Package 'james.analysis'

August 29, 2016

Type Package

Title Analysis Tools for the 'JAMES' Framework

Version 1.0.1

Date 2015-06-18

Author Herman De Beukelaer

Maintainer Herman De Beukelaer <Herman.DeBeukelaer@UGent.be>

Description Analyze and visualize results of studies performed with the analysis tools in 'JAMES', a modern object-oriented Java framework for discrete optimization using local search metaheuristics (see http://www.jamesframework.org).

License MIT + file LICENSE

Depends R (>= 2.10), graphics (>= 2.15.3)

Imports rjson (>= 0.2.14), naturalsort (>= 0.1.2)

Suggests testthat (>= 0.9.1)

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2015-06-18 18:00:29

R topics documented:

`boxplot.james` *Solution quality and convergence time box plots*

Description

Produce box-and-whisker plots for the searches that have been applied to the given problem, visualizing the distribution of the best found solution's value (solution quality) or time until convergence in subsequent search runs.

Usage

```
## S3 method for class 'james'
boxplot(x, problem, type = c("quality", "time"), r = 0.99,
  time.unit = c("milliseconds", "seconds", "minutes", "hours"), title,
  subtitle, ylab, names, ...)
```

Arguments

`x`	data object containing the analysis results
`problem`	name of the problem for which the plot is made. Can be omitted if the data `x` contains results for a single problem only.
`type`	one of `"quality"` (default) or `"time"`. If set to `"quality"`, the final solution's value is reported; if set to `"time"`, the time until convergence is reported. In both cases, the respective distribution of values found during the different search runs is visualized. In the latter case, the argument `r` is used to decide when a search run has converged.
`r`	convergence ratio, only used if `type` is `"time"`. Defaults to 0.99. Should be a numeric value in [0,1]. This parameter is passed to `getConvergenceTimes`.
`time.unit`	one of `"milliseconds"` (default), `"seconds"`, `"minutes"` or `"hours"`. Only used if `type` is `"time"`. Determines the time unit of the convergence times on the `y`-axis.
`title`	plot title. Defaults to `"Solution quality"` or `"Convergence time"` when `type` is set to `"quality"` or `"time"`, respectively.
`subtitle`	plot subtitle. By default, a subtitle is added that states the name of the problem for which the plot is made. If `type` is `"time"` the subtitle also mentions the applied convergence ratio `r`. If no subtitle is desired set `subtitle = ""`.
`ylab`	y-axis label. Defaults to `"Value"` or `"Time"` (with the time unit indicated between brackets) when `type` is set to `"quality"` or `"time"`, respectively.

names	names to be shown on the x-axis under the box plots. Defaults to the search names obtained from calling getSearches for the given data x and problem.
...	any additional parameters are passed to boxplot.

Details

If the data x contains results for a single problem only, the argument problem can be omitted. If desired to produce box plots for a selection of the applied searches, use reduceJAMES to extract the respective data.

Convergence times are computed with getConvergenceTimes.

The plots are made using the generic boxplot method called on a list of vectors containing the distribution samples for each search.

Any additional parameters are passed to boxplot.

See Also

getConvergenceTimes, boxplot

---

getBestSolutions *Get best found solutions*

Description

Get the best found solutions during the different runs of the given search applied to the given problem. This is a generic S3 method.

Usage

```
getBestSolutions(data, problem, search)
```

Arguments

data	data object containing the analysis results
problem	name of the analyzed problem. Can be omitted if the data contains results for a single problem only.
search	name of the applied search. Can be omitted if the data contains results for a single search only (for the considered problem).

Details

If the data contains results for a single problem only, the argument problem can be omitted. Likewise, if for the considered problem results are available for a single search only, the argument search can be omitted.

