Package 'broadcast'

January 11, 2025

Title Simple Broadcasted Binary Operations for Atomic and Recursive Arrays with Minimal Dependencies

Version 0.0.0.9000

Description Implements simple broadcasted binary operations,
for atomic and recursive arrays.
All operations are element-wise binary operations
(i.e. involving only 2 arrays at a time),
to prevent the user from accidentally exploding dimension sizes of the resulting array.
Besides linking to 'Rcpp',
'broadcast' does not rely on any external libraries,
thus minimizing the number of dependencies.
'broadcast' supports the following operators.
1) Relational operators (`==`, `!=`, `<`, `>`, `<=`, `>=`);
2) Boolean combiner operators (`&`, `|`, `xor()`, ``not-and");
3) Arithmetic operators (`+`, `-`, `*`, `/`, `^` ``intmod", `pmin()`, `pmax()`);
4) String concatination operators (`+` , `*`);
6) Broadcasted 'lapply' for recursive arrays;
7) Broadcasted insertion.
The broadcasted operations are fast and memory efficient
(which is not just good for computer efficiency, but also for the environment).

License MPL-2.0 | file LICENSE

Encoding UTF-8

LinkingTo Rcpp

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Depends R (>= 4.2.0)

Imports Rcpp (>= 1.0.11)

Suggests tinytest

Contents

`aaa00_broadcast_help` *broadcast: Subset Methods as Alternatives to the Square Brackets Operators for Programming*

Description

broadcast:
Simple broadcasted binary operations for atomic and recursive arrays in 'R'.

Details

The 'broadcast' package provides a set of type-specific functions for broadcasted operations. The functions use an API similar to the outer and sweep functions.

The following functions are available:

- bc.num: Broadcasted operations for numeric (types `int` and `dbl`) arrays.
- `bc.bool`: Broadcasted operations for logical/Boolean arrays.
- `bc.cplx`: Broadcasted operations for `complex` arrays.
- `bc.str`: Broadcasted operations for `character` arrays.
- `bc.list`: Broadcasted operations for recursive arrays.

Each of these functions support 2 types of operations:

- `regular` operations, which return an array of the same type.
  For example: +, -, *, /, etc.
- `relational` operations, which return a logical array.
  For example: ==, !=, etc.

The 'broadcast' package supports relational operators (==, !=, <, > <=, >=), logical combiners (&, |, xor, nand), arithmetic (+, -, *, /, ^)

Author(s)

Author, Maintainer: Tony Wilkes <tony_a_wilkes@outlook.com> (ORCID)

References

The badges shown in the documentation of this R-package were made using the services of: https://shields.io/

array_recycle *Recycle Array Dimensions*

Description

The `array_recycle()` function recycles array dimensions until the specified dimension sizes are reached, and returns the array.

The various broadcasting functions "recycle" an array virtually, meaning little to no additional memory is needed.
The `array_recycle()` function, however, physically recycles an array (and thus actually occupies memory space).

Usage

```
array_recycle(x, tdim)
```

Arguments

`x`	an atomic or recursive array or matrix.
`tdim`	an integer vector, giving the target dimension to reach.

Value

Returns the recycled array.

Examples

```
x <- matrix(1:9, 3,3)
colnames(x) <- LETTERS[1:3]
rownames(x) <- letters[1:3]
names(x) <- month.abb[1:9]
print(x)

array_recycle(x, c(3,3,2)) # recycle to larger size
```

atomic_typecast *Atomic Type Casting With Names and Dimensions Preserved*

Description

Atomic type casting in R is generally performed using the functions as.logical, as.integer, as.double, as.character, as.complex, and as.raw.

Converting an object between atomic types using these functions strips the object of its attributes, including (dim)names and dimensions.

The functions provided here by the 'tinycodet' package preserve the dimensions, dimnames, and

names.

The functions are as follows:

- `as_bool()`: converts object to atomic type `logical` (`TRUE`, `FALSE`, `NA`).
- `as_int()`: converts object to atomic type `integer`.
- `as_dbl()`: converts object to atomic type `double` (AKA decimal numbers).
- `as_chr()`: converts object to atomic type `character`.
- `as_cplx()`: converts object to atomic type `complex`.
- `as_raw()`:converts object to atomic type `raw`.

