Package ‘SteinerNet’

September 7, 2020

Type Package
Version 3.1.0
Date 2020-08-21
Title Steiner Tree Approach for Graph Analysis
Author Afshin Sadeghi <sadeghi.afshin@gmail.com>
Maintainer Aleksei Krasikov <krasikov.as@phystech.edu>
URL https://github.com/krashkov/SteinerNet
BugReports https://github.com/krashkov/SteinerNet/issues
Depends R (>= 3.1.0), igraph (>= 0.6.0)
Imports grDevices, stats, utils, graphics
Suggests knitr, rmarkdown, testthat
VignetteBuilder knitr
License GPL-3
Repository CRAN
NeedsCompilation no
Encoding UTF-8
RoxygenNote 7.1.1
Date/Publication 2020-09-07 09:50:08 UTC

R topics documented:

  generate_st_samples ........................................... 2
  steinertree ................................................... 3
  steiner_comparison_plots ..................................... 4
  steiner_comparison_wilcox ................................... 5
  steiner_simulation ........................................... 7

Index 9
generate_st_samples  

Select terminals

description

Provides random walk procedure. Starting from the randomly selected node, choose a neighbour node uniformly randomly, until given number of terminals won’t be found.

usage

generate_st_samples(graph, ter_number, prob)

arguments

graph  
an igraph graph; should be undirected, otherwise it is converted to undirected.

ter_number  
a numeric vector; each element indicates the number of terminals to be selected and length of vector indicates the number of terminal sets to be picked.

prob  
a numeric vector of the same length as ter_number; prob[i] defines a probability with which each next node accepted or rejected while selecting ter_number[i] terminals.

value

A list of the same length as ter_number. Each element of list contains a vector of ids of selected vertices.

references


examples

generate_st_samples(graph = graph("Zachary"),  
  ter_number = c(3, 4),  
  prob = c(0.1, 0.2))
steinertree

Find Steiner Tree

Description

A set of functions for finding Steiner Tree. Includes both exact and heuristic approaches.

Usage

steinertree(type, repeattimes = 70, optimize = TRUE, terminals, graph, color = TRUE, merge = FALSE)

Arguments

- **type**: a character scalar, which indicates type of algorithms to perform. Can be "EXA", "SP", "KB", "RSP", "SPM" or "ASP".
- **repeattimes**: a numeric scalar to specify "RSP" algorithm; number of times the optimization procedure is repeated.
- **optimize**: a logical scalar to specify all algorithms except "EXA"; if TRUE, an optimization of the resultant steiner tree is performed, otherwise nothing is done.
- **terminals**: a numeric vector (ids of terminals are passed) or character vector (vertices must have 'name' attribute).
- **graph**: an igraph graph; should be undirected, otherwise it is converted to undirected.
- **color**: a logical scalar; whether to return an original graph with terminals colored in red and steiner nodes colored in green. Note, if several trees will be found, steiner nodes from all trees are colored in green.
- **merge**: a logical scalar to specify "EXA" and "SPM" algorithms; if several trees will be found, whether to return a list with trees or merge them

Details

If input graph doesn’t have 'name' attribute, one is created. In this case it will contain character ids of vertices. Also before execution all vertices will be colored in yellow and terminals will be colored in red.

Value

(color = FALSE) Returns a list first element of which is a steiner tree (or a graph of merged trees). If several steiner trees are found, return a list, each element of which is a steiner tree.

(color = TRUE) Returns a list, first element of which is a colored original graph and second element is a steiner tree (or a graph of merged trees) or list of steiner trees.
References

1. Path heuristic and Original path heuristic ,Section 4.1.3 of the book "The Steiner tree Problem", PetterL, Hammer

See Also

generate_st_samples

Examples

steinertree(type = "RSP", optimize = FALSE,
    terminals = c(1, 3),
    graph = graph("Cubical"),
    color = TRUE, merge = FALSE)

steiner_comparison_plots

Plot simulated data

Description

This function plots the comparison results of simulated data and stores it in PDF file.

Usage

steiner_comparison_plots(type, method, data, outputname)

Arguments

type        a character vector, which indicates types of algorithms to analyse. Can be "EXA", "SP", "KB", "RSP", "SPM" or "ASP".
method      a character vector; specifies a calculated parameter based on which comparison is performed. Can be "runtime" (for time of execution), "log10runtime", "edge" (for number of edges in resultant steiner tree), "log10edge", "ter_freq" (for terminal frequency in resultant steiner tree) or "edge_dens" (for edge density in resultant steiner tree).
data        should have structure as output of steiner_simulation function.
outputname  a character scalar; name of file in which the result is stored
Value

For each method a plot with comparison of algorithms (pointed in type variable) is created. An additional information about the number of experiments and number of terminals for each type of algorithm is added. If the number of terminals is the same for each type, then their number is printed, otherwise the range is printed.

