Package 'Lab6'

October 20, 2015

Type Package

Title knapsack-algos

Version 1.0

Date 2015-10-06

Author Niclas Lovsj<c3><b6>, Maxime Bonneau

Maintainer <niclas.lovsjo@me.com>

Description tests different approaches to solving the knapsack-problem. Incl. a brute force, dynamic programming and greedy algorithm

License GPL-2

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

R topics documented:

`Lab6-package` *knapsack-algos*

Description

tests different approaches to solving the knapsack-problem. Incl. a brute force, dynamic programming and greedy algorithm

Details

The DESCRIPTION file:

Package:	Lab6
Type:	Package
Title:	knapsack-algos
Version:	1.0
Date:	2015-10-06
Author:	Niclas Lovsjö, Maxime Bonneau
Maintainer:	<niclas.lovsjo@me.com>
Description:	tests different approaches to solving the knapsack-problem. Incl. a brute force, dynamic programming
License:	GPL-2
Suggests:	testthat, knitr, rmarkdown
VignetteBuilder:	knitr

Index of help topics:

~~ An overview of how to use the package, including the most important ~~ ~~ functions ~~

Author(s)

Niclas Lovsjö, Maxime Bonneau

Maintainer: <niclas.lovsjo@me.com>

References

~~ Literature or other references for background information ~~

See Also

~~ Optional links to other man pages, e.g. ~~ ~~ <pkg> ~~

Examples

```
~~ simple examples of the most important functions ~~
```

Brute force — *Brute force algorithm for the knapsack problem*

Description

uses brute force, i.e. tests all combinations and finds the one with max value under the restriction total weight<W.

Arguments

x	is a 2dim matrix containing the weights and values
W	is the capacity of the knapsack

Value

a list with the _value_ of the optimally packed knapsack and the _elements_ that gives this value.

References

https://en.wikipedia.org/wiki/Knapsack_problem

Brute force parallel — *Brute force algorithm for the knapsack problem using parallelizing*

Description

uses brute force, i.e. tests all combinations and finds the one with max value under the restriction total weight<W.

uses brute force, i.e. tests all combinations and finds the one with max value under the restriction total weight<W.

Arguments

x	is a 2dim matrix containing the weights and values
W	is the capacity of the knapsack
x	is a 2dim matrix containing the weights and values
W	is the capacity of the knapsack

Value

a list with the _value_ of the optimally packed knapsack and the _elements_ that gives this value.

a list with the _value_ of the optimally packed knapsack and the _elements_ that gives this value.

References

https://en.wikipedia.org/wiki/Knapsack_problem

https://en.wikipedia.org/wiki/Knapsack_problem

Dynamic programming *Dynamic programming algorithm for the knapsack problem*

Description

Uses DP to find optimal value and elements, i.e. divides the problem into subproblems and solve each one, memoizes it and solve the whole problem by using that

Arguments

x	is a 2dim matrix containing the weights and values
W	is the capacity of the knapsack

Value

a list with the _value_ of the optimally packed knapsack and the _elements_ that gives this value.

References

https://en.wikipedia.org/wiki/Knapsack_problem

Greedy heuristic *Greedy heuristic for the knapsack problem*

Description

Uses greedy heuristic to solve knapsack problem, i.e. orders x by ratio v/w and picks up the first lines until the knapsack is full (or almost)

Arguments

x	is a 2dim matrix containing the weights and values
W	is the capacity of the knapsack

Value

a list with the _value_ of the optimally packed knapsack and the _elements_ that gives this value.

References

https://en.wikipedia.org/wiki/Knapsack_problem#Greedy_approximation_algorithm

Index