Usage

```
as_bool(x, ...)

as_int(x, ...)

as_dbl(x, ...)

as_chr(x, ...)

as_cplx(x, ...)

as_raw(x, ...)
```

Arguments

x	vector, matrix, array (or a similar object where all elements share the same type).
...	further arguments passed to or from other methods.

Value

The converted object.

Examples

```
# matrix example ====
x <- matrix(sample(-1:28), ncol = 5)
colnames(x) <- month.name[1:5]
rownames(x) <- month.abb[1:6]
names(x) <- c(letters[1:20], LETTERS[1:10])
print(x)

as_bool(x)
as_int(x)
as_dbl(x)
as_chr(x)
as_cplx(x)
as_raw(x)
```

```
################################################################################

# factor example ====
x <- factor(month.abb, levels = month.abb)
names(x) <- month.name
print(x)

as_bool(as_int(x) > 6)
as_int(x)
as_dbl(x)
as_chr(x)
as_cplx(x)
as_raw(x)
```

---

bc.b *Broadcasted Operations for Logical Arrays*

Description

The bc.b() function performs broadcasted operations on 2 logical arrays.

Usage

```
bc.b(x, y, op)
```

Arguments

x, y	conformable atomic arrays of types logical, integer, or double.
op	a single string, giving the operator. Supported Boolean combiner operators: &, |, xor, nand. Supported relational operators: ==, !=, <, >, <=, >=.

Value

For the boolean combiner operators:
A logical array as a result of the broadcasted arithmetic operation.

For relational operators:
A logical array as a result of the broadcasted relational comparison.

Examples

```
x.dim <- c(10:8)
x.len <- prod(x.dim)
x.data <- sample(c(TRUE, FALSE, NA), x.len, TRUE)
x <- array(x.data, x.dim)
y <- array(1:50, c(10,1,1))
```

```
bc.b(x, y, "&")
bc.b(x, y, "|")
bc.b(x, y, "xor")
bc.b(x, y, "nand")

bc.b(x, y, "==")
bc.b(x, y, "!=")
bc.b(x, y, "<")
bc.b(x, y, ">")
bc.b(x, y, "<=")
bc.b(x, y, ">=")
```

`bc.num` *Broadcasted Operations for Numeric Arrays*

Description

The `bc.num()` function performs broadcasted operations on 2 numeric arrays.

Usage

```
bc.num(x, y, op, prec = sqrt(.Machine$double.eps))
```

Arguments

`x`, `y`	conformable atomic arrays of types `logical`, `integer`, or `double`.
`op`	a single string, giving the operator. Supported arithmetic operators: +, -, *, /, ^, intmod, pmin, pmax. Supported relational operators: ==, !=, <, >, <=, >=, d==, d!=, d<, d>, d<=, d>=.
`prec`	a single number between 0 and 0.1, giving the machine precision to use. Only relevant for the following operators: d==, d!=, d<, d>, d<=, d>= See the d==, d!=, d<, d>, d<=, d>= operators from the 'tinycodet' package for details.

Value

For arithmetic operators:
A numeric array as a result of the broadcasted arithmetic operation.

For relational operators:
A logical array as a result of the broadcasted relational comparison.

Examples

```
x.dim <- c(10:8)
x.len <- prod(x.dim)
x.data <- sample(c(NA, 1.1:1000.1), x.len, TRUE)
x <- array(x.data, x.dim)
y <- array(1:50, c(10,1,1))

bc.num(x, y, "+")
bc.num(x, y, "-")
bc.num(x, y, "*")
bc.num(x, y, "/")
bc.num(x, y, "^")

bc.num(x, y, "==")
bc.num(x, y, "!=")
bc.num(x, y, "<")
bc.num(x, y, ">")
bc.num(x, y, "<=")
bc.num(x, y, ">=")
```

`bc_pred_dim` *Predict Broadcasted dimensions*

Description

`bc_pred_dim()`

Usage

```
bc_pred_dim(x, y)
```

Arguments

x, y	an atomic array or matrix.

Value

Returns the recycled array.

Examples

```
x <- matrix(1:9, 3,3)
colnames(x) <- LETTERS[1:3]
rownames(x) <- letters[1:3]
names(x) <- month.abb[1:9]
print(x)

array_recycle(x, c(3,3,2)) # recycle to larger size
```

Index