

Package ‘fingerprint’

November 1, 2009

Version 3.2

Date 2009-10-31

Title Functions to operate on binary fingerprint data

Author Rajarshi Guha <rajarshi.guha@gmail.com>

Maintainer Rajarshi Guha <rajarshi.guha@gmail.com>

Description This package contains functions to manipulate binary fingerprints of arbitrary length. A fingerprint is represented by an object of S4 class ‘fingerprint’ which is internally represented a vector of integers, such that each element represents the position in the fingerprint that is set to 1. The bitwise logical functions in R are overridden so that they can be used directly with ‘fingerprint’ objects. A number of distance metrics are also available (many contributed by Michael Faddock). Fingerprints can be converted to Euclidean vectors (i.e., points on the unit hypersphere) and can also be folded using OR. Arbitrary fingerprint formats can be handled via line handlers. Currently handlers are provided for CDK, MOE and BCI fingerprint data.

License GPL

Depends methods

LazyLoad yes

Repository CRAN

Date/Publication 2009-11-01 17:33:48

R topics documented:

as.character	2
bit.spectrum	3
cdk.lf, moe.lf, bci.lf	4
distance	4
euc.vector	7
fingerprint-class	8
fold	9
fp.factor.matrix	10

fp.read, fp.read.to.matrix	11
fp.sim.matrix	12
fp.to.matrix	13
fp.logical	14
length	14
random.fingerprint	15
show	16

Index	17
--------------	-----------

as.character	<i>Generates a String Representation of a Fingerprint</i>
--------------	---

Description

The function returns a string of 1's and 0's corresponding to the fingerprint object supplied

Usage

```
## S4 method for signature 'fingerprint':
as.character(x)
```

Arguments

x An object of class fingerprint

Value

A string of 1's and 0's

Author(s)

Rajarshi Guha (rguha@indiana.edu)

Examples

```
# make a fingerprint vector
fp <- new("fingerprint", nbit=32, bits=sample(1:32, 20))

# print out the string representation
as.character(fp)
```

Description

The idea of comparing datasets using fingerprints was described in Guha & Schurer (2008). The idea is that one can summarize the dataset by counting the frequency of occurrence of each bit position. The frequency is normalized by the number of fingerprints considered. Thus a collection of N fingerprints can be converted to a single vector of numbers highlighting the most frequent bits with respect to a given dataset. A plot of this vector looks like a traditional spectrum and hence the name.

The bit spectra for two datasets (assuming that the same types of fingerprints have been used) allows one to compare the similarity of the datasets, without having to do a full pairwise similarity calculation. The difference between the structural features of the datasets can be quantified by evaluating the distance between the two bit spectra.

Usage

```
bit.spectrum(fplist)
```

Arguments

`fplist` A list structure with each element being an object of class `fingerprint`. These will can be constructed by hand or read from disk via [fp.read](#). All fingerprints in the list should be of the same length.

Value

A numeric vector of length equal to the size of the fingerprints.

Author(s)

Rajarshi Guha (rguha@indiana.edu)

References

Guha, R.; Schurer, S.; *J. Comp. Aid. Molec. Des.*, **2008**, *22*, 367-384.

See Also

[distance](#), [fp.read](#)

```
cdk.lf, moe.lf, bci.lf
```

Functions to parse lines from fingerprint files

Description

These functions take a single line and parse it to produce a vector of integers which represents the position of the 'on' bits in a fingerprint. This allows the user to use `read.fp` with arbitrary fingerprint files. A new file format can be handled by defining a new line parser function. Currently the three functions process fingerprint files obtained from the CDK (<http://cdk.sourceforge.net>), MOE (<http://chemcomp.com>) and BCI (<http://www.digitalchemistry.co.uk/>)

Usage

```
cdk.lf(line)
moe.lf(line)
bci.lf(line)
```

Arguments

`line` The line to parse

Value

A vector of integers representing 'on' bits

Author(s)

Rajarshi Guha <rguha@indiana.edu>

```
distance
```

Calculates the Similarity or Dissimilarity Between Two Fingerprints

Description

A number of distance metrics can be calculated for binary fingerprints. Some of these are actually similarity metrics and thus represent the reverse of a distance metric.

