Package ‘uta.mondisc’

September 21, 2014

Type Package

Title Inferring additive value functions with unknown monotonicity

Version 0.1-1

Date 2014-08-02

Author Jakub Wasikowski

Maintainer Jakub Wasikowski <`jakub.wasikowski@gmail.com`>

URL https://github.com/Mostesz/uta.mondisc

Depends Rglpk (>= 0.5-1), sets (>= 1.0-13)

Description TODO

License TODO

Suggests testthat (>= 0.7.1)

R topics documented:

buildProblem *Build problem*

Description

This function allows to build instance of the problem.

Usage

```
buildProblem(alternatives, margValueFuncShapes, M, strictPreferences = NULL,
  weakPreferences = NULL, indifferences = NULL)
```

Arguments

`alternatives`	Matrix of alternatives values for criteria. Criteria arranged in columns, alternatives in rows. Number of alternatives (rows) must be greater than 3. Number of criteria (columns) must be greater than 1.
`margValueFuncShapes`	Vector of criterion types, which correspond to each criterion. Each value of vector must be one of either "GAIN", "COST", "NOT_PREDEFINED", "A_TYPE", "V_TYPE", "NON_MON", which represent knowledge about the monotonicity for criteria, respectively criterion is of predefined gain type (GAIN), pre-defined cost type (COST), monotonic, but the type of the monotonicity is not pre-defined (NOT_PREDEFINED), criterion for which the most preferred evaluation is possibly not in the extreme point of evaluation scale (A_TYPE), criterion for which the least preferred evaluation is possibly not in the extreme point of evaluation scale (V_TYPE) and criterion with no prior information on the monotonicity (NON_MON). Index of each type must correspond to index of column of the criterion in alternatives matrix.
`M`	Big positive numeric value which is much greater that any value of each alternative.
`strictPreferences`	Matrix of the strict preferences provided by DM. Each row represents comparision between two alternatives and it means that alternative provided in the first column is strictly preferred over alternative provided in the second column. Value of each matrix cell must correspond to index of alternative defined in alternatives matrix.
`weakPreferences`	Matrix of the weak preferences provided by DM. Each row represents comparision between two alternatives and it means that alternative provided in the first column is weakly preferred over alternative provided in the second column. Value of each matrix cell must correspond to index of alternative defined in alternatives matrix.
`indifferences`	Matrix of the indifferences provided by DM. Each row represents comparision between two alternatives and it means that alternative provided in the first column is indifferent for alternative provided in the second column. Value of each matrix cell must correspond to index of alternative defined in alternatives matrix.

Value

Instance of the problem

Examples

```
alternatives = matrix(c(100, 1000,  9,   110,  300,  800,
                        90,  1100,  11,  120,  310,  801,
                        100, 1500,  12,  100,  340,  803,
                        120, 800,   13,  105,  360,  809), ncol = 6, byrow=TRUE);
margValueFuncShapes = c("GAIN", "COST", "NOT_PREDEFINED", "A_TYPE", "V_TYPE", "NON_MON");
M = 1000000;

strictPreferences = matrix(c(1,2), ncol=2, byrow=TRUE);
weakPreferences = matrix(c(2,3), ncol=2, byrow=TRUE);
indifferences = matrix(c(3,4), ncol=2, byrow=TRUE);

problem = buildProblem (
  alternatives,
  margValueFuncShapes,
  M,
  strictPreferences = strictPreferences,
  weakPreferences = weakPreferences,
  indifferences = indifferences
)
```

`calcSolution` *Calculate solution*

Description

This function allows to calculate solution of the problem

Usage

```
calcSolution(problem)
```

Arguments

`problem`	Instance of the problem

Value

Result of the LP calculation

Examples

```
alternatives = matrix(c(100, 1000,  9,   110,  300,  800,
                        90,  1100,  11,  120,  310,  801,
                        100, 1500,  12,  100,  340,  803,
                        120, 800,   13,  105,  360,  809), ncol = 6, byrow=TRUE);
margValueFuncShapes = c("GAIN", "COST", "NOT_PREDEFINED", "A_TYPE", "V_TYPE", "NON_MON");
M = 1000000;

strictPreferences = matrix(c(1,2), ncol=2, byrow=TRUE);
weakPreferences = matrix(c(2,3), ncol=2, byrow=TRUE);
indifferences = matrix(c(3,4), ncol=2, byrow=TRUE);

problem = buildProblem (
  alternatives,
  margValueFuncShapes,
  M,
  strictPreferences = strictPreferences,
  weakPreferences = weakPreferences,
  indifferences = indifferences
)
lpresult = calcSolution(problem);
```

Index