p2p: Peer-to-Peer Compute Clusters in R

Henrik Bengtsson

Palabras clave: R - parallel processing - distributed processing - peer-to-peer - p2p - futures

TL;DR

Want to team up with friends or research collaborators and build a world-wide virtual compute cluster using your idle computers? The **future.p2p** package in R lets you do exactly that.

Abstract

Parallel processing can speed up computationally intensive tasks. As the size of these tasks grows — and as access to more CPU cores increases – so does the demand for parallel-processing solutions. In R, the futureverse ecosystem has, over the past decade, become a popular go-to for parallel and distributed computing. Its simplicity and strict, minimalistic application programming interface (API) are key reasons for its success. Many R packages use futureverse internally for parallelization, making it extremely easy for end-users to scale up their analyses without needing to understand the technical details of running code in parallel.

With futureverse, users have many options for where their parallel R code runs. Anyone can parallelize on their personal notebook or any machine they have access to. Users with accounts on multiple machines can easily parallelize across them – as long as they have SSH access – e.g., office desktops, remote servers, or cloud instances. Those with access to high-performance computing (HPC) clusters can run R code in parallel via job schedulers like Slurm and SGE.

But not all users have access to powerful machines or computing environments. Some lose access to high-end compute resources when they finish school or leave a job. For them, running certain analyses becomes impractical – code might take days or even weeks to complete. The **future.p2p** package aims to solve this problem. It lets users come together and build a global, peer-to-peer (P2P) compute cluster. This way, they can share and harness compute power among one another. Setup is minimal and requires only basic knowledge of SSH and R.

Availability: https://future.p2p.futureverse.org/.

Introduction

The futureverse framework, a mature and widely adopted solution over the past decade, significantly simplifies the parallelization of computationally intensive R tasks, offering a minimalistic application programming interface (API) that shields users from the underlying complexities of parallel execution. At its core is the **future** package. While the futureverse provides robust capabilities for scaling R code across various environments, including local machines, remote servers, cloud instances, and high-performance computing (HPC) clusters, a significant challenge persists for users who lack consistent access to these powerful computational resources. Such limitations can render complex analyses impractical, extending processing times from hours to weeks. **future.p2p** directly addresses this limitation by enabling the construction of a global, peer-to-peer compute cluster, allowing users to collaboratively share and leverage distributed computational power.

A **future.p2p** cluster comprise on two fundamental components:

- *A task message board*: This centralized, lightweight component serves as a 'forum' where clients post computational tasks (futures), and available workers offer to process them. This board manages metadata related to task assignment.

- *A peer-to-peer file-transfer backend*: This decentralized component is responsible for the secure, direct transfer of futures and their results between clients and workers, bypassing the message board for data exchange.

To participate in a P2P cluster, users must possess an SSH key pair and a P2P account. The account grants access to the task message board, while P2P file transfers between peers are anonymous and do not require individual accounts.

future.p2p supports two primary modes of operation:

- *Personal P2P Clusters*: An individual user with access to multiple machines can configure a personal cluster to distribute R tasks among their own computational resources, locally or remotely.
- *Shared P2P Clusters*: Users can establish a shared cluster with friends or collaborators, where one user acts as the host, controlling access for designated members.

All authorized users, including the host, can then contribute their idle computational resources as P2P workers to the cluster by executing `future.p2p::worker()`. Once workers are active, any user with access can use the cluster for parallel processing by setting the future plan as illustrated in:

```
library(future)

## Connect to a shared P2P cluster
plan(future.p2p::cluster, cluster = "alice/friends")

## Parallelized version of purrr::map()
y <- furrr::future_map(xs, slow_fcn)
```

A critical consideration in the **future.p2p** framework is security. The framework does *not* prevent a user within a P2P cluster from submitting malicious R code to a worker. Such code could attempt to delete files (e.g. `future(system("erase-harddrive"))`) or access sensitive data on the worker machine. Because of this, users should only join shared P2P clusters where all members trust each other. To mitigate security risks further, one could potential launch the P2P workers in sandboxed environments, e.g. virtual machines, Linux containers, or via dedicated sandboxing tools.

We believe **future.p2p** represents a step towards democratizing access to distributed computing resources in R, empowering users to overcome the limitations of individual machine capacity through collective compute power.