

Package: mappp (via r-universe)

September 10, 2024

Title Map in Parallel with Progress

Version 1.0.0

Description Provides one function, which is a wrapper around `purrr::map()` with some extras on top, including parallel computation, progress bars, error handling, and result caching.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports memoise, progress, pbmcapply, parallel, parallelly, purrr, rlang

RoxygenNote 7.1.1

URL <https://github.com/cole-brokamp/mappp>

BugReports <https://github.com/cole-brokamp/mappp/issues>

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Repository <https://cole-brokamp.r-universe.dev>

RemoteUrl <https://github.com/cole-brokamp/mappp>

RemoteRef HEAD

RemoteSha aab1390575f96602db5aefd03fe4c5eeb7819ff2

Contents

mappp	2
Index	4

mappp

map in parallel with progress

Description

This function is a wrapper around `purrr::map()` (which applies a function to each element of a list or atomic vector) with some extras on top, including parallel computation, progress bar, error handling, and result caching.

Usage

```
mappp(
  .x,
  .f,
  parallel = FALSE,
  cache = FALSE,
  cache_name = "cache",
  error_capture = TRUE,
  error_quiet = TRUE,
  num_cores = NULL
)
```

Arguments

<code>.x</code>	list or vector of objects to apply over
<code>.f</code>	function to apply; allows for compact anonymous functions (see <code>rlang::as_function()</code> for details)
<code>parallel</code>	logical; use parallel processing?
<code>cache</code>	defaults to <code>FALSE</code> , which means no cache used. If <code>TRUE</code> , cache the results locally in a folder named according to <code>cache_name</code> using the <code>memoise</code> package
<code>cache_name</code>	a character string to use a custom cache folder name (e.g. "my_cache"); defaults to "cache"
<code>error_capture</code>	apply function to all elements and return those that error as <code>NA</code> ; this also messages user with name/index of offending element and resulting error message
<code>error_quiet</code>	quiet individual error messages when capturing error messages? or show them as they occur?
<code>num_cores</code>	the number of cores used for parallel processing. Can be specified as an integer, or it will guess the number of cores available with <code>parallelly::availableCores()</code> . won't have an effect if <code>parallel</code> is <code>FALSE</code>

Details

`mappp` is designed for long computations and as such it always uses a progress bar, and always returns a list. Long computations shouldn't worry about being type strict; instead, extract results in the right type from the results list.

A progress bar will be shown in the terminal using an interactive R session or in an .Rout file, if using R CMD BATCH and submitting R scripts for non-interactive completion. Although R Studio supports the progress bar for single process workers, it has a problem showing the progress bar if using parallel processing (see the discussion at <http://stackoverflow.com/questions/27314011/mcfork-in-rstudio>). In this specific case (R Studio + parallel processing), text updates will be printed to the file `‘.progress’`. Use a shell and `‘tail -f .progress’` to see the updates.

Value

a list the same length as `.x`

Examples

```
X <- list("x" = 100, "y" = "a", "z" = 200)
slow_log <- function(.x) {
  Sys.sleep(0.5)
  log(.x)
}
# by default returns NA on error
mapp(X, slow_log)
# when not using error, entire calculation will fail
# mapp(X, slow_log, error_capture = FALSE)
# showing error messages when they occur rather than afterwards can be useful
# but will cause problems with progress bar displays
mapp(X, slow_log, error_quiet = FALSE)
```

Index

mapp, [2](#)