When writing results obtained from the analysis tools in the 'JAMES' extensions module to a JSON file, one should provide a JSON converter for the solution type of the analyzed problems if it is desired that the actual best found solutions are contained in the output file. Therefore, these solutions

might not be available for all problems, searches or search runs. In case a best solution is missing for a search run, the corresponding entry in the returned list will be set to NA. It is possible that a list of only NA is returned.

Value

List containing the best found solutions during each run. May contain NA values.

`getBestSolutionValues` *Get values of best found solutions*

Description

Get the values of the best found solutions during all runs of the given `search` applied to the given `problem`. This is a generic S3 method.

Usage

```
getBestSolutionValues(data, problem, search)
```

Arguments

`data`	data object containing the analysis results
`problem`	name of the analyzed problem. Can be omitted if the `data` contains results for a single problem only.
`search`	name of the applied search. Can be omitted if the `data` contains results for a single search only (for the considered `problem`).

Details

If the `data` contains results for a single problem only, the argument `problem` can be omitted. Likewise, if for the considered `problem` results are available for a single search only, the argument `search` can be omitted.

Value

Numeric vector containing the values of the best found solutions during each run.

getConvergenceTimes *Get convergence times*

Description

Get the convergence times of the different runs of the given search applied to the given problem (in milliseconds). This is a generic S3 method.

Usage

```
getConvergenceTimes(data, problem, search, r = 0.99)
```

Arguments

data	data object containing the analysis results
problem	name of the analyzed problem. Can be omitted if the data contains results for a single problem only.
search	name of the applied search. Can be omitted if the data contains results for a single search only (for the considered problem).
r	convergence ratio. Defaults to 0.99. Numeric value in [0,1].

Details

If the data contains results for a single problem only, the argument problem can be omitted. Likewise, if for the considered problem results are available for a single search only, the argument search can be omitted.

The convergence time of a search run is defined as the time at which a certain value threshold is crossed. This threshold is computed from the given convergence ratio r as follows: if values are being maximized, thr = (1-r)*min + r*max; else, thr = (1-r)*max + r*min, where min and max are the minimum and maximum observed value in the considered search run, respectively. In case of maximization, a search run is said to have converged as soon as it reaches a value which is larger than or equal to the threshold thr. In case of minimization, convergence occurs when the values drop below the threshold. The convergence ratio r defaults to 0.99. If set to 1, a search run is said to have converged when the final best solution is found.

Value

Numeric vector containing the convergence times of each run (in milliseconds). All convergence times are greater than or equal to -1.

getNumSearchRuns *Get number of applied search runs*

Description

Get the number of applied runs of the given `search` when solving the given `problem`. This is a generic S3 method.

Usage

```
getNumSearchRuns(data, problem, search)
```

Arguments

`data`	data object containing the analysis results
`problem`	name of the analyzed problem. Can be omitted if the `data` contains results for a single problem only.
`search`	name of the applied search. Can be omitted if the `data` contains results for a single search only (for the considered `problem`).

Details

If the `data` contains results for a single problem only, the argument `problem` can be omitted. Likewise, if – for the considered `problem` – results are available for a single search only, the argument `search` can be omitted.

Value

numeric: number of applied search runs

getProblems *Get names of analyzed problems*

Description

Extracts the names of the problems for which analysis results are contained in the given data. This is a generic S3 method.

Usage

```
getProblems(data, filter, ...)
```

Arguments

`data`	data object containing the analysis results
`filter`	regular expression (optional). Only problem names that match the given regex are returned, if any.
`...`	any additional arguments are passed to `grep`.

Details

Problem names are sorted using naturalsort. If a `filter` is set, only those problem names matching the given regular expression are returned (pattern matching is done with `grep`).

Value

Sorted vector of strings containing the names of all analyzed problems that occur in the given data and match the applied filter (if any).

`getSearches` *Get names of applied searches*

Description

Extracts the names of all searches that have been applied to the given `problem`.This is a generic S3 method.