The following are distance (dissimilarity) metrics

- Hamming
- Mean Hamming
- Soergel
- Pattern Difference

- Variance
- Size
- Shape

The following metrics are similarity metrics and so the distance can be obtained by subtracting the value from 1.0

- Tanimoto
- Dice
- Modified Tanimoto
- Simple
- Jaccard
- Russel-Rao
- Rodgers Tanimoto
- Cosine
- Achiai
- Carbo
- Baroniurbanibuser
- Kulczynski2

Finally the method also provides a set of composite and asymmetric distance metrics

- Hamann
- Yule
- Pearson
- Dispersion
- McConnaughey
- Stiles
- Simpson
- Petke

The default metric is the Tanimoto coefficient.

Usage

```
distance(fp1, fp2, method)
```

Arguments

fp1	An object of class fingerprint
fp2	An object of class fingerprint
method	The type of distance metric desired. Partial matching is supported and the default is <code>tanimoto</code> . Alternative values are <ul style="list-style-type: none">• <code>euclidean</code>

- hamming
- meanHamming
- soergel
- patternDifference
- variance
- size
- shape
- jaccard
- dice
- mt
- simple
- russelrao
- rodgerstanimoto
- cosine
- achiai
- carbo
- baroniurbanibuser
- kulczynski2
- hamann
- yule
- pearson
- mcconnaughey
- stiles
- simpson
- petke

Value

Numeric value representing the distance in the specified metric between the supplied fingerprint objects

Author(s)

Rajarshi Guha (rguha@indiana.edu)

References

Fligner, M.A.; Verducci, J.S.; Blower, P.E.; A Modification of the Jaccard-Tanimoto Similarity Index for Diverse Selection of Chemical Compounds Using Binary Strings, *Technometrics*, 2002, 44(2), 110-119

Monve, V.; Introduction to Similarity Searching in Chemistry, *MATCH - Comm. Math. Comp. Chem.*, 2004, 51, 7-38

Examples

```
# make a 2 fingerprint vectors
fp1 <- new("fingerprint", nbit=6, bits=c(1,2,5,6))
fp2 <- new("fingerprint", nbit=6, bits=c(1,2,5,6))

# calculate the tanimoto coefficient
distance(fp1,fp2) # should be 1

# Invert the second fingerprint
fp3 <- !fp2

distance(fp1,fp3) # should be 0
```

`euc.vector`*Euclidean Representation of Binary Fingerprints*

Description

Ordinarily, a binary fingerprint can be considered to represent a corner of a nD hypercube. However in many cases using such a representation can lead to a very sparse space. Consequently one approach is to convert the fingerprint so that it represents points on a nD unit hypersphere.

The resultant fingerprint is then a nD coordinate.

Usage

```
euc.vector(fp)
```

Arguments

`fp` An object of class fingerprint.

Value

A numeric of length equal to the bit length of the fingerprint. The result corresponds to a unit vector for a point on the nD hypersphere

Author(s)

Rajarshi Guha (rguha@indiana.edu)

Examples

```
# make a fingerprint vector
fp <- new("fingerprint", nbit=8, bits=c(1,3,4,5,7))
vec <- euc.vector(fp)
```

fingerprint-class *Class "fingerprint"*

Description

This class represents binary fingerprints, usually generated by a variety of cheminformatics software, but not restricted to such

Objects from the Class

Objects can be created by calls of the form `new("fingerprint", ...)`. Fingerprints can traditionally thought of as a vector of 1's and 0's. However for large fingerprints this is inefficient and instead we simply store the positions of the bits that are on. Certain operations also need to know the length of the original bit string and this length is stored in the object at construction. Even though we store extra information along with the bit positions, conceptually we still consider the objects as simple bit strings. Thus the usual bitwise logical operations (&, |, !, xor) can be applied to objects of this class.

