Package ‘dosearch’

August 19, 2021

Type Package

Version 1.0.8

Title Causal Effect Identification from Multiple Incomplete Data Sources

Description Identification of causal effects from arbitrary observational and experimental probability distributions via do-calculus and standard probability manipulations using a search-based algorithm by Tikka et al. (2021) <doi:10.18637/jss.v099.i05>. Allows for the presence of mechanisms related to selection bias (Bareinboim, E. and Tian, J. (2015) <http://ftp.cs.ucla.edu/pub/stat_ser/r445.pdf>), transportability (Bareinboim, E. and Pearl, J. (2014) <http://ftp.cs.ucla.edu/pub/stat_ser/r443.pdf>), missing data (Mohan, K. and Pearl, J. and Tian., J. (2013) <http://ftp.cs.ucla.edu/pub/stat_ser/r410.pdf>) and arbitrary combinations of these. Also supports identification in the presence of context-specific independence (CSI) relations through labeled directed acyclic graphs (LDAG). For details on CSIs see Corander et al. (2019) <doi:10.1016/j.apal.2019.04.004>.

License GPL (>= 2)

Imports Rcpp (>= 0.12.19)

Suggests dagitty, DOT, igraph

LinkingTo Rcpp

SystemRequirements C++11

NeedsCompilation yes

Author Santtu Tikka [aut, cre] (<https://orcid.org/0000-0003-4039-4342>),
Antti Hyttinen [ctb] (<https://orcid.org/0000-0002-6649-3229>),
Juha Karvanen [ctb] (<https://orcid.org/0000-0001-5530-769X>)

Maintainer Santtu Tikka <santtuth@gmail.com>

Repository CRAN

Date/Publication 2021-08-19 16:40:02 UTC

R topics documented:

`dosearch-package` *Causal Effect Identification from Multiple Incomplete Data Sources*

Description

Solves causal effect identifiability problems from arbitrary observational and experimental data using a heuristic search. Allows for the presence of advanced data-generating mechanims. See Tikka et al. (2021) <doi:10.18637/jss.v099.i05> for further details.

Author(s)

Santtu Tikka, Antti Hyttinen, Juha Karvanen

References

G. Aleksandrowicz, H. Chockler, J. Y. Halpern, and A. Ivrii. The computational complexity of structure-based causality. *Journal of Artificial Intelligence Research*, 58:431–451, 2017.

J. D. Angrist, G. W. Imbens, and D. B. Rubin. Identification of causal effects using instrumental variables. *Journal of the American Statistical Association*, 91(434):444–455, 1996.

Y. Barash and N. Friedman. Context-specific Bayesian clustering for gene expression data. *Journal of Computational Biology*, 9(2):169–191, 2002.

E. Bareinboim and J. Pearl. Controlling selection bias in causal inference. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, 22:100–108, 2012a.

E. Bareinboim and J. Pearl. Causal inference by surrogate experiments: z-identifiability. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 113–120, 2012b.

E. Bareinboim and J. Pearl. A general algorithm for deciding transportability of experimental results. *Journal of Causal Inference*, 1:107–134, 2013.

E. Bareinboim and J. Pearl. Transportability from multiple environments with limited experiments: Completeness results. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, 280–288, 2014.

E. Bareinboim and J. Tian. Recovering causal effects from selection bias. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 3475–3481, 2015.

E. Bareinboim, J. Tian, and J. Pearl. Recovering from selection bias in causal and statistical inference. In *Proceedings of the 28th AAAI Conference on Neural Information Processing Systems*, 2014.

C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proceedings of the 12th International Conference on Uncertainty in Artificial Intelligence*, 115–123, 1996.

N. E. Breslow. Statistics in epidemiology: The case-control study. *Journal of the American Statistical Association*, 91(433):14–28, 1996.

C. J. Butz, A. E. dos Santos, and J. S. Oliveira. Relevant path separation: A faster method for testing independencies in Bayesian networks. In *8th International Conference on Probabilistic Graphical Models*, 74–85, 2016.

B. Chen, D. Kumor, and E. Bareinboim. Identification and model testing in linear structural equation models using auxiliary variables. In *Proceedings of the 34th International Conference on Machine Learning*, 70:757–766, 2017.

G. F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. Artificial Intelligence, 42(2):393–405, 1990.

J. Corander, A. Hyttinen, J. Kontinen, J. Pensar, and J. Vaananen. A logical approach to context-specific independence. Annals of Pure and Applied Logic, 2019.

J. Correa and E. Bareinboim. Causal effect identification by adjustment under confounding and selection biases. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, 2017.

