Package 'processmapR'

February 24, 2019

Type Package

Title Construct Process Maps Using Event Data

Version 0.3.3

Date 2019-02-22

Description Visualize of process maps based on
event logs, in the form of directed graphs. Part of the 'bupaR' framework.

License MIT + file LICENSE

Imports dplyr, bupaR (>= 0.4.0), edeaR (>= 0.8.0), DiagrammeR (>=
1.0.0), ggplot2, ggthemes, stringr, purrr, data.table, shiny,
miniUI, glue, forcats, hms, RColorBrewer, plotly, rlang,
scales, tidyr

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

URL https://www.bupar.net

NeedsCompilation no

Author Gert Janssenswillen [aut, cre],
Benoît Depaire [ctb],
Felix Mannhardt [ctb],
Thijs Beuving [ctb]

Maintainer Gert Janssenswillen <gert.janssenswillen@uhasselt.be>

Repository CRAN

Date/Publication 2019-02-24 18:00:15 UTC

R topics documented:

`custom` *Custom map profile*

Description

Function to create a custom map profile based on some event log attribute.

Usage

```
custom(FUN = mean, attribute, units = "", color_scale = "PuBu",
  color_edges = "dodgerblue4")
```

Arguments

`FUN`	A summary function to be called on the provided event attribute, e.g. mean, median, min, max. na.rm = T by default.
`attribute`	The name of the case attribute to visualize (should be numeric)
`units`	Character to be placed after values (e.g. EUR for monitary euro values)
`color_scale`	Name of color scale to be used for nodes. Defaults to PuBu. See 'Rcolorbrewer::brewer.pal.info()' for all options.
`color_edges`	The color used for edges. Defaults to dodgerblue4.

Details

If used for edges, it will show the attribute values which related to the out-going node of the edge.#'

Examples

```
## Not run:
library(eventdataR)
library(processmapR)
data(traffic_fines)
# make sure the amount attribute is propagated forward in each trace
# using zoo::na.locf instead of tidyr::fill since it is much faster
# still the whole pre-processing is still very slow
```

```
library(zoo)

traffic_fines_prepared <- traffic_fines %>%
filter_trace_frequency(percentage = 0.8) %>%
group_by_case() %>%
mutate(amount = na.locf(amount, na.rm = F)) %>%
ungroup_eventlog()

process_map(traffic_fines_prepared, type_nodes = custom(attribute = "amount", units = "EUR"))

## End(Not run)
```

`dotted_chart` *Dotted chart*

Description

Create a dotted chart to view all events in a glance

Usage

```
dotted_chart(eventlog, x, sort, color, units, ...)

## S3 method for class 'eventlog'
dotted_chart(eventlog, x = c("absolute", "relative",
  "relative_week", "relative_day"), sort = c("start", "end", "duration",
  "start_week", "start_day"), color = NULL, units = c("weeks", "days",
  "hours", "mins", "secs"), ...)

## S3 method for class 'grouped_eventlog'
dotted_chart(eventlog, x = c("absolute",
  "relative", "relative_week", "relative_day"), sort = c("start", "end",
  "duration", "start_week", "start_day"), color = NULL, units = c("weeks",
  "days", "hours", "mins", "secs"), ...)

idotted_chart(eventlog, plotly = FALSE)

iplotly_dotted_chart(eventlog)

plotly_dotted_chart(eventlog, x = c("absolute", "relative", "relative_week",
  "relative_day"), sort = c("start", "end", "duration", "start_week",
  "start_day"), color = NULL, units = c("weeks", "days", "hours", "mins",
  "secs"), ...)
```

Arguments

`eventlog`	Eventlog object

x	Value for plot on x-axis: absolute time or relative time (since start, since start of week, since start of day)
sort	Ordering of the cases on y-axis: start, end or duration
color	Optional, variable to use for coloring dots. Default is the activity identifier. Use NA for no colors.
units	Time units to use on x-axis in case of relative time.
...	Deprecated arguments
plotly	Return plotly object

Methods (by class)

- eventlog: Dotted chart for event log
- grouped_eventlog: Dotted chart for grouped event log

---

frequency *Frequency map profile*

Description

Function to create a frequency profile for a process map.