Usage

```
getSearches(data, problem, filter, ...)
```

Arguments

`data`	data object containing the analysis results
`problem`	name of the analyzed problem. Can be omitted if the `data` contains results for a single problem only.
`filter`	regular expression (optional). Only search names that match the given regex are returned, if any.
`...`	any additional arguments are passed to `grep`.

Details

Search names are sorted using naturalsort. If the `data` contains results for a single problem only, the argument `problem` can be omitted. If a `filter` is set, only those search names matching the given regular expression are returned (pattern matching is done with `grep`).

Value

Sorted vector of strings containing the names of all searches that have been applied to the given problem and match the applied filter (if any).

getSearchRuns *Get search run data*

Description

Extract the data corresponding to the subsequent runs of a specific `search` being applied to a specific `problem`. This is a generic S3 method.

Usage

```
getSearchRuns(data, problem, search)
```

Arguments

`data`	data object containing the analysis results
`problem`	name of the analyzed problem. Can be omitted if the `data` contains results for a single problem only.
`search`	name of the applied search. Can be omitted if the `data` contains results for a single search only (for the considered `problem`).

Details

If the `data` contains results for a single problem only, the argument `problem` can be omitted. Likewise, if for the considered `problem` results are available for a single search only, the argument `search` can be omitted.

Value

A list containing one element for each search run.

Each run has at least two elements `time` and `values`, which are both numeric vectors. The `time` vector indicates when the best solution was updated during search and the new best solution's value is found at the respective index in `values`. Times are expressed in milliseconds since starting the search. A time of -1 indicates that the search was not yet running, which e.g. occurs when a local search adopts a random current solution during initialization. Times are always positive (or -1) and increasing. Values are either increasing (in case of maximization) or decreasing (in case of minimization).

If contained in the given `data`, a run also has an element `best.solution` representing the final best solution found during that search run. The last element of `values` then indicates the value of this best solution. When writing results obtained from the analysis tools in the 'JAMES' extensions module to a JSON file, one should provide a JSON converter for the solution type of the analyzed problems if it is desired that the actual best found solutions are contained in the output file.

james *Results of example algorithm comparison*

Description

Contains results of an example analysis performed with the 'JAMES' extensions module. The performance of two algorithms is compared (random descent and parallel tempering) for a core selection problem in which the mean entry-to-nearest-entry distance is maximized. Four different data sets have been analyzed. Details about the performed analysis are provided at the website (see below).

Usage

```
james
```

Format

S3 object of class "james", as if produced by readJAMES.

Source

http://www.jamesframework.org/examples/#analysis

See Also

readJAMES

Examples

```
# load data
data(james)
summary(james)

# plot convergence curves for coconut data set
plotConvergence(james, problem = "coconut", min.time = 1000, max.time = 100000)

# create box plots of solution values (quality) and convergence times
boxplot(james, problem = "coconut")
boxplot(james, problem = "coconut", type = "time")

# extract solution values and convergence times for parallel tempering and random descent
values.pt <- getBestSolutionValues(james, problem = "coconut", search = "Parallel Tempering")
times.pt <- getConvergenceTimes(james, problem = "coconut", search = "Parallel Tempering")
values.rd <- getBestSolutionValues(james, problem = "coconut", search = "Random Descent")
times.rd <- getConvergenceTimes(james, problem = "coconut", search = "Random Descent")

# perform wilcoxon test to compare distributions across algorithms
values.test <- wilcox.test(values.pt, values.rd)
values.test
```

```
times.test <- wilcox.test(times.pt, times.rd)
times.test

# adjust p-values for multiple testing
p.adjust(c(values.test$p.value, times.test$p.value))
```

`james.analysis` *Analysis Tools for the 'JAMES' Framework.*

Description

This package can be used to further analyze and visualize results of studies performed with the analysis tools in 'JAMES', a modern object-oriented Java framework for discrete optimization using local search metaheuristics (see references). Functions are provided to plot convergence curves, draw box plots of solution quality or convergence times and to summarize, manipulate or extract data from the results.

