Package ‘orbweaver’

April 28, 2025

Title Fast and Efficient Graph Data Structures

Version 0.18.2

Description Seamlessly build and manipulate graph structures, leveraging its high-performance methods for filtering, joining, and mutating data. Ensures that mutations and changes to the graph are performed in place, streamlining your workflow for optimal productivity.

License MIT + file LICENSE

URL `https://github.com/ixpantia/orbweaver-r`

BugReports `https://github.com/ixpantia/orbweaver-r/issues`

Depends R (>= 4.2.0)

Imports glue, methods, rlang

Suggests testthat (>= 3.0.0)

Config/rextendr/version 0.3.1.9001

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

SystemRequirements Cargo (Rust's package manager) >= 1.70, rustc >= 1.70

Config/Needs/website rmarkdown

NeedsCompilation yes

Author ixpantia, SRL [cph],
Andres Quintero [aut, cre],
The authors of the dependency Rust crates [ctb] (see inst/AUTHORS file for details)

Maintainer Andres Quintero <`andres@ixpantia.com`>

Repository CRAN

Date/Publication 2025-04-28 13:50:07 UTC

Contents

add_edge *Add an edge to a graph builder*

Description

Adds an edge from one node to another in a a directed graph builder.

Usage

```
add_edge(graph_builder, from, to)
```

Arguments

graph_builder	A graph builder_object
from	The from node.
to	The to node.

Value

The updated graph builder object

See Also

Other build graphs: add_path(), build_acyclic(), build_directed(), graph_builder(), populate_edges()

Examples

```
graph_builder() |>
  add_edge("A", "B")
```

---

add_path   *Add a path to a graph*

Description

Adds all of the edges that make up the given path to the graph.

Usage

```
add_path(graph_builder, path)
```

Arguments

graph_builder	A graph builder_object
path	A character vector that describes the path

Value

The updated graph builder object

See Also

Other build graphs: add_edge(), build_acyclic(), build_directed(), graph_builder(), populate_edges()

Examples

```
graph_builder() |>
  add_path(c("A", "B", "C"))
```

`build_acyclic` *Build a DirectedAcyclicGraph from a builder*

Description

Builds a graph builder into a new DirectedAcyclicGraph object.

NOTE: This will consume the builder. It will leave an empty builder in its place.

Usage

```
build_acyclic(graph_builder)
```

Arguments

`graph_builder`	A graph builder object

Value

A DirectedAcyclicGraph Object

See Also

Other build graphs: `add_edge()`, `add_path()`, `build_directed()`, `graph_builder()`, `populate_edges()`

Examples

```
graph_builder() |>
  add_path(c("1", "2", "3", "4")) |>
  build_acyclic()
```

`build_directed` *Build a DirectedGraph from a builder*

Description

Builds a graph builder into a new DirectedGraph object.

NOTE: This will consume the builder. It will leave an empty builder in its place.

Usage

```
build_directed(graph_builder)
```

Arguments

`graph_builder`	A graph builder object

Value

A DirectedGraph Object

See Also

Other build graphs: `add_edge()`, `add_path()`, `build_acyclic()`, `graph_builder()`, `populate_edges()`

Examples

```
graph_builder() |>
  add_path(c("1", "2", "3", "4")) |>
  build_directed()
```

`children` *Get the children on a node*

Description

Get a list of the node ids of the children of the provided node.

Usage

```
children(graph, nodes)
```

Arguments

`graph`	A graph object
`nodes`	A character vector of nodes to find children for

Value

A character vector

Examples

```
graph <- graph_builder() |>
  add_edge(from = "A", to = "B") |>
  build_directed()

graph |> children("A")
```

find_all_paths *Find all paths between two nodes*

Description

Find all the paths between two nodes in a graph.

Not all graphs support this function. Currently only `DirectedAcyclicGraph` supports this.

Usage

```
find_all_paths(graph, from, to)
```

Arguments

`graph`	A graph object
`from`	The starting node of the path
`to`	The ending node of the path

Value

A list of character vectors

See Also

Other analyze graphs: `find_path()`, `find_path_one_to_many()`, `get_all_leaves()`, `get_all_roots()`, `get_leaves_under()`, `get_roots_over()`, `least_common_parents()`

Examples

```
graph <- graph_builder() |>
  add_path(c("A", "B", "C")) |>
  add_path(c("A", "Z", "C")) |>
  add_path(c("A", "B", "A")) |>
  build_directed()

find_all_paths(graph, "A", "C")
```

find_path *Find a path between two nodes*

Description

Finds a path between two nodes in a graph.

Different types of graphs use different algorithms to find the paths. a `DirectedGraph` uses breadth-first search while an `DirectedAcyclicGraph` uses topological sort.

