

Package ‘delayed’

October 20, 2022

Title A Framework for Parallelizing Dependent Tasks

Version 0.4.0

Description Mechanisms to parallelize dependent tasks in a manner that optimizes the compute resources available. It provides access to ``delayed`` computations, which may be parallelized using futures. It is, to an extent, a facsimile of the 'Dask' library (<<https://www.dask.org/>>), for the 'Python' language.

Depends R (>= 3.2.0)

Imports R6, igraph, future, rstackdeque, rlang, data.table, visNetwork, uuid, BBmisc, progress, R.utils, R.oo

Suggests testthat, knitr, rmarkdown, shiny

License GPL-3

URL <https://tlverse.org/delayed/>

BugReports <https://github.com/tlverse/delayed/issues>

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.2.0

NeedsCompilation no

Author Jeremy Coyle [aut, cre, cph] (<<https://orcid.org/0000-0002-9874-6649>>),
Nima Hejazi [ctb] (<<https://orcid.org/0000-0002-7127-2789>>)

Maintainer Jeremy Coyle <jeremyrcoyle@gmail.com>

Repository CRAN

Date/Publication 2022-10-19 22:25:09 UTC

R topics documented:

| | |
|--------------------|---|
| Delayed | 2 |
| delayed | 2 |
| eval_delayed | 3 |
| find_delayed_error | 3 |

| | |
|------------------------------|---|
| FutureJob | 4 |
| plot.Delayed | 4 |
| plot_delayed_shiny | 5 |
| Scheduler | 5 |
| SequentialJob | 6 |

| | |
|--------------|----------|
| Index | 7 |
|--------------|----------|

| | |
|---------|--|
| Delayed | <i>Delayed class that manages dependencies and computes when necessary</i> |
|---------|--|

Description

Delayed class that manages dependencies and computes when necessary

Examples

```
d <- delayed(3 + 4)
methods::is(d, "Delayed")
d$compute()
```

| | |
|---------|---|
| delayed | <i>Generates Delayed Version of an Expression</i> |
|---------|---|

Description

A Delayed version of a function may be called to generate Delayed objects

Usage

```
delayed(expr, sequential = FALSE, expect_error = FALSE, timeout = NULL)
```

```
delayed_fun(fun, sequential = FALSE, expect_error = FALSE)
```

Arguments

| | |
|--------------|--|
| expr | expression to delay |
| sequential | if TRUE, never parallelize this task |
| expect_error | if TRUE, pass error to downstream tasks instead of |
| timeout | specify a time limit for computation halting computation |
| fun | function to delay |

Examples

```
d <- delayed(3 + 4)
d$compute()
adder <- function(x, y) {
  x + y
}
delayed_adder <- delayed_fun(adder)
z <- delayed_adder(3, 4)
z$compute()
```

| | |
|--------------|--|
| eval_delayed | <i>Helper Function to Evaluate Delayed</i> |
|--------------|--|

Description

Helper Function to Evaluate Delayed

Usage

```
eval_delayed(to_eval, timeout = Inf)
```

Arguments

| | |
|---------|--|
| to_eval | a list as generated from Delayed\$prepare_eval() |
| timeout | a timeout indicating when to terminate the job |

| | |
|--------------------|------------------------------------|
| find_delayed_error | <i>Find error in delayed chain</i> |
|--------------------|------------------------------------|

Description

Searches through a network of delayed objects for the first object with state "error"

Usage

```
find_delayed_error(delayed_object)
```

Arguments

| | |
|----------------|---------------------------------------|
| delayed_object | the object in which an error occurred |
|----------------|---------------------------------------|

Examples

```
delayed_error <- delayed_fun(stop)
error_message <- "this is an error"
broken_delayed <- delayed_error(error_message)
broken_delayed$expect_error <- TRUE
result <- broken_delayed$compute()
```

 FutureJob

Future Delayed Jobs

Description

A Job that leverages the future framework to evaluate asynchronously.

Examples

```
library(future)
plan(multicore, workers = 1)
d <- delayed(3 + 4)
sched <- Scheduler$new(d, FutureJob, nworkers = 1)
```

plot.Delayed

Plot Method for Delayed Objects

Description

Plot Method for Delayed Objects

Usage

```
## S3 method for class 'Delayed'
plot(x, color = TRUE, height = "500px", width = "100%", ...)
```

Arguments

| | |
|--------|---|
| x | An object of class Delayed for which a task dependency graph will be generated. |
| color | If TRUE, color-code nodes according to status, and display legend |
| height | passed to visNetwork |
| width | passed to visNetwork |
| ... | Additional arguments (passed to visNetwork). |

Examples

```
adder <- function(x, y) {
  x + y
}
delayed_adder <- delayed_fun(adder)
z <- delayed_adder(3, 4)
z2 <- delayed_adder(z, 4)
z2$sequential <- TRUE
z3 <- delayed_adder(z2, z)
plot(z3)
```

plot_delayed_shiny *Animated Representation a Task Dependency Structure*

Description

uses shiny

Usage

```
plot_delayed_shiny(scheduler)
```

Arguments

scheduler the scheduler to animate

Examples

```
## Not run:
adder <- function(x, y) {
  x + y
}
delayed_adder <- delayed_fun(adder)
z <- delayed_adder(3, 4)
z2 <- delayed_adder(z, 4)
z2$sequential <- TRUE
z3 <- delayed_adder(z2, z)
plot_delayed_shiny(z3)

## End(Not run)
```

Scheduler *Scheduler class that orders compute tasks and dispatches tasks to workers*

Description

Scheduler class that orders compute tasks and dispatches tasks to workers

Examples

```
d <- delayed(3 + 4)
sched <- Scheduler$new(d, SequentialJob)
sched$compute()
```

SequentialJob

Sequential Delayed Jobs

Description

A Job that will evaluate immediately (i.e., in a sequential fashion), blocking the current process until it completes.

Examples

```
d <- delayed(3 + 4)
sched <- Scheduler$new(d, SequentialJob)
```

Index

Delayed, [2](#)
delayed, [2](#)
delayed_fun (delayed), [2](#)

eval_delayed, [3](#)

find_delayed_error, [3](#)
FutureJob, [4](#)

plot.Delayed, [4](#)
plot_delayed_shiny, [5](#)

Scheduler, [5](#)
SequentialJob, [6](#)