Package 'SharedObject'

April 14, 2020

Type Package

Title Sharing R objects across multiple R processes without memory duplication

Version 1.0.0

Date 2019-6-10

Author Jiefei Wang

Maintainer Jiefei Wang <szwjf08@gmail.com>

Description This package is developed for facilitating parallel computing in R.
It is capable to create an R object in the shared memory space and share the data across multiple R processes.
It avoids the overhead of memory dulplication and data transfer, which make sharing big data object across many clusters possible.

License GPL-3

LinkingTo BH, Rcpp

Depends R (>= 3.6.0)

Imports Rcpp, methods, stats, xptr, BiocGenerics

biocViews Infrastructure

BugReports https://github.com/Jiefei-Wang/SharedObject/issues

Suggests testthat, parallel, knitr, rmarkdown, BiocStyle

RoxygenNote 6.1.1

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Encoding UTF-8

git_url https://git.bioconductor.org/packages/SharedObject

git_branch RELEASE_3_10

git_last_commit 5a594e2

git_last_commit_date 2019-10-29

Date/Publication 2020-04-13

R topics documented:

`.removeObject` *Internal functions*

Description

Internal functions

Usage

```
.removeObject(dataId)

.getProperty(x, property)

.setProperty(x, property, value)
```

Arguments

`dataId`	A vector of numeric numbers. The keys of shared objects
`x`	A shared object
`property`	Character, the name of the property
`value`	The new value of the property

Details

.removeObject: This function will delete the data associated with the key provided by the argument `dataId`. Any try to read the data after the function call might crash R. If `dataId` is set to "all", the function will delete all data. This function is for the case where R is terminated abnormally without clearing the shared memory. In normal case, R's garbage collector will free the shared memory and there is no need to call this function.

.getProperty: This function returns a proporty of a shared object. The available properties are `dataId`,`typeName`,`ownData`,`processId`, `typeId`,`length`,`totalSize`,`copyOnWrite`,`sharedSubset`,`sharedCopy`, `setCopyOnWrite`,`setSharedSubset` and `setSharedCopy`.

`.setProperty`: Set a property of a shared object.

Value

.removeObject: no return value

.getProperty: The property of a shared vector or a list of properties of a data.frame

.setProperty: No return value

Examples

```
## Create a shared object
A = share(1:10)
## Get the ID of the shared object
id = .getProperty(A, "dataId")
## Delete the data associated with the ID
.removeObject(id)
## Accessing the variable will cause an error since its
## data has been deleted.
## Not run:
A

## End(Not run)
```

`getCopyOnWrite` *Get or set the properties of a shared object*

Description

Get or set the properties of a shared object

Usage

```
getCopyOnWrite(x)

getSharedSubset(x)

getSharedCopy(x)

setCopyOnWrite(x, value)

setSharedSubset(x, value)

setSharedCopy(x, value)
```

Arguments

`x`	A shared object
`value`	logical, the value you want the property to be. For a `data.frame`, it can be either a single value or a vector. If the length of `value` does not match the column of the `data.frame`, `value` will be replicate to match the length.

Value

`get`: The property of a shared object

`set`: No return value

Examples

```
x = share(1:20)
## Check the default values
getCopyOnWrite(x)
getSharedSubset(x)
getSharedCopy(x)

## Set the values
setCopyOnWrite(x,FALSE)
setSharedSubset(x,FALSE)
setSharedCopy(x,TRUE)

## Check the values again
getCopyOnWrite(x)
getSharedSubset(x)
getSharedCopy(x)
```

---

getDataInfo *Get a summary report of the data in the shared memory*

Description

This function will return a list of the summary of the data in the shared memory.

Usage

```
getDataInfo(data_ids = NULL)
```

Arguments

data_ids	The data ID. If NULL(default), all data will be returned.

Value

A data.frame

Examples

```
getDataInfo()
```

---

getSharedProperties *get a summary report for a shared object*

Description

The function reports the properties related to the shared object. the individual property can be accessed via $ operator.

Usage

```
getSharedProperties(x, as.list = FALSE)
```

Arguments

x	A shared object
as.list	Logical, Whether the result should be a list of shared properties even when only one sharedProperty object is reported.

Value

If x is not a shared object, return NULL.

If x is a shared vector and as.list is FALSE, the function returns a sharedProperty object.

Otherwise, the function returns a list of sharedProperty object(s).