The path is represented as a character vector with the node ids of the nodes that make up the path.

Usage

```
find_path(graph, from, to)
```

Arguments

`graph`	A graph object
`from`	The starting node of the path
`to`	The ending node of the path

Value

A character vector

See Also

Other analyze graphs: `find_all_paths()`, `find_path_one_to_many()`, `get_all_leaves()`, `get_all_roots()`, `get_leaves_under()`, `get_roots_over()`, `least_common_parents()`

Examples

```
graph <- graph_builder() |>
  add_path(c("A", "B", "C")) |>
  build_directed()

find_path(graph, "A", "C")
```

find_path_one_to_many *Find the a valid path from one node to many*

Description

Find a valid path from one node to many

Usage

```
find_path_one_to_many(graph, from, to)
```

Arguments

graph	A graph object
from	The starting node of the path
to	A character vector of nodes

Value

A list of paths

See Also

Other analyze graphs: find_all_paths(), find_path(), get_all_leaves(), get_all_roots(), get_leaves_under(), get_roots_over(), least_common_parents()

Examples

```
edges <- data.frame(
  parent = c("A", "A", "B", "Z"),
  child =  c("B", "Z", "Z", "F")
)

graph <- graph_builder() |>
  populate_edges(edges, parent, child) |>
  build_acyclic()

find_path_one_to_many(graph, "A", edges$child)
```

`get_all_leaves` *Get all the leaf nodes of a graph*

Description

Retrieves the nodes in a graph that have no children

Usage

```
get_all_leaves(graph, ...)
```

Arguments

`graph`	A graph object
`...`	Unused

Value

A character vector of nodes

See Also

Other analyze graphs: `find_all_paths()`, `find_path()`, `find_path_one_to_many()`, `get_all_roots()`, `get_leaves_under()`, `get_roots_over()`, `least_common_parents()`

Examples

```
graph <- graph_builder() |>
  add_path(c("A", "B", "C")) |>
  add_path(c("A", "D", "C")) |>
  add_path(c("Z", "B", "C")) |>
  add_path(c("Z", "B", "H")) |>
  build_directed()

get_all_leaves(graph)
```

`get_all_roots` *Get the all the root nodes of a graph*

Description

Retrieves the nodes in a graph that have no parents

Usage

```
get_all_roots(graph, ...)
```

Arguments

`graph`	A graph object
`...`	Unused

Value

A character vector of nodes

See Also

Other analyze graphs: `find_all_paths()`, `find_path()`, `find_path_one_to_many()`, `get_all_leaves()`, `get_leaves_under()`, `get_roots_over()`, `least_common_parents()`

Examples

```
graph <- graph_builder() |>
  add_path(c("A", "B", "C")) |>
  add_path(c("A", "D", "C")) |>
  add_path(c("Z", "B", "C")) |>
  build_directed()

get_all_roots(graph)
```

`get_leaves_as_df` *Get leaves as a data frame*

Description

Get leaves of a set of nodes in a data frame format.

Usage

```
get_leaves_as_df(graph, nodes)
```

Arguments

`graph`	A graph object
`nodes`	A character vector of node IDs

Value

A data frame of leaves

get_leaves_under *Get the leaf nodes of a graph under some nodes*

Description

Retrieves the nodes in a graph that have no children under a certain node or group of nodes

Usage

```
get_leaves_under(graph, nodes)
```

Arguments

graph	A graph object
nodes	A character vector of nodes to find leaves for

Value

A character vector of nodes

See Also

Other analyze graphs: find_all_paths(), find_path(), find_path_one_to_many(), get_all_leaves(), get_all_roots(), get_roots_over(), least_common_parents()

Examples

```
graph <- graph_builder() |>
  add_path(c("A", "B", "C")) |>
  add_path(c("A", "D", "C")) |>
  add_path(c("Z", "B", "C")) |>
  add_path(c("Z", "B", "H")) |>
  build_directed()

get_leaves_under(graph, "D")
```

get_roots_over *Get the root nodes of a graph over some nodes*

Description

Retrieves the nodes in a graph that have no parents over a certain node or group of nodes

Usage

```
get_roots_over(graph, nodes)
```

Arguments

graph	A graph object
nodes	A character vector of nodes to find roots for

Value

A character vector of nodes

See Also

Other analyze graphs: find_all_paths(), find_path(), find_path_one_to_many(), get_all_leaves(), get_all_roots(), get_leaves_under(), least_common_parents()

Examples

```
graph <- graph_builder() |>
  add_path(c("A", "B", "C")) |>
  add_path(c("A", "D", "C")) |>
  add_path(c("Z", "B", "C")) |>
  build_directed()

get_roots_over(graph, "D")
```

graph_builder *A new builder for a graph based on the type*

Description

Object used to build graphs

Usage

```
graph_builder(type = c("directed"))
```

Arguments

type	The type of graph

Value

An object of class 'DirectedGraphBuilder'.