Details

Package:	james.analysis
Type:	Package
Version:	1.0.1
Date:	2015-06-18
License:	MIT

Example data

`james`

Data manipulation functions

- `readJAMES`
- `reduceJAMES`
- `mergeJAMES`
- `getProblems`
- `getSearches`
- `getSearchRuns`
- `getNumSearchRuns`
- `getBestSolutionValues`
- `getBestSolutions`
- `getConvergenceTimes`

Plot functions

- plotConvergence
- boxplot.james

Author(s)

Herman De Beukelaer <<Herman.DeBeukelaer@UGent.be>>

References

'JAMES' Website: http://www.jamesframework.org

Examples

```
# load example data
data(james)
summary(james)

# plot convergence curves for coconut data set
plotConvergence(james, problem = "coconut", min.time = 1000, max.time = 100000)

# create box plots of solution values (quality) and convergence times
boxplot(james, problem = "coconut")
boxplot(james, problem = "coconut", type = "time")

# extract solution values and convergence times for parallel tempering and random descent
values.pt <- getBestSolutionValues(james, problem = "coconut", search = "Parallel Tempering")
times.pt <- getConvergenceTimes(james, problem = "coconut", search = "Parallel Tempering")
values.rd <- getBestSolutionValues(james, problem = "coconut", search = "Random Descent")
times.rd <- getConvergenceTimes(james, problem = "coconut", search = "Random Descent")

# perform wilcoxon test to compare distributions across algorithms
values.test <- wilcox.test(values.pt, values.rd)
values.test
times.test <- wilcox.test(times.pt, times.rd)
times.test

# adjust p-values for multiple testing
p.adjust(c(values.test$p.value, times.test$p.value))
```

---

mergeJAMES *Merge analysis results*

Description

Merge results from different analyses. If runs of the same search applied to the same problem are found in both data sets, these runs are merged into a single list. This is a generic S3 method.

Usage

```
mergeJAMES(data1, data2)
```

Arguments

data1	results from the first analysis (of same class as data2).
data2	results from the second analysis (of same class as data1).

Value

merged data (assigned classes are retained)

---

plotConvergence *Plot convergence curves*

Description

Creates a plot showing the convergence curve of each search that has been applied to the given problem, aggregated over all search runs (mean or median). This is a generic S3 method.

Usage

```
plotConvergence(data, problem, type = c("mean", "median"), col = "black",
  plot.type = "s", lty, title = "Convergence curve(s)", subtitle, xlab,
  ylab = "Value", time.unit = c("milliseconds", "seconds", "minutes",
  "hours"), min.time, max.time, legend = TRUE, legend.pos,
  legend.inset = c(0.02, 0.05), legend.names, ...)
```

Arguments

data	data object containing the analysis results
problem	name of the problem for which the plot is made. Can be omitted if the data contains results for a single problem only.
type	one of "mean" (default) or "median". Determines how the values from the different search runs are aggregated.
col	color(s) of the plotted lines and/or symbols, used cyclically when providing a vector. Defaults to "black".
plot.type	defaults to "s" (staircase). See matplot and plot for more information about the possible plot types.
lty	line type(s), used cyclically when providing a vector. Line types default to 1:n where n is the number of plotted curves.
title	plot title. Defaults to "Convergence curve(s)".
subtitle	plot subtitle. By default, a subtitle will be added that states the name of the problem for which the plot was made. If no substitle is desired, set subtitle = "".