Usage

```
frequency(value = c("absolute", "relative", "absolute-case", "relative-case"),
  color_scale = "PuBu", color_edges = "dodgerblue4")
```

Arguments

value	The type of frequency value to be used: absolute, relative (percentage of activity instances) or relative_case (percentage of cases the activity occurs in).
color_scale	Name of color scale to be used for nodes. Defaults to PuBu. See 'Rcolorbrewer::brewer.pal.info()' for all options.
color_edges	The color used for edges. Defaults to dodgerblue4.

performance *Performance map profile*

Description

Function to create a performance map profile to be used as the type of a process map. It results in a process map describing process time.

Usage

```
performance(FUN = mean, units = c("mins", "secs", "hours", "days", "weeks",
  "months", "quarters", "semesters", "years"), flow_time = c("idle_time",
  "inter_start_time"), color_scale = "Reds", color_edges = "red4", ...)
```

Arguments

FUN	A summary function to be called on the process time of a specific activity, e.g. mean, median, min, max
units	The time unit in which processing time should be presented (mins, hours, days, weeks, months, quarters, semesters, years. A month is defined as 30 days. A quarter is 13 weeks. A semester is 26 weeks and a year is 365 days
flow_time	The time to depict on the flows: the inter start time is the time between the start timestamp of consecutive activity instances, the idle time is the time between the end and start time of consecutive activity instances.
color_scale	Name of color scale to be used for nodes. Defaults to Reds. See 'Rcolorbrewer::brewer.pal.info()' for all options.
color_edges	The color used for edges. Defaults to red4.
...	Additional arguments too FUN

plot.process_matrix *Process Matrix Plot*

Description

Visualize a precendence matrix. A generic plot function for precedences matrices.

Usage

```
## S3 method for class 'process_matrix'
plot(x, ...)
```

Arguments

x	Precedence matrix
...	Additional paramters

Value

A ggplot object, which can be customized further, if deemed necessary.

`precedence_matrix` *Precendence Matrix*

Description

Construct a precendence matrix, showing how activities are followed by each other.

Usage

```
precedence_matrix(eventlog, type = c("absolute", "relative",
  "relative-antecedent", "relative-consequent", "relative-case"))
```

Arguments

`eventlog`	The event log object to be used
`type`	The type of precedence matrix, which can be absolulte, relative, relative-antecedent or relative-consequent. Absolute will return a matrix with absolute frequencies, relative will return global relative frequencies for all antecedent-consequent pairs. Relative-antecedent will return relative frequencies within each antecendent, i.e. showing the relative proportion of consequents within each antecedent. Relative-consequent will do the reverse.

Examples

```
## Not run:
library(eventdataR)
data(patients)
precedence_matrix(patients)

## End(Not run)
```

`processmapR` *processmapR - Process Maps in R*

Description

This package provides several useful techniques process visualization.

process_map *Process Map*

Description

A function for creating a process map of an event log.

Usage

```
process_map(eventlog, type, sec, type_nodes, type_edges, sec_nodes, sec_edges,
  rankdir, render, fixed_edge_width, fixed_node_pos, ...)

## S3 method for class 'eventlog'
process_map(eventlog, type = frequency("absolute"),
  sec = NULL, type_nodes = type, type_edges = type, sec_nodes = sec,
  sec_edges = sec, rankdir = "LR", render = T, fixed_edge_width = F,
  fixed_node_pos = NULL, ...)
```