See Also

Other build graphs: add_edge(), add_path(), build_acyclic(), build_directed(), populate_edges()

Examples

```
graph_builder()
```

graph_from_bin *Read the graph from a binary blob*

Description

Read the graph from a binary blob

Usage

```
graph_from_bin(path, bin, type = c("directed", "dag"))
```

Arguments

`path`	(Optional) Path to a file containing a graph binary
`bin`	(Optional) The raw binary of the graph
`type`	The type of graph the JSON represents

Value

A graph object

See Also

Other graphs i/o: `graph_to_bin()`

Examples

```
bin <- graph_builder() |>
  add_edge("A", "B") |>
  build_directed() |>
  graph_to_bin()
bin

graph_from_bin(bin = bin)
```

graph_to_bin *Save the graph into a binary blob*

Description

Save the graph into a binary blob

Usage

```
graph_to_bin(graph, path)
```

Arguments

`graph`	A graph object
`path`	Path to a file to save the graph into

Value

Run for its side-effects

See Also

Other graphs i/o: `graph_from_bin()`

Examples

```
graph <- graph_builder() |>
  add_edge("A", "B") |>
  build_directed()

graph_to_bin(graph)
```

`has_children` *Checks if a node in a graph has children*

Description

This function validates if the node has an edge pointing to any other node.

Usage

```
has_children(graph, nodes)
```

Arguments

`graph`	A graph object
`nodes`	A character vector of nodes to determine

Value

A logical vector with the same length as `nodes`

Examples

```
graph <- graph_builder() |>
  add_edge(from = "A", to = "B") |>
  build_directed()
graph

graph |> has_children(nodes = "A")
graph |> has_children(nodes = "B")
```

has_parents *Checks if a node in a graph has parents*

Description

This function validates if any edge points to the given node.

Usage

```
has_parents(graph, nodes)
```

Arguments

graph	A graph object
nodes	A character vector of nodes to determine

Value

A logical vector with the same length as `nodes`

Examples

```
graph <- graph_builder() |>
  add_edge(from = "A", to = "B") |>
  build_directed()
graph

graph |> has_parents(nodes = "A")
graph |> has_parents(nodes = "B")
```

least_common_parents *Find the least common parents in a graph*

Description

It finds the nodes that have no parents in the given set.

Usage

```
least_common_parents(graph, selected)
```

Arguments

graph	A graph object
selected	A character vector of node ids

Value

A character vector of node ids

See Also

Other analyze graphs: `find_all_paths()`, `find_path()`, `find_path_one_to_many()`, `get_all_leaves()`, `get_all_roots()`, `get_leaves_under()`, `get_roots_over()`

Examples

```
graph_edges <- data.frame(
  parent = c("A", "B", "C", "C", "F"),
  child = c("B", "C", "D", "E", "D")
)

graph <- graph_builder() |>
  populate_edges(graph_edges, parent, child) |>
  build_directed()
graph

graph |> least_common_parents(c("D", "E"))
```

`nodes` *Get the nodes in the graph*

Description

Returns the unique nodes in the graph

Usage

```
nodes(graph, ...)
```

Arguments

`graph`	A directed or directed acyclic graph
`...`	Reserved for later use

Value

A character vector with the nodes

Examples

```
graph <- graph_builder() |>
  add_edge(from = "A", to = "B") |>
  build_directed()
graph

nodes(graph)
```

`parents` *Get the parents on a node*

Description

Get a list of the node ids of the parents of the provided node.

Usage

```
parents(graph, nodes)
```

Arguments

`graph`	A graph object
`nodes`	A character vector of nodes to find parents for

Value

A character vector

Examples

```
graph <- graph_builder() |>
  add_edge(from = "A", to = "B") |>
  build_directed()

graph |> parents("A")
graph |> parents("B")
```

`populate_edges` *Populates the edges of a graph from a* `data.frame`

Description

Adds a set of edges from a `data.frame` to a graph

Usage

```
populate_edges(graph_builder, edges_df, parent_col, child_col)
```

Arguments

`graph_builder`	A graph builder object
`edges_df`	A `data.frame` with a parent and child variable
`parent_col`	The name of the column containing the parents
`child_col`	The name of the column containing the children

Value

The updated graph builder object

See Also

Other build graphs: add_edge(), add_path(), build_acyclic(), build_directed(), graph_builder()

Examples

```
graph_edges <- data.frame(
  parent = c("A", "B", "C"),
  child = c("B", "C", "D")
)

graph_builder() |>
  populate_edges(
    edges_df = graph_edges,
    parent_col = "parent",
    child_col = "child"
  )
```

Index