Package 'petrinetR'

July 3, 2018

Type Package

Title Building, Visualizing, Exporting and Replaying Petri Nets

Version 0.2.0

Date 2018-07-03

Description Functions for the construction of Petri Nets. Petri Nets can be replayed by firing enabled transitions.
Silent transitions will be hidden by the execution handler. Also includes functionalities for the visualization of Petri Nets and
export of Petri Nets to PNML (Petri Net Markup Language) files.

License GPL-3

Depends R(>= 3.0.0)

LazyData true

Imports dplyr, visNetwork, DiagrammeR, xml2, purrr

RoxygenNote 6.0.1

URL https://www.bupar.net

BugReports https://github.com/gertjanssenswillen/petrinetR/issues

NeedsCompilation no

Author Gert Janssenswillen [aut, cre]

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

Repository CRAN

Date/Publication 2018-07-03 15:50:07 UTC

R topics documented:

`create_PN` *Create Petri Net*

Description

Function to create a petri net by specificing a set of places, transitions, flows and a marking.

Usage

```
create_PN(places, transitions, flows, marking)
```

Arguments

`places`	A vector of unique places.
`transitions`	A vector of unique transitions.
`flows`	A data.frame of flows, with columns named "from" and "to".
`marking`	The names of the places to be marked.

Examples

```
create_PN("place_1",
"transition_1",
data.frame(from = "place_1",to = "transition_1"),
marking = "place_1")
```

enabled *Enabled transitions*

Description

List the enabled transitions in a marked Petri Net. Silent transitions, i.e. starting with "inv_" or "tau" are assumed to be able to fire silently, thereby possible enabling other transitions.

Usage

```
enabled(PN)
```

Arguments

PN	A Petri Net

enabled_transition *Enabled Transition*

Description

Check if a transition is currently enabled

Usage

```
enabled_transition(PN, transition)
```

Arguments

PN	A Petri Net
transition	A Transition

execute *Execute*

Description

Executes (fire) an enabled transition and returns the Petri Net with the New marking. If the transition is enabled via the firing of silent transition (i.e. starting with "inv_" of "tau"), it will fire these first. If the transition is not enabled, it will return FALSE.

Usage

```
execute(PN, transition)
```

Arguments

PN	A Petri Net
transition	The transition to be fired

flows *Flows*

Description

Extracts the flows from a Petri Net

Usage

```
flows(PN)
```

Arguments

PN	A Petri Net

is_place *Is place*

Description

Check if a place is part of a petri net

Usage

```
is_place(place, PN)
```

Arguments

place	A place
PN	A Petri Net

is_transition *Is transition*

Description

Check if a transition is part of a petri net

Usage

```
is_transition(transition, PN)
```

Arguments

transition	A transition
PN	A Petri Net

marking *Marking*

Description

Get the current marking of a Petri Net

Usage

```
marking(PN)
```

Arguments

PN	A Petri Net

n_places *Utils*

Description

Several auxilliary functions for Petri Net objects.

Usage

```
n_places(PN)

n_transitions(PN)

n_flows(PN)

n_nodes(PN)

nodes(PN)

rename_transitions(PN, .f, ...)

rename_places(PN, .f, ...)

add_places(PN, .p)

add_transitions(PN, .t)

add_flows(PN, .flows)
```

Arguments

PN	A petri net
.f	A function name to apply for renaming
...	Additional arguments
.p	A character vector of places
.t	A character vector of transitions
.flows	A data.frame with a to and from column

parse *Parse*

Description

Parses a sequence of transitions. If possible returns the Petri Net with the updated marking. Otherwise returns FALSE

Usage

```
parse(PN, trace)
```

Arguments

PN	A Petri Net
trace	A sequence of transitions, stored in a vector.

parsel *Parse (logical)*

Description

Tests whether a sequence of transitions can be fired by a Petri Net. If so returns TRUE, otherwise FALSE.

Usage

```
parsel(PN, trace)
```

Arguments

PN	A Petri Net
trace	A sequence of transitions, stored in a vector.

part_of *Part of*

Description

Check if a node is part of a petri net

Usage

```
part_of(node, PN)
```

Arguments

node	A node
PN	A Petri Net

petrinetR *petrinetR - Building, visualizing, exporting and replaying Petri Nets*

Description

Functions for the construction of Petri Nets. Petri Nets can be replayed by firing enabled transitions. Silent transitions will be hidden by the execution handler. Also includes functionalities for the visualization of Petri Nets and export of Petri Nets to PNML-files.

places *Places*

Description

Extracts the places from a Petri Net

Usage

```
places(PN)
```

Arguments

PN	A Petri Net

post_set *Postset*

Description

Get the postset of a transition or place in a Petri Net

Usage

```
post_set(PN, node)
```

Arguments

PN	A Petri Net
node	A place or transition in the petri net

pre_set *Preset*

Description

Get the preset of a transition or place in a Petri Net

Usage

```
pre_set(PN, node)
```

Arguments

PN	A Petri Net
node	A place or transition in the petri net

print.petrinet *Generic print function for petrinet*

Description

Generic print function for petrinet

Usage

```
## S3 method for class 'petrinet'
print(x, ...)
```

Arguments

x	petrinet object
...	Additional Arguments

read_PN *Read PNML*

Description

Function

Usage

```
read_PN(file)
```

Arguments

file	Path to .pnml file

render_PN *Render Petri Net*

Description

Function

Usage

```
render_PN(PN)
```

Arguments

PN	A petri net

transitions *Transitions*

Description

Extracts the transitions from a Petri Net

Usage

```
transitions(PN)
```

Arguments

PN	A Petri Net

tree_to_PN *tree_to_PN*

Description

Create of petri net from a process tree.

Usage

```
tree_to_PN(tree)
```

Arguments

tree	The process tree to be converted

visNetwork_from_PN *VisNetwork from PN*

Description

Visualize a Petri Net with an interactive network

Usage

```
visNetwork_from_PN(PN)
```

Arguments

PN	Petri Net to visualize

Index