See Also

getCopyOnWrite, getSharedSubset, getSharedCopy

Examples

```
x = share(1:10)
getSharedProperties(x)
```

---

is.altrep *Is an object of a desired type?*

Description

Is an object of a desired type?

Usage

```
is.altrep(x)

is.shared(x, recursive = TRUE)
```

Arguments

x	an R object
recursive	Logical, whether a data.frame can be treated as a shared object. If TRUE, a data.frame is called a shared object if and only if all of its columns are shared objects.

Value

A logical value

Examples

```
x = share(1:10)
is.altrep(x)
is.shared(x)
```

setSharedObjectOptions *Get or set the global options for the SharedObject package*

Description

Get or set the global options for the SharedObject package

Usage

```
setSharedObjectOptions(...)

getSharedObjectOptions(...)
```

Arguments

...	setSharedObjectOptions: the options you want to set, it can be copyOnWrite, sharedSubset and sharedCopy. getSharedObjectOptions: A character vector. If empty, all options will be returned.

Value

setSharedObjectOptions: No return value getSharedObjectOptions: A list of the package options

Examples

```
getSharedObjectOptions()
setSharedObjectOptions(copyOnWrite = FALSE)
getSharedObjectOptions()
getSharedObjectOptions("copyOnWrite")
```

share *Create an R object in the shared memory*

Description

This function will create an object in the shared memory for the function argument x and return a shared object. There is no duplication of the shared object when it is exported to the other processes. All the shared objects will use the data located in the same shared memory space.

Usage

```
share(x, ...)
```

Arguments

x	An R object that you want to shared. The supported data types are raw, logical, integer and real. The data structure can be vector, matrix and data.frame. List is not supported but can be created manually.
...	Additional parameters that will be passed to the shared object, see details.

Details

When the function argument x is an atomic object(e.g vector, matrix), the function will create an ALTREP object to replace it. When x is a data frame, each column of x will be replaced by an ALTREP object.

In the R level, the behaviors of an ALTREP object is exactly the same as an atomic object but the data of an ALTREP object is allocated in the shared memory space. Therefore an ALTREP object can be easily exported to the other R processes without dulplicating the data, which reduces the memory usage and the overhead of data transmission.

The behavior of a shared object can be controlled through three parameters: copyOnWrite, sharedSubset and sharedCopy.

copyOnWrite determines Whether a new R object need to be allocated when the shared object is changed. The default value is TRUE, but can be altered by passing an argument copyOnWrite=FALSE to the function.

Please note that the no-copy-on-write feature is not fully supported by R. When copyOnWrite is FALSE, a shared object might not behaves as user expects. Please refer to the example code to see the exceptions.

sharedSubset`` determines whether the subset of a shared object is still a shared object. The default value is TRUE,and can be changed by passing sharedSubset=FALSE‘ to the function

At the time this documentation is being written, The shared subset feature will cause an unnecessary memory duplication in R studio. Therefore, for the performance consideration, it is better to turn the feature off when using R studio.

sharedCopy determines whether the object is still a shared object after a duplication. If copyOnWrite is FALSE, this feature is off since the duplication cannot be triggered. In current version (R 3.6), an object will be duplicated four times for creating a shared object and lead to a serious performance problem. Therefore, the default value is FALSE, user can alter it by passing sharedCopy=FALSE to the function. alter

Value

A shared object

Examples

```
## For vector
x = runif(10)
so = share(x)
x
so

## For matrix
x = matrix(runif(10), 2, 5)
so = share(x)
x
```

```
so

## For data frame
x = as.data.frame(matrix(runif(10), 2, 5))
so = share(x)
x
so

## export the object
library(parallel)
cl = makeCluster(1)
clusterExport(cl, "so")
## check the exported object in the other process
clusterEvalQ(cl, so)

## close the connection
stopCluster(cl)

## Copy-on-write
## This is the default setting
x = runif(10)
so1 = share(x, copyOnWrite = TRUE)
so2 = so1
so2[1] = 10
## so1 is unchanged since copy-on-write feature is on.
so1
so2

## No-copy-on-write
so1 = share(x, copyOnWrite = FALSE)
so2 = so1
so2[1] = 10
#so1 is changed
so1
so2

## Flaw of no-copy-on-write
## The following code changes the value of so1, highly unexpected! Please use with caution!
-so1
so1
## The reason is that the minus function trys to dulplicate so1 object,
## but the dulplicate function will return so1 itself, so the value in so1 also get changed.
```

Index