Arguments

eventlog	The event log object for which to create a process map
type	A process map type, which can be created with the functions frequency, performance and custom. The first type focusses on the frequency aspect of a process, while the second one focussed on processing time. The third one allows custom attributes to be used.
sec	A secondary process map type. Values are shown between brackets.
type_nodes	A process map type to be used for nodes only, which can be created with the functions frequency and performance. The first type focusses on the frequency aspect of a process, while the second one focussed on processing time.
type_edges	A process map type to be used for edges only, which can be created with the functions frequency and performance. The first type focusses on the frequency aspect of a process, while the second one focussed on processing time.
sec_nodes	A secondary process map type for nodes only.
sec_edges	A secondary process map type for edges only.
rankdir	The direction in which to layout the graph: "LR" (default),"TB", "BT", "RL", corresponding to directed graphs drawn from top to bottom, from left to right, from bottom to top, and from right to left, respectively.
render	Whether the map should be rendered immediately (default), or rather an object of type dgr_graph should be returned.
fixed_edge_width	If TRUE, don't vary the width of edges.
fixed_node_pos	When specified as a data.frame with three columns 'act', 'x', and 'y' the position of nodes is fixed. Note that his can only be used with the 'neato' layout engine.
...	Deprecated arguments

Methods (by class)

- eventlog: Process map for event log

Examples

```
## Not run:
library(eventdataR)
data(patients)
process_map(patients)

## End(Not run)
```

---

process_matrix *Create process matrix*

Description

Create process matrix

Usage

```
process_matrix(eventlog, type, ...)

## S3 method for class 'eventlog'
process_matrix(eventlog, type = frequency(), ...)
```

Arguments

eventlog	The event log object for which to create a process matrix
type	A process matrix type, which can be created with the functions frequency, performance and custom. The first type focusses on the frequency aspect of a process, while the second one focussed on processing time. The third one allows custom attributes to be used.
...	Other arguments

Methods (by class)

- eventlog: Process matrix for event log

resource_map *Resource Map*

Description

A function for creating a resource map of an event log based on handover of work.

Usage

```
resource_map(eventlog, type = frequency("absolute"), render = T, ...)
```

Arguments

eventlog	The event log object for which to create a resource map
type	A process map type, which can be created with the functions frequency and performance. The first type focusses on the frequency aspect of a process, while the second one focussed on processing time.
render	Whether the map should be rendered immediately (default), or rather an object of type dgr_graph should be returned.
...	Deprecated arguments

Examples

```
## Not run:
library(eventdataR)
data(patients)
resource_map(patients)

## End(Not run)
```

resource_matrix *Resource Matrix*

Description

Construct a resource matrix, showing how work is handed over

Usage

```
resource_matrix(eventlog, type = c("absolute", "relative",
  "relative_antecedent", "relative_consequent"))
```

Arguments

eventlog	The event log object to be used
type	The type of resource matrix, which can be absolulte, relative, relative_antecedent or relative_consequent. Absolute will return a matrix with absolute frequencies, relative will return global relative frequencies for all antecedent-consequent pairs. Relative_antecedent will return relative frequencies within each antecendent, i.e. showing the relative proportion of consequents within each antecedent. Relative_consequent will do the reverse.

Examples

```
## Not run:
library(eventdataR)
data(patients)
precedence_matrix(patients)

## End(Not run)
```

---

trace_explorer *Trace explorer*

Description

Explore traces, ordered by relative trace frequency

Usage

```
trace_explorer(eventlog, coverage = NULL, n_traces = NULL,
  type = c("frequent", "infrequent"), .abbreviate = T, show_labels = T,
  scale_fill = scale_fill_discrete(h = c(0, 360) + 15, l = 40),
  raw_data = F)

plotly_trace_explorer(eventlog, coverage = NULL, n_traces = NULL,
  type = c("frequent", "infrequent"), .abbreviate = T, show_labels = T,
  scale_fill = scale_fill_discrete(h = c(0, 360) + 15, l = 40),
  raw_data = F)
```

Arguments

eventlog	Eventlog object
coverage	The percentage coverage of the trace to explore. Default is 20% most (in)frequent
n_traces	Instead of setting coverage, you can set an exact number of traces. Should be an integer larger than 0.
type	Frequent or infrequenct traces to explore
.abbreviate	If TRUE, abbreviate activity labels

show_labels	If False, activity labels are not shown.
scale_fill	Set color scale
raw_data	Retrun raw data

Index