xlab	x-axis label. Defaults to "Time" followed by the time unit within brackets (abbreviated).
ylab	y-axis label. Defaults to "Value".
time.unit	one of "milliseconds" (default), "seconds", "minutes" or "hours". Determines the time unit of the values on the x-axis.
min.time	zoom in on the part of the curve(s) after this point in time on the x-axis, according to the specified time.unit.
max.time	zoom in on the part of the curve(s) before this point in time on the x-axis, according to the specified time.unit.
legend	logical: indicates whether a legend should be added to the plot. Defaults to TRUE.
legend.pos	position of the legend, specified as a keyword keyword from the list "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center". Defaults to "bottomright" in case values are being maximized or "topright" in case of minimization.
legend.inset	inset distance(s) from the margins as a fraction of the plot region. If a single value is given, it is used for both margins; if two values are given, the first is used for x-distance, the second for y-distance. Defaults to c(0.02, 0.05).
legend.names	names to be shown in the legend. Defaults to the search names obtained from calling getSearches for the given data and problem.
...	optional other arguments passed to matplot.

Details

If the data contains results for a single problem only, the argument problem can be omitted. If desired to plot convergence curves for a selection of the applied searches, use reduceJAMES to extract the respective data.

The curves are plotted using matplot. More information about the graphical parameters are provided in the documentation of this function. By default, a legend is added to the plot. This can be omitted by setting legend = FALSE. If desired, a custom legend may then be added. It is possible to zoom in on a specific region of the plot using the parameters min.time and max.time.

Any additional parameters are passed to matplot.

See Also

matplot

---

readJAMES *Read analysis results from JSON file*

Description

Read results from a JSON file produced by the analysis tools in the 'JAMES' extensions module.

Usage

```
readJAMES(file)
```

Arguments

file	string: path to a JSON file containing results produced by the analysis tools from the 'JAMES' extensions module.

Value

S3 object of class "james" containing the results of running a number of searches on a set of problems, where each search has been repeatedly applied for a number of runs. Data can be manipulated and extracted using the provided functions (see below).

See Also

Example data: james.

Data access and manipulations methods: reduceJAMES, mergeJAMES, getProblems, getSearches, getSearchRuns, getNumSearchRuns, getBestSolutionValues, getBestSolutions, getConvergenceTimes.

Plot functions: plotConvergence, boxplot.james.

Examples

```
# get path to raw JSON file included in package distribution
json.file <- system.file("extdata", "james.json", package = "james.analysis")

# read results from file
james <- readJAMES(json.file)
summary(james)

# plot convergence curves for coconut data set
plotConvergence(james, problem = "coconut", min.time = 1000, max.time = 100000)

# create box plots of solution values (quality) and convergence times
boxplot(james, problem = "coconut")
boxplot(james, problem = "coconut", type = "time")

# extract solution values and convergence times for parallel tempering and random descent
values.pt <- getBestSolutionValues(james, problem = "coconut", search = "Parallel Tempering")
times.pt <- getConvergenceTimes(james, problem = "coconut", search = "Parallel Tempering")
values.rd <- getBestSolutionValues(james, problem = "coconut", search = "Random Descent")
times.rd <- getConvergenceTimes(james, problem = "coconut", search = "Random Descent")

# perform wilcoxon test to compare distributions across algorithms
values.test <- wilcox.test(values.pt, values.rd)
values.test
times.test <- wilcox.test(times.pt, times.rd)
times.test

# adjust p-values for multiple testing
p.adjust(c(values.test$p.value, times.test$p.value))
```

`reduceJAMES` *Reduce analysis results to selected problems and searches*

Description

Reduce the given `data` by filtering the analyzed problems and applied searches based on the given list of names or regular expression (pattern matching is done with `grep`). This is a generic S3 method.

Usage

```
reduceJAMES(data, problems = ".*", searches = ".*", ...)
```

Arguments

`data`	data object containing the analysis results
`problems`	regular expression or list of strings. Only those problems that match the regular expression or occur in the list are retained.
`searches`	regular expression or list of strings. Only those searches that match the regular expression or occur in the list are retained.
`...`	any additional arguments are passed to `grep`.

Value

Reduced data set containing only those problems and searches whose names match the respective regular expression or occur in the respective list of strings. Assigned classes are retained.

Index