Slots

bits: Object of class "numeric" ~~ A vector indicating the bit positions that are on.

nbit: Object of class "numeric" ~~ Indicates the length of the original bit string.

folded: Object of class "logical" ~~ Indicates whether the fingerprint has been folded.

provider: Object of class "character" ~~ Indicates the source of the fingerprint. Can be useful to keep track of what software generated the fingerprint.

name: Object of class "character" ~~ The name associated with the fingerprint. If not name is available this gets set to an empty string

Methods

distance signature(fp1 = "fingerprint", fp2 = "fingerprint", method = "missing"):...

distance signature(fp1 = "fingerprint", fp2 = "fingerprint", method = "character"):...

euc.vector signature(fp = "fingerprint"):...

fold signature(fp = "fingerprint"):...

random.fingerprint signature(nbit = "numeric", on = "numeric"):...

Author(s)

Rajarshi Guha (rajarshi.guha@gmail.com)

See Also

[fp.read](#), [fp.read.to.matrix](#) [fp.sim.matrix](#), [fp.to.matrix](#), [fp.factor.matrix](#)
[random.fingerprint](#)

Examples

```
## make fingerprints
x <- new("fingerprint", nbit=128, bits=sample(1:128, 100))
y <- x
distance(x,y) # should be 1
x <- new("fingerprint", nbit=128, bits=sample(1:128, 100))
distance(x,y)
folded <- fold(x)

## binary operations on fingerprints
x <- new("fingerprint", nbit=8, bits=c(1,2,3,6,8))
y <- new("fingerprint", nbit=8, bits=c(1,2,4,5,7,8))
x & y
x | y
!x
```

fold	<i>Fold a fingerprint</i>
------	---------------------------

Description

In many situations a fingerprint is generated using a large length (such as 1024 bits or more). As a result of this, the fingerprints for a dataset can be very sparse. One approach to increasing bit density of such fingerprints is to fold them. This is performed by dividing the original fingerprint bitstring into two substrings of equal length and then perform an OR on the two substrings.

It should be noted that many fingerprint generating routines will perform this internally.

Usage

```
fold(fp)
```

Arguments

fp The fingerprint to fold. Should be of class fingerprint.

Value

An object of class `fingerprint` representing the folded fingerprint.

Author(s)

Rajarshi Guha (rguha@indiana.edu)

Examples

```
# make a fingerprint vector
fp <- new("fingerprint", nbit=64, bits=sample(1:64, 30))
fold(fp)
```

fp.factor.matrix *Converts a List of Fingerprints to a data.frame of Factors*

Description

This function will convert a `list` of fingerprint objects to a `data.frame` of factors with levels 1 and 0.

Usage

```
fp.factor.matrix(fplist)
```

Arguments

`fplist` A list structure with each element being an object of class `fingerprint`. These will can be constructed by hand or read from disk via [fp.read](#)

Value

A matrix with dimensions equal to `(length(fplist), length(fplist))`

Author(s)

Rajarshi Guha (rguha@indiana.edu)

See Also

[distance](#), [fp.read](#)

Examples

```
# make fingerprint objects
fp1 <- new("fingerprint", nbit=6, bits=c(1,2,5,6))
fp2 <- new("fingerprint", nbit=6, bits=c(1,4,5,6))
fp3 <- new("fingerprint", nbit=6, bits=c(2,3,4,5,6))

fp.factor.matrix( list(fp1,fp2,fp3) )
```

`fp.read, fp.read.to.matrix`*Functions to Read Fingerprints From Files*

Description

`fp.read` reads in a set of fingerprints from a file. Fingerprint output from the CDK, MOE and BCI can be handled.

Each fingerprint is represented as a `fingerprint` object. `fp.read` returns a `list` structure, each element being a `fingerprint` object.