J. Correa, J. Tian, and E. Bareinboim. Generalized adjustment under confounding and selection biases. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

D. Danks, C. Glymour, and R. E. Tillman. Integrating locally learned causal structures with overlapping variables. In *Advances in Neural Information Processing Systems*, 1665–1672, 2009.

A. P. Dawid. Influence diagrams for causal modelling and inference. *International Statistical Review*, 70(2):161–189, 2002.

D. Entner, P. Hoyer, and P. Spirtes. Data-driven covariate selection for nonparametric estimation of causal effects. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, 31:256–264, 2013.

D. Galles and J. Pearl. Testing identifiability of causal effects. In *Proceedings of the 11th Conference Annual Conference on Uncertainty in Artificial Intelligence*, 185–195, 1995.

B. Georgi, J. Schultz, and A. Schliep. Context-specific independence mixture modelling for protein families. In *European Conference on Principles of Data Mining and Knowledge Discovery*, 79–90, 2007.

S. Greenland, J. M. Robins, and J. Pearl. Confounding and collapsibility in causal inference. *Statistical Science*, 14(1):29–46, 1999.

J. Y. Halpern. Axiomatizing causal reasoning. Journal of Artificial Intelligence Research, 12:317–337, 2000.

Y. Huang and M. Valtorta. Pearl's calculus of intervention is complete. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, 217–224, 2006.

A. Hyttinen, F. Eberhardt, and P. O. Hoyer. Causal discovery of linear cyclic models from multiple experimental data sets with overlapping variables. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 387–396, 2012.

A. Hyttinen, F. Eberhardt, and M. Jarvisalo. Do-calculus when the true graph is unknown. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*, 395–404, 2015.

A. Jaber, J. Zhang, and E. Bareinboim. Causal identification under Markov equivalence. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*, 978–987, 2018.

J. Karvanen. Study design in causal models. *Scandinavian Journal of Statistics*, 42(2):361–377, 2015.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*, 2009.

S. L. Lauritzen. Causal inference from graphical models. In *Complex Stochastic Systems*, 67–107, 2000.

R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*, 1986.

M. H. Maathuis, M. Kalisch, and P. Buhlmann. Estimating high-dimensional intervention effects from observational data. *The Annals of Statistics*, 37(6A):3133–3164, 2009.

D. Malinsky and P. Spirtes. Estimating bounds on causal effects in high-dimensional and possibly confounded systems. *International Journal of Approximate Reasoning*, 88:371–384, 2017.

K. Mohan and J. Pearl. Graphical models for processing missing data. 2018. Forthcoming, https://arxiv.org/abs/1801.03583.

K. Mohan, J. Pearl, and J. Tian. Graphical models for inference with missing data. In *Advances in Neural Information Systems*, 26:1277–1285, 2013.

H. Nyman, J. Pensar, T. Koski, and J. Corander. Stratified graphical models-context-specific independence in graphical models. Bayesian Analysis, 9(4):883–908, 2014.

J. M. Pena and M. Bendtsen. Causal effect identification in acyclic directed mixed graphs and gated models. *International Journal of Approximate Reasoning*, 90:56–75, 2017.

J. Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688, 1995.

J. Pearl. *Causality: Models, Reasoning, and Inference*, 2009.

J. Pensar, H. J. Nyman, T. Koski, and J. Corander. Labeled directed acyclic graphs: a generalization of context-specific independence in directed graphical models. Data Mining and Knowledge Discovery, 29(2):503–533, 2015.

E. Perkovic, J. Textor, M. Kalisch, and M. Maathuis. A complete generalized adjustment criterion. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*, 682–691, 2015.

J. Peters, J. M. Mooij, D. Janzing, and B. Scholkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 15:2009–2053, 2014.

S. E. Shimony. Explanation, irrelevance, and statistical independence. In *Proceedings of the 9th National conference on Artificial intelligence - Volume 1*, 482–487, 1991.

I. Shpitser and J. Pearl. Identification of joint interventional distributions in recursive semi-Markovian causal models. In *Proceedings of the 21st National Conference on Artificial Intelligence – Volume 2*, 1219–1226, 2006a.

I. Shpitser and J. Pearl. Identification of conditional interventional distributions. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, 437–444, 2006b.

I. Shpitser and J. Pearl. Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research*, 9:1941–1979, 2008.

I. Shpitser, K. Mohan, and J. Pearl. Missing data as a causal and probabilistic problem. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*, 802–811, 2015.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*, 1993.

S. Tikka and J. Karvanen. Identifying causal effects with the R package causaleffect. *Journal of Statistical Software*, 76(12):1–30, 2017a.

