

Package ‘atime’

April 3, 2023

Type Package

Title Asymptotic Timing

Version 2023.3.31

Description Computing and visualizing comparative asymptotic timings of different algorithms and code versions. Also includes functionality for comparing empirical timings with expected references such as linear or quadratic, https://en.wikipedia.org/wiki/Asymptotic_computational_complexity Also includes functionality for measuring asymptotic memory and other quantities.

License GPL-3

URL <https://github.com/tdhock/atime>

BugReports <https://github.com/tdhock/atime/issues>

Imports data.table, bench, lattice, git2r, utils, stats, grDevices

Suggests directlabels, ggplot2, testthat, knitr, markdown, stringi, re2, binsegRcpp, wbs, fpop, changepoint, LOPART, cumstats, PeakSegDisk, callr, readr, dplyr, tidyr, nc, RColorBrewer

VignetteBuilder knitr

NeedsCompilation no

Author Toby Hocking [aut, cre]

Maintainer Toby Hocking <toby.hocking@r-project.org>

Repository CRAN

Date/Publication 2023-04-03 10:50:08 UTC

R topics documented:

atime	2
atime_grid	3
atime_pkg	5
atime_versions	5

atime_versions_exprs	7
atime_versions_remove	8
glob_find_replace	9
references_best	10

Index	12
--------------	-----------

atime	<i>Asymptotic timing</i>
-------	--------------------------

Description

Computation time and memory for several R expressions of several different data sizes.

Usage

```
atime(
  N, setup, expr.list=NULL, times=10, seconds.limit=0.01, verbose=FALSE,
  result=FALSE, ...)
```

Arguments

N	numeric vector of data sizes to vary.
setup	expression to evaluate for every data size, before timings.
expr.list	named list of expressions to time.
times	number of times to evaluate each timed expression.
seconds.limit	if the median timing of any expression exceeds this many seconds, then no timings for larger N are computed.
verbose	logical, print messages after every data size?
result	logical, save each result?
...	named expressions to time.

Details

Each iteration involves first computing the setup expression, and then computing several times the ...expressions. For convenience, expressions may be specified either via code (...) or data (expr.list arg).

Value

list of class atime with elements seconds.limit (numeric input param), measurements (data table of results).

Author(s)

Toby Dylan Hocking

Examples

```
## Example 1: polynomial and exponential time string functions.
string.result <- atime::atime(
  N=unique(as.integer(10^seq(0,3.5,l=100))),
  setup={
    subject <- paste(rep("a", N), collapse="")
    pattern <- paste(rep(c("a?", "a"), each=N), collapse="")
  },
  seconds.limit=0.001,
  PCRE.match=regexpr(pattern, subject, perl=TRUE),
  TRE.match=regexpr(pattern, subject, perl=FALSE),
  constant.replacement=gsub("a", "constant size replacement", subject),
  linear.replacement=gsub("a", subject, subject))
plot(string.result)

## Example 2: split data table vs frame, constant factor difference.
library(data.table)
split.result <- atime::atime(
  N=as.integer(10^seq(1, 7)),
  setup={
    set.seed(1)
    DT <- data.table(
      x1 = rep(c("c", "d"), l=N),
      x2 = rep(c("x", "y"), l=N),
      x3 = rep(c("a", "b"), l=N),
      y = rnorm(N)
    )[sample(.N)]
    DF <- as.data.frame(DT)
  },
  seconds.limit=0.001,
  frame=split(DF[names(DF) != "x1"], DF["x1"], drop = TRUE),
  table=split(DT, by = "x1", keep.by = FALSE, drop = TRUE)
)
plot(split.result)
```

atime_grid

Asymptotic timing grid

Description

Create expressions for asymptotic timing by substituting values into expressions.

Usage

```
atime_grid(
  param.list = list(),
  name.value.sep="=",
```

```

expr.param.sep=" ",
collapse = ", ",
...)
```

Arguments

`param.list` Named list of items to replace in ... expressions, default empty list means nothing to replace.

`name.value.sep` string: separator between names and values from `param.list`, default "=".

`expr.param.sep` string: separator between expressions and parameters, default " ".

`collapse` string: separator between parameters, default ", ".

... Named expressions which each must contain each name of `param.list`.

Value

Named list of expressions which can be used as `expr.list` argument of `atime`.

Author(s)

Toby Dylan Hocking

Examples

```

## Example 0: with no param.list, same as quote inside named list.
atime::atime_grid(m=mean(data), s=sum(data))
list(m=quote(mean(data)), s=quote(sum(data)))

## Example 1: polynomial vs exponential time regex.
(expr.list <- atime::atime_grid(
  list(PERL=c(TRUE, FALSE)),
  expr.param.sep="\n",
  regexpr=regexpr(pattern, subject, perl=PERL)))
atime.list <- atime::atime(
  N=unique(as.integer(10^seq(0,3.5,l=20))),
  setup={
    subject <- paste(rep("a", N), collapse="")
    pattern <- paste(rep(c("a?", "a"), each=N), collapse="")
  },
  expr.list=expr.list)
plot(atime.list)
```

`atime_pkg`*Asymptotic timing package tests*

Description

Computation time and memory for several R expressions of several different data sizes, for up to four different package versions (base, HEAD, CRAN, merge-base).