`fp.read.to.matrix` is a utility function that reads the fingerprints directly to matrix form (columns are the bit positions and the rows are the objects whose fingerprints have been evaluated)

Usage

```
fp.read(f='fingerprint.txt', size=1024, lf=cdk.lf, header=FALSE)
fp.read.to.matrix(f='fingerprint.txt', size=1024, lf=cdk.lf, header=FALSE)
```

Arguments

<code>f</code>	File containing the fingerprints
<code>size</code>	The bit length of the fingerprints being considered
<code>lf</code>	A line reading function that parses a single line from a fingerprint file. Currently, three such functions are provided that parse the fingerprints from the output of the CDK, MOE and the BCI toolkit, respectively.
<code>header</code>	Indicates whether the first line of the fingerprint file is a header line

Value

A `list` or `matrix` of fingerprints

Author(s)

Rajarshi Guha (rguha@indiana.edu)

See Also

[cdk.lf](#), [moe.lf](#), [bci.lf](#)

`fp.sim.matrix`*Calculates a Similarity Matrix for a Set of Fingerprints*

Description

Given a set of fingerprints, a pairwise similarity can be calculated using the various distance metrics defined for binary strings. This function calculates the pairwise similarity matrix for a set of fingerprint objects supplied in a `list` structure. Any of the distance metrics provided by [distance](#) can be used and the default is the Tanimoto metric.

Note that if the the Euclidean distance is specified then the resultant matrix is a distance matrix and not a similarity matrix

Usage

```
fp.sim.matrix(fplist, method='tanimoto')
```

Arguments

<code>fplist</code>	A list structure with each element being an object of class <code>fingerprint</code> . These will can be constructed by hand or read from disk via fp.read
<code>method</code>	The type of distance metric to use. Alternatives are <code>euclidean</code> and <code>dice</code> and <code>mt</code> . The default is <code>tanimoto</code> . Partial matching is supported.

Value

A matrix with dimensions equal to `(length(fplist), length(fplist))`

Author(s)

Rajarshi Guha (rguha@indiana.edu)

See Also

[distance](#), [fp.read](#)

Examples

```
# make fingerprint objects
fp1 <- new("fingerprint", nbit=6, bits=c(1,2,5,6))
fp2 <- new("fingerprint", nbit=6, bits=c(1,4,5,6))
fp3 <- new("fingerprint", nbit=6, bits=c(2,3,4,5,6))

fp.sim.matrix( list(fp1,fp2,fp3) )
```

fp.to.matrix	<i>Converts a List of Fingerprints to a Matrix</i>
--------------	--

Description

In general, fingerprint data is read from a file or obtained via calls to an external generator and the return value is a list of fingerprints. This function takes the list and returns a matrix having number of rows equal to the number of fingerprints and the number of columns equal to the length of the fingerprint. Each element is 1 or 0 (1's being specified by the positions in each fingerprint vector)

Usage

```
fp.to.matrix(fplist)
```

Arguments

`fplist` A list structure with each element being an object of class `fingerprint`. These will can be constructed by hand or read from disk via [fp.read](#)

Value

A matrix with dimensions equal to `length(fplist)`, `bit length`) where `bit length` is a property of the fingerprint objects in the list.

Author(s)

Rajarshi Guha <rguha@indiana.edu>

See Also

[distance](#), [fp.read](#)

Examples

```
# make fingerprint objects
fp1 <- new("fingerprint", nbit=6, bits=c(1,2,5,6))
fp2 <- new("fingerprint", nbit=6, bits=c(1,4,5,6))
fp3 <- new("fingerprint", nbit=6, bits=c(2,3,4,5,6))

fp.to.matrix( list(fp1,fp2,fp3) )
```

 fplogical

Logical Operators for Fingerprints

Description

These functions perform logical operations (AND, OR, NOT, XOR) on the supplied binary fingerprints. Thus for two fingerprints A and B we have

& Logical AND

| Logical OR

xor Logical XOR

! Logical NOT (negation)

Arguments

e1 An object of class fingerprint

e2 An object of class fingerprint

Value

A fingerprint object

Author(s)

Rajarshi Guha (rguha@indiana.edu)

 length

Fingerprint Bit Length

Description

Returns the length of the fingerprint. That is, this is the length of the entire bit string and not simply the number of bits that are on.