S. Tikka and J. Karvanen. Simplifying probabilistic expressions in causal inference. *Journal of Machine Learning Research*, 18(36):1–30, 2017b.

S. Tikka and J. Karvanen. Enhancing identification of causal effects by pruning. *Journal of Machine Learning Research*, 18(194):1–23, 2018.

S. Tikka and J. Karvanen. Surrogate outcomes and transportability. *International Journal of Approximate Reasoning*, 108:21–37, 2019.

S. Tikka, A. Hyttinen and J. Karvanen. Causal effect identification from multiple incomplete data sources: a general search-based approach. *Journal of Statistical Software*, 99(5):1–40, 2021.

R. Tillman and P. Spirtes. Learning equivalence classes of acyclic models with latent and selection variables from multiple datasets with overlapping variables. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, 3–15, 2011.

S. Triantafillou and I. Tsamardinos. Constraint-based causal discovery from multiple interventions over overlapping variable sets. *Journal of Machine Learning Research*, 16:2147–2205, 2015.

S. Triantafillou, I. Tsamardinos, and I. Tollis. Learning causal structure from overlapping variable sets. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 860–867, 2010.

B. van der Zander and M. Liskiewicz. On searching for generalized instrumental variables. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, 2016.

S. Visscher, P. Lucas, I. Flesch, and K. Schurink. Using temporal context-specific independence information in the exploratory analysis of disease processes. In *Conference on Artificial Intelligence in Medicine in Europe*, 87–96, 2007.

`bivariate_missingness` *Systematic Analysis of Bivariate Missing Data Problems*

Description

This data set contains the results of a systematic analysis of all missing data problems of two variables. Each problem is associated with a graph containing two vertices, X and Y, and their response indicators, R_X and R_Y.

Usage

```
data(bivariate_missingness)
```

Format

A data frame with 6144 rows and 8 variables:

graph the graph of the instance, see `get_derivation` for more details on the syntax

nedges number of edges in the graph (directed and bidirected)

arrowXtoY whether the graph contains an arrow from X to Y or not

jointXY identifiability of the joint distribution of X and Y

marginX identifiability of the marginal distribution of X

marginY identifiability of the marginal distribution of Y

YcondX identifiability of the conditional distribution of Y given X

YdoX identifiability of the causal effect of X on Y

Source

Tikka et. al. (2019) <arXiv:1902.01073>

`dosearch` *Identify a causal effect from arbitrary experiments and observations*

Description

Identify a causal `query` from available `data` in a causal model described by a `graph` that is a semi-Markovian DAG or a labeled directed acyclic graph (LDAG). For DAGs, special mechanisms related to transportability of causal effects, recoverability from selection bias and identifiability under missing data can also be included.

Usage

```
dosearch(data, query, graph,
         transportability, selection_bias, missing_data,
         control)
```

Arguments

`data`	a character string describing the available distributions in the package syntax. Alternatively, a list of character vectors. See 'Details'.
`query`	a character string describing the target distribution in the package syntax. Alternatively, a character vector. See 'Details'.
`graph`	a character string describing either a DAG or an LDAG in the package syntax. Alternatively, an "igraph" graph as used in the "causaleffect" package or a DAG constructed using the "dagitty" package. See 'Details'.
`transportability`	a character string describing the transportability nodes of the model in the package syntax (for DAGs only). See 'Details'.
`selection_bias`	a character string describing the selection bias nodes of the model in the package syntax (for DAGs only). See 'Details'.
`missing_data`	a character string describing the missing data mechanisms of the model in the package syntax (for DAGs only). See 'Details'.
`control`	a list of control parameters. See 'Details'.

Details

`data` is used to list the available input distributions. When `graph` is a DAG the distributions should be of the form

$$P(A_i|do(B_i), C_i).$$

Individual variables within sets should be separated by a comma. For example, three input distributions

$$P(Z|do(X)), P(W, Y|do(Z, X)), P(W, Y, X|Z),$$

should be given as follows:

```
> data <- "
+  P(Z|do(X))
+  P(W,Y|do(Z,X))
+  P(W,Y,X|Z)
+"
```

The use of multiple do-operators is not permitted. Furthermore, when both conditioning variables and a do-operator are present, every conditioning variable must either precede the do-operator or follow it. When `graph` is an LDAG, the do-operation is represented by an intervention node, i.e.,

$$P(Y|do(X), Z) = P(Y|X, Z, I_X = 1)$$

For example, in the case of the previous example in an LDAG, the three input distributions become:

```
> data <- "
+  P(Z|X,I_X = 1)
+  P(W,Y|Z,X,I_X=1,I_Z=1)
+  P(W,Y,X|Z)
+"
```

The intervention nodes I_X and I_Z must be explicitly defined in the `graph` along with the relevant labels for the edges.