References


See Also

generate_st_samples, steiner_simulation, steinertree

Examples

g <- graph("Cubical")

data <- steiner_simulation(type = c("SP", "KB", "SPM"),
                               graph = g,
                               ter_list = generate_st_samples(graph = g,
                                                   ter_number = c(2, 3),
                                                   prob = c(0.1, 0.2)))

steiner_comparison_plots(type = c("SP", "KB"),
                         method = c("runtime", "ter_freq"),
                         data = data,
                         outputname = tempfile(pattern = "file", tmpdir = tempdir()))

steiner_comparison_wilcox

Perform wilcox test

Description

Perform pairwise Wilcoxon rank sum tests

Usage

steiner_comparison_wilcox(type, method, data)
**Arguments**

- **type**: a character vector, which indicates type of algorithms to analyse. Can be "EXA", "SP", "KB", "RSP", "SPM" or "ASP".

- **method**: a character scalar; specifies a calculated parameter based on which comparison is performed. Can be "runtime" (for time of execution), "log10runtime" "edge" (for number of edges in resultant steiner tree), "log10edge", "ter_freq" (for terminal frequency in resultant steiner tree) or "edge_dens" (for edge density in resultant steiner tree).

- **data**: should have structure as output of steiner_simulation function.

**Details**

"holm" method for adjusting p-values is used.

**Value**

Object of class "pairwise.htest"

**References**


**See Also**

`generate_st_samples`, `steiner_simulation`, `steinertree`, `pairwise.wilcox.test`

**Examples**

```r
g <- graph("Cubical")
data <- steiner_simulation(type = c("SP", "KB", "SPM"),
                           graph = g,
                           ter_list = generate_st_samples(graph = g,
                                                  ter_number = c(2, 3),
                                                  prob = c(0.1, 0.2)))
steiner_comparison_wilcox(type = c("SP", "KB"),
                          method = "ter_freq",
                          data = data)
```
steiner_simulation

`steiner_simulation(type, repeattimes = 70, optimize = TRUE, graph, ter_list)`

**Description**

This function executes given Steiner Algorithms and calculates such parameters as runtime of each algorithm, number of edges, number of vertices, terminal frequency, edge density of resultant Steiner tree and number of trees in case of "EXA" or "SPM" algorithm. Wraps steinertree function.

**Usage**

steiner_simulation(type, repeattimes = 70, optimize = TRUE, graph, ter_list)

**Arguments**

- **type**: a character vector, which indicates types of algorithms to perform. Can be "EXA", "SP", "KB", "RSP", "SPM" or "ASP".
- **repeattimes**: a numeric scalar to specify "RSP" algorithm; number of times the optimization procedure is repeated.
- **optimize**: a logical scalar to specify all algorithms in type variable (ignored for "EXA"); if TRUE, an optimization of the resultant steiner tree is performed, otherwise nothing is done.
- **graph**: an igraph graph; should be undirected, otherwise it is converted to undirected.
- **ter_list**: a list each element of which contains a numeric or character vector with ids of terminals or a character vector with names of vertices (only if vertices have name attribute). Therefore, length of ter_list declares a number of experiments to perform with different terminal sets for each type of algorithm.

**Details**

As a ter_list, a vector can be passed. In this case it is converted to a list containing only one element.

**Value**

List, each element of which corresponds to specific algorithms type. Each element of a list also is a list and contains a named vector of evaluated parameters of steiner tree (runtime, edges_num, vert_num, trees_num, term_freq, edge_den). Number of sublist corresponds to number of terminal set in ter_list.

**References**

See Also

generate_st_samples, steinertree

Examples

g <- graph("Cubical")

steiner_simulation(type = c("SP", "KB", "SPM"),
                   graph = g,
                   ter_list = generate_st_samples(graph = g,
                                                   ter_number = c(2, 3),
                                                   prob = c(0.1, 0.2)))

steiner_simulation(type = c("EXA", "RSP"),
                    graph = g,
                    ter_list = c(1, 3, 8))
Index

generate_st_samples, 2, 4–6, 8

steiner_comparison_plots, 4
steiner_comparison_wilcox, 5
steiner_simulation, 5, 6, 7
steinertree, 3, 5, 6, 8