Usage

```
## S4 method for signature 'fingerprint':
length(x)
```

Arguments

x An object of class fingerprint

Value

The length of the bit string

Author(s)

Rajarshi Guha (rguha@indiana.edu)

`random.fingerprint` *Generate Randomized Fingerprints*

Description

A utility function that can be used to generate binary fingerprints of a specified length with a specified number of bit positions (selected randomly) set to 1. Currently bit positions are selected uniformly

Usage

```
random.fingerprint(nbit, on)
```

Arguments

<code>nbit</code>	The length of the fingerprint, that is, the total number of bits. Must be a positive integer.
<code>on</code>	How many positions should be set to 1

Value

An object of class `fingerprint`

Author(s)

Rajarshi Guha (rguha@indiana.edu)

Examples

```
# make a fingerprint vector
fp <- random.fingerprint(32, 16)
as.character(fp)
```

show

String Representation of a Fingerprint

Description

Simply summarize the fingerprint.

Usage

```
## S4 method for signature 'fingerprint':  
show(object)
```

Arguments

object An object of class fingerprint

Author(s)

Rajarshi Guha (rguha@indiana.edu)

Index

! (*fplogical*), 13
!, fingerprint-method (*fplogical*),
13
*Topic **classes**
fingerprint-class, 7
*Topic **logic**
as.character, 2
cdk.lf, moe.lf, bci.lf, 3
distance, 4
euc.vector, 6
fingerprint-class, 7
fold, 8
fp.factor.matrix, 9
fp.read, fp.read.to.matrix,
10
fp.sim.matrix, 11
fp.to.matrix, 12
fplogical, 13
length, 13
random.fingerprint, 14
show, 15
*Topic **methods**
as.character, 2
fplogical, 13
length, 13
*Topic **programming**
bit.spectrum, 2
& (*fplogical*), 13
&, fingerprint, fingerprint-method
(*fplogical*), 13
| (*fplogical*), 13
|, fingerprint, fingerprint-method
(*fplogical*), 13

as.character, 2
as.character, fingerprint-method
(*as.character*), 2

bci.lf, 10
bci.lf (*cdk.lf, moe.lf, bci.lf*), 3

bit.spectrum, 2

cdk.lf, 10
cdk.lf (*cdk.lf, moe.lf, bci.lf*), 3
cdk.lf, moe.lf, bci.lf, 3

distance, 3, 4, 9, 11, 12
distance, fingerprint, fingerprint, character-method
(*fingerprint-class*), 7
distance, fingerprint, fingerprint, missing-method
(*fingerprint-class*), 7

euc.vector, 6
euc.vector, fingerprint-method
(*fingerprint-class*), 7

fingerprint-class, 7
fold, 8
fold, fingerprint-method
(*fingerprint-class*), 7
fp.factor.matrix, 8, 9
fp.read, 3, 8, 9, 11, 12
fp.read (*fp.read,*
fp.read.to.matrix), 10
fp.read, fp.read.to.matrix, 10
fp.read.to.matrix, 8
fp.read.to.matrix (*fp.read,*
fp.read.to.matrix), 10
fp.sim.matrix, 8, 11
fp.to.matrix, 8, 12
fplogical, 13

length, 13
length, fingerprint-method
(*length*), 13

moe.lf, 10
moe.lf (*cdk.lf, moe.lf, bci.lf*), 3

random.fingerprint, 8, 14

`random.fingerprint, numeric, numeric-method`
`(fingerprint-class), 7`

`show, 15`

`show, fingerprint-method (show), 15`

`xor (fplogical), 13`

`xor, fingerprint, fingerprint-method`
`(fplogical), 13`