`query` is the target distribution of the search. It has the same syntax as `data`, but only a single distribution should be given.

`graph` is a description of a directed acyclic graph where directed edges are denoted by `->` and bidirected arcs corresponding to unobserved confounders are denoted by `<->` (or by `--`). As an example, a DAG with two directed edges and one bidirected edge is constructed as follows:

```
> graph <- "
+  X -> Z
+  Z -> Y
+  X <-> Y
+"
```

Some alternative formats for DAGs are supported as well. Graphs created using the `igraph` package in the `causal.effect` syntax can be used here. Similarly, DAGs created using `dagitty` are supported.

LDAGs are constructed similarly with the addition of labels and with the omission bidirected edges (latent variables must be explicitly defined). As an example, an LDAG with two labeled edges can be constructed as follows:

```
  > graph <- "
+  X -> Z : A = 0
+  Z -> Y : A = 1
+  A -> Z
+  A -> Y
+"
```

Here the labels indicate that the edge from X to Z vanishes when A has the value 0 and the edge from Z to Y vanishes when A has the value 1. Multiple labels on the same edge should be separated by a semi-colon.

`transportability` enumerates the nodes that should be understood as transportability nodes responsible for discrepancies between domains. Individual variables should be separated by a comma. See e.g., Bareinboim and Pearl (2014) for details on transportability.

`selection_bias` enumerates the nodes that should be understood as selection bias nodes responsible for bias in the input data sets. Individual variables should be separated by a comma. See e.g., Bareinboim and Pearl (2014) for details on selection bias recoverability.

`missing_data` enumerates the missingness mechanisms of the model. The syntax for a single mechanism is `M_X : X` where M_X is the mechanism for X. Individual mechanisms should be separated by a comma. Note that both M_X and X must be present in the graph if the corresponding mechanism is given as input. Proxy variables should not be included in the graph, since they are automatically generated based on `missing_data`. By default, a warning is issued if a proxy variable is present in an input distribution but its corresponding mechanism is not present in any input. See e.g., Mohan, Pearl and Tian (2013) for details on missing data as a causal inference problem.

The `control` argument is a list that can supply any of the following components:

`benchmark` A logical value. If `TRUE`, the search time is recorded and returned (in milliseconds). Defaults to `FALSE`.

`benchmark_rules` A logical value. If `TRUE`, the time taken by each individual inference rule is also recorded in the benchmark (in milliseconds). Defaults to `FALSE`.

`draw_derivation` A logical value. If `TRUE`, a string representing the derivation steps as a DOT graph is returned. The graph can be exported as an image for example by using the `DOT` package. Defaults to `FALSE`.

`draw_all` A logical value. If `TRUE` and if `draw_derivation = TRUE`, the derivation will contain every step taken by the search. If `FALSE`, only steps that resulted in an identifiable target are returned. Defaults to `FALSE`.

`formula` A logical value. If `TRUE`, a string representing the identifiable query is returned when the target query is identifiable. If `FALSE`, only a logical value is returned that takes the value `TRUE` for an identifiable target and `FALSE` otherwise. Defaults to `TRUE`.

`heuristic` A logical value. If `TRUE`, new distributions are expanded according to a search heuristic (see Tikka et al. (2019) for details). Otherwise, distributions are expanded in the order in which they were identified. Defaults to `FALSE`.

`md_sym` A single character describing the symbol to use for active missing data mechanisms. Defaults to `"1"`.

`time_limit` A numeric value giving a time limit for the search (in hours). Defaults to a negative value that disables the limit.

`verbose` A logical value. If `TRUE`, diagnostic information is printed to the console during the search. Defaults to `FALSE`.

`warn` A logical value. If `TRUE`, a warning is issued for possibly unintentionally misspecified but syntactically correct input distributions.

Value

An object of class `dosearch` which is a list with the following components by default. See the options of `control` for how to obtain a graphical representation of the derivation or how to benchmark the search.

`identifiable` A logical value that attains the value `TRUE` is the target quantity is identifiable and `FALSE` otherwise.

`formula` A character string describing a formula for an identifiable query or an empty character vector for an unidentifiable effect.

Author(s)

Santtu Tikka

References

S. Tikka, A. Hyttinen and J. Karvanen. Causal effect identification from multiple incomplete data sources: a general search-based approach. *Journal of Statistical Software*, 99(5):1–40, 2021.

