# Package ‘atime’

March 8, 2024

**Type**  Package  
**Title**  Asymptotic Timing  
**Version**  2024.3.5  

**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**  2024-03-08 09:10:11 UTC

**R topics documented:**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>atime</td>
<td>2</td>
</tr>
<tr>
<td>atime_grid</td>
<td>3</td>
</tr>
<tr>
<td>atime_pkg</td>
<td>5</td>
</tr>
<tr>
<td>atime_versions</td>
<td>6</td>
</tr>
</tbody>
</table>
Description

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

Usage

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

Arguments

- **N**: numeric vector of at least two data sizes, default is \(2^{\text{seq}(2,20)}\).
- **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
## Example 1: polynomial and exponential time string functions.

```r
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.

```r
library(data.table)
split_result <- atime::atime(
  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

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

```r
expr.param.sep=" ",
collapse = ",",
symbol.params=character()
```

**Arguments**

- `param.list` Named list of items to replace in ... expressions, default empty list means nothing to replace.
- `...` Named expressions which each must contain each name of `param.list`.
- `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 ",,.
- `symbol.params` character vector: these elements of `param.list` will be converted to symbols before substitution.

**Value**

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

**Author(s)**

Toby Dylan Hocking

**Examples**

```r
## 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("a?", "a"), each=N), collapse="")
  },
  expr.list=expr.list)
plot(atime.list)

## Example 2: symbol.params arg.
sub.param.list <- list(FUN=c("sub","gsub"), PERL=c(TRUE,FALSE))
## with base R we can use as.symbol and substitute:
sapply(sub.param.list$FUN,function(name)substitute(fun("a","",subject), list(fun=as.symbol(name))))
## with atime_grid the analog is to use symbol.params argument:
(sub.expr.list <- atime::atime_grid(
  list(FUN=c("sub","gsub"), PERL=c(TRUE,FALSE)),
  symbol.params=c("sub","gsub"),
  expr.param.sep=" ",
  collapse="\n",)
```
sub.param.list, replace=FUN("a","",subject,perl=PERL), symbol.params="FUN")
sub.atime.list <- atime::atime(
  setup=(
    subject <- paste(rep("a",N),collapse="")
  ),
  expr.list=sub.expr.list)
plot(sub.atime.list)

---

atime_pkg

Asymptotic timing package tests

Description

Computation time and memory for several R expressions of several different data sizes, for several package versions (base, HEAD, CRAN, merge-base, others specified by user).

Usage

atime_pkg(pkg.path)

Arguments

pkg.path    path to git repository containing R package.

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 at least three named elements: N, setup, expr, (possibly others such as pkg.edit.fun and version_name="1234567890abcdef") to be passed as named arguments to atime_versions, along with the following versions which are passed using the sha.vec argument: base ref comes from GITHUB_BASE_REF environment variable (default master), HEAD ref is the branch that you want to merge into base, CRAN is current published version (sha value ""), merge-base is most recent common ancestor commit between base and HEAD. The tests.R file can define version.colors which should be a character vector (names for versions, values for colors).

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
Examples

```r
if(FALSE){
  tdir <- tempfile()
  dir.create(tdir)
  git2r::clone("https://github.com/tdhock/binsegRcpp", tdir)
  repo <- git2r::repository(tdir)
  git2r::checkout(repo, "another-branch")
  result.list <- atime::atime_pkg(tdir)
  inst.atime <- file.path(tdir, "inst", "atime")
  dir(inst.atime)
  tests.RData <- file.path(inst.atime, "tests.RData")
  (objs <- load(tests.RData))

  atime::atime_versions_remove("binsegRcpp")
}
```

### atime_versions

**Asymptotic timing of git versions**

#### Description

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

#### Usage

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

#### Arguments

- **pkg.path**
  - Path to git repo containing R package.
- **N**
  - numeric vector of data sizes to vary.
- **setup**
  - expression to evaluate for every data size, before timings. In contrast to `expr`, no replacement of `Package:` is performed.
- **expr**
  - code with package double-colon prefix, for example `PKG::fun(argA, argB)`, where `PKG` is the name of the package specified by `pkg.path`. This code will be evaluated for each different package version, by replacing `PKG: ` by `PKG.SHA: `. To run different versions of implicitly-called functions like `DT[i,j]`, you need to call them explicitly, as in `data.table:::\data.table(DT, i, j)`.
atime_versions

- `sha.vec` named character vector / list of versions.
- `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?
- `pkg.edit.fun` function called to edit package before installation, should typically replace instances of PKG with PKG.SHA, default works with Rcpp packages.
- `result` logical, save results? (default FALSE)
- ... named versions.

Details

For convenience, versions can be specified either as code (...), data (sha.vec), or both. Each version should be either "" (to use currently installed version of package, or if missing, install most recent version from CRAN) or a SHA1 hash, which is passed as branch arg to git2r::checkout; version names used to identify/interpret the output/plots.

Value

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

Author(s)

Toby Dylan Hocking

See Also

atime_versions_exprs converts expr into a list of expressions, one for each version, passed to atime as the expr.list argument.

Examples

```r
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="988b77c411bc7f4fcbcf53759245e738ae724c3e",
    "rm unord map"="dcd0808f52b0b9858352106cc7852e36d7f5b15d",
  )
}
```
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

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

Arguments

- `pkg.path` Path to git repo containing R package.
- `expr` code with package double-colon prefix, for example `PKG::fun(argA, argB)`, where `PKG` is the name of the package specified by `pkg.path`. This code will be evaluated for each different package version, by replacing `PKG:` by `PKG.SHA:`. To run different versions of implicitly-called functions like `DT[i,j]`, you need to call them explicitly, as in `data.table:::`.`data.table`(`DT`, `i`, `j`).
- `sha.vec` named character vector / list of versions.
- `verbose` logical, print messages after every data size?
- `pkg.edit.fun` function called to edit package before installation, should typically replace instances of `PKG` with `PKG.SHA`, default works with Rcpp packages, but does not work with all packages. For an example of a custom package editing function, see the atime vignette about data.table.
- `...` named versions.

Details

For convenience, versions can be specified either as code (``...``), data (`sha.vec`), or both. Each version should be either "" (to install most recent version from CRAN) or a SHA1 hash, which is passed as branch arg to `git2r::checkout`; version names used to identify/interpret the output/plots. Each version is installed as a separate package (to whatever R library is first on `.libPaths()`), using the package name `PKG.SHA`.
Value

A list of expressions, one for each version, created by replacing PKG: in expr with PKG.SHA::

atime(name1=Package.SHA1::fun(argA, argB), name2=Package.SHA2::fun(argA,argB)).

Author(s)

Toby Dylan Hocking

Examples

```r
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="908b77c411bc7f4fbc5f3759245e738ae724c3e",
      "rm unord map"="dcd0d808f52b0b9858352106cc7852e36d7f5b15d",
      "mvl_construct"="5942af6064142825b0e63c7da331c4d44c091")
    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")
}
```

Description

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

Compute best asymptotic references.

Usage

```r
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

```r
## 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)
```
## predict method computes N for given units (default seconds limit).

```r
string.pred = predict(string.best)
plot(string.pred)
```
Index

atime, 2, 4, 7
atime_grid, 3
atime_pkg, 5
atime_versions, 6
atime_versions_exprs, 7, 8
atime_versions_remove, 9

glob_find_replace, 10

references_best, 11