Usage

```
atime_pkg(pkg.path)
```

Arguments

`pkg.path` path to package source directory.

Details

There should be a file named `pkg.path/inst/atime/tests.R` which defines `test.list`, a list with names corresponding to different tests. Each element should be a list with named elements `N`, `setup`, `expr`, to be passed as named arguments to `atime_versions`.

Value

Named list, names come from names of `test.list`, and values come from results of `atime_versions`. Side effect is that `data/plot` files are saved to the `inst/atime` directory.

Author(s)

Toby Dylan Hocking

`atime_versions`*Asymptotic timing of git versions*

Description

Computation time and memory for a single R expression evaluated using several different git versions.

Usage

```
atime_versions(  
  pkg.path, N, setup, expr, sha.vec=NULL,  
  times=10, seconds.limit=0.01, verbose=FALSE,  
  pkg.edit.fun=pkg.edit.default, results=TRUE,  
  ...)
```

Arguments

<code>pkg.path</code>	Path to git repo containing R package.
<code>N</code>	numeric vector of data sizes to vary.
<code>setup</code>	expression to evaluate for every data size, before timings.
<code>expr</code>	code with package double-colon prefix, for example <code>Package::fun(argA, argB)</code> which will be evaluated for each different package version.
<code>sha.vec</code>	named character vector / list of SHA commit IDs.
<code>times</code>	number of times to evaluate each timed expression.
<code>seconds.limit</code>	if the median timing of any expression exceeds this many seconds, then no timings for larger N are computed.
<code>verbose</code>	logical, print messages after every data size?
<code>pkg.edit.fun</code>	function called to edit package before installation, should typically replace instances of <code>PKG</code> with <code>PKG.SHA</code> , default works with Rcpp packages.
<code>results</code>	logical, save results?
<code>...</code>	named SHA/commit IDs to time. Values passed as <code>branch</code> arg to <code>git2r::checkout</code> , names used to identify/interpret this version of the code in the output.

Details

First each version specified by `...` is checked out and installed (to whatever R library is first on `.libPaths()`), using the package name `Package.SHA`. Then the `atime` function is called with arguments defined by the different SHA arguments, `atime(name1=Package.SHA1::fun(argA, argB), name2=Package.SHA2::fun(argA, argB))`.

Value

list of class `atime` with elements `seconds.limit` (numeric input param), `timings` (data table of results).

Author(s)

Toby Dylan Hocking

Examples

```
if(FALSE){
  tdir <- tempfile()
  dir.create(tdir)
  git2r::clone("https://github.com/tdhock/binsegRcpp", tdir)
  atime.list <- atime::atime_versions(
    pkg.path=tdir,
    N=2^seq(2, 20),
    setup={
      max.segs <- as.integer(N/2)
      data.vec <- 1:N
    }
  )
}
```

```

    },
    expr=binsegRcpp::binseg_normal(data.vec, max.segs),
    cv="908b77c411bc7f4fcbcf53759245e738ae724c3e",
    "rm unord map"="dcd0808f52b0b9858352106cc7852e36d7f5b15d",
    "mvl_construct"="5942af606641428315b0e63c7da331c4cd44c091")
  plot(atime.list)

  atime::atime_versions_remove("binsegRcpp")
}

```

atime_versions_exprs *Create expressions for different git versions*

Description

Install different git commit versions as different packages, then create a list of expressions, one for each version. For most use cases `atime_versions` is simpler, but `atime_versions_exprs` is more flexible for the case of comparing different versions of one expression to another expression.

Usage

```

atime_versions_exprs(
  pkg.path, expr, sha.vec=NULL,
  verbose=FALSE,
  pkg.edit.fun=pkg.edit.default, ...)

```

Arguments

<code>pkg.path</code>	Path to git repo containing R package.
<code>expr</code>	code with package double-colon prefix, for example <code>Package::fun(argA, argB)</code> which will be evaluated for each different package version.
<code>sha.vec</code>	named character vector / list of SHA commit IDs.
<code>verbose</code>	logical, print messages after every data size?
<code>pkg.edit.fun</code>	function called to edit package before installation, should typically replace instances of <code>PKG</code> with <code>PKG.SHA</code> , default works with Rcpp packages.
<code>...</code>	named SHA/commit IDs to time. Values passed as <code>branch arg</code> to <code>git2r::checkout</code> , names used to identify/interpret this version of the code in the output.

Details

First each version is checked out and installed (to whatever R library is first on `.libPaths()`), using the package name `Package.SHA`. Then an expression is created for each version, by replacing the `PKG` name in colon-prefix with `PKG.SHA`, `atime(name1=Package.SHA1::fun(argA, argB), name2=Package.SHA2::fun(argA, argB))`. For convenience, versions can be specified either as code `(...)` or data `(sha.vec)`.

Value

list of expressions.

Author(s)

Toby Dylan Hocking

Examples

```
if(FALSE){  
  
  if(requireNamespace("changepoint")){  
    tdir <- tempfile()  
    dir.create(tdir)  
    git2r::clone("https://github.com/tdhock/binsegRcpp", tdir)  
    expr.list <- atime::atime_versions_exprs(  
      pkg.path=tdir,  
      expr=binsegRcpp::binseg_normal(data.vec, max.segs),  
      cv="908b77c411bc7f4fcbcf53759245e738ae724c3e",  
      "rm unord map"="dcd0808f52b0b9858352106cc7852e36d7f5b15d",  
      "mvl_construct"="5942af606641428315b0e63c7da331c4cd44c091")  
    atime.list <- atime::atime(  
      N=2^seq(2, 20),  
      setup={  
        max.segs <- as.integer(N/2)  
        data.vec <- 1:N  
      },  
      expr.list=expr.list,  
      changepoint=changepoint::cpt.mean(  
        data.vec, penalty="Manual", pen.value=0, method="BinSeg",  
        Q=max.segs-1))  
    plot(atime.list)  
  }  
  
  atime::atime_versions_remove("binsegRcpp")  
  
}
```

atime_versions_remove *Remove packages installed by atime*

Description

atime_versions_exprs installs different git versions of a package, and this function removes them.

Usage

```
atime_versions_remove(Package)
```

Arguments

Package Name of package without SHA.

Details

The library searched is the first on `.libPaths()`.

Value

integer exit status code from unlink, non-zero if removal failed.

Author(s)

Toby Dylan Hocking

glob_find_replace *Find and replace within files*

Description

Find and replace for every file specified by glob.

Usage

```
glob_find_replace(glob, FIND, REPLACE)
```

Arguments

glob character string: glob defining files.
FIND character string: regex to find.
REPLACE character string: regex to use for replacement.

Value

nothing.

Author(s)

Toby Dylan Hocking

Examples

```
## see vignette("data.table", package="atime")
```

references_best	<i>Best references</i>
-----------------	------------------------

Description

Compute best asymptotic references.

Usage

```
references_best(L, unit.col.vec=NULL, more.units=NULL, fun.list=NULL)
```

Arguments

L	List output from atime.
unit.col.vec	Named character vector of units, default NULL means standard units (kilobytes and seconds).
more.units	Named character vector of units to add to unit.col.vec, default NULL means nothing.
fun.list	List of asymptotic complexity reference functions, default NULL means to use package default.

Value

list of class "references_best" with elements references (data table of references), measurements (data table of measurements).

Author(s)

Toby Dylan Hocking

Examples

```
## Example 1: polynomial and exponential time string functions.
string.result <- atime::atime(
  N=unique(as.integer(10^seq(0,3.5,l=100))),
  setup={
    subject <- paste(rep("a", N), collapse="")
    pattern <- paste(rep(c("a?", "a"), each=N), collapse="")
  },
  seconds.limit=0.001,
  PCRE.match=regexpr(pattern, subject, perl=TRUE),
  TRE.match=regexpr(pattern, subject, perl=FALSE),
  constant.replacement=gsub("a", "constant size replacement", subject),
  linear.replacement=gsub("a", subject, subject))
(string.best <- atime::references_best(string.result))
## plot method shows each expr in a separate panel.
plot(string.best)
```

```
## Example 2: split data table vs frame, constant factor difference.
library(data.table)
split.result <- atime::atime(
  N=as.integer(10^seq(1, 7)),
  setup={
    set.seed(1)
    DT <- data.table(
      x1 = rep(c("c","d"), l=N),
      x2 = rep(c("x","y"), l=N),
      x3 = rep(c("a","b"), l=N),
      y = rnorm(N)
    )[sample(.N)]
    DF <- as.data.frame(DT)
  },
  seconds.limit=0.001,
  frame=split(DF[names(DF) != "x1"], DF["x1"], drop = TRUE),
  table=split(DT, by = "x1", keep.by = FALSE, drop = TRUE)
)
split.best <- atime::references_best(split.result)
plot(split.best)
```

Index

atime, [2](#), [4](#)
atime_grid, [3](#)
atime_pkg, [5](#)
atime_versions, [5](#)
atime_versions_exprs, [7](#)
atime_versions_remove, [8](#)

glob_find_replace, [9](#)

references_best, [10](#)