Examples

```
## Simple back-door formula
data1 <- "P(x,y,z)"
query1 <- "P(y|do(x))"
graph1 <- "
  x -> y
  z -> x
  z -> y
"
dosearch(data1, query1, graph1)

## Simple front-door formula
data2 <- "P(x,y,z)"
query2 <- "P(y|do(x))"
graph2 <- "
  x -> z
```

```
  z -> y
  x <-> y
"
dosearch(data2, query2, graph2)

## Graph input using 'igraph' in the 'causaleffect' syntax
if (requireNamespace("igraph", quietly = TRUE)) {
  g_igraph <- igraph::graph.formula(x -+ z, z -+ y, x -+ y, y -+ x)
  g_igraph <- igraph::set.edge.attribute(g_igraph, "description", 3:4, "U")
  dosearch(data2, query2, g_igraph)
}

## Graph input with 'dagitty'
if (requireNamespace("dagitty", quietly = TRUE)) {
  g_dagitty <- dagitty::dagitty("dag{x -> z -> y; x <-> y}")
  dosearch(data2, query2, g_dagitty)
}

## Alternative distribution input style using lists and vectors:
## Each element of the list describes a single distribution
## Each element is a character vector that describes the role
## of each variable in the distribution as follows:
## For a variable V and a distribution P(A|do(B),C) we have
##   V = 0, if V is in A
##   V = 1, if V is in B
##   V = 2, if V is in C
data_alt <- list(
  c(x = 0, y = 0, z = 0) # = P(x,y,z)
)
query_alt <- c(x = 1, y = 0) # = P(y|do(x))
dosearch(data_alt, query_alt, graph2)

## Additional examples
## Not run:

## Multiple input distributions (both observational and interventional)
data3 <- "
  p(z_2,x_2|do(x_1))
  p(z_1|x_2,do(x_1,y))
  p(x_1|w_1,do(x_2))
  p(y|z_1,z_2,x_1,do(x_2))
  p(w|y,x_1,do(x_2))
"
query3 <- "p(y,x_1|w,do(x_2))"
graph3 <- "
  x_1 -> z_2
  x_1 -> z_1
  x_2 -> z_1
  x_2 -> z_2
  z_1 -> y
  z_2 -> y
  x_1 -> w
  x_2 -> w
```

```
  z_1 -> w
  z_2 -> w
"
dosearch(data3, query3, graph3)

## Selection bias
data4 <- "
  p(x,y,z_1,z_2|s)
  p(z_1,z_2)
"
query4 <- "p(y|do(x))"
graph4 <- "
  x   -> z_1
  z_1 -> z_2
  x   -> y
  y   -- z_2
  z_2 -> s
"
dosearch(data4, query4, graph4, selection_bias = "s")

## Transportability
data5 <- "
  p(x,y,z_1,z_2)
  p(x,y,z_1|s_1,s_2,do(z_2))
  p(x,y,z_2|s_3,do(z_1))
"
query5 <- "p(y|do(x))"
graph5 <- "
  z_1 -> x
  x   -> z_2
  z_2 -> y
  z_1 <-> x
  z_1 <-> z_2
  z_1 <-> y
  t_1 -> z_1
  t_2 -> z_2
  t_3 -> y
"
dosearch(data5, query5, graph5, transportability = "t_1, t_2, t_3")

## Missing data
## Proxy variables are denoted by an asterisk (*)
data6 <- "
  p(x*,y*,z*,m_x,m_y,m_z)
"
query6 <- "p(x,y,z)"
graph6 <- "
  z -> x
  x -> y
  x -> m_z
  y -> m_z
  y -> m_x
  z <-> y
```

```
"
dosearch(data6, query6, graph6, missing_data = "m_x : x, m_y : y, m_z : z")

## An LDAG
data7 <- "P(X,Y,Z)"
query7 <- "P(Y|X,I_X=1)"
graph7 <- "
  X -> Y : Z = 1
  Z -> Y
  Z -> X : I_X = 1
  I_X -> X
  H -> X : I_X = 1
  H -> Z
  Q -> Z
  Q -> Y : Z = 0
"
dosearch(data7, query7, graph7)

## A more complicated LDAG
## with multiple assignments for the edge X -> Z

data8 <- "P(X,Y,Z,A,W)"
query8 <- "P(Y|X,I_X=1)"
graph8 <- "
  I_X -> X
  I_Z -> Z
  A -> W
  Z -> Y
  A -> Z
  X -> Z : I_Z = 1; A = 1
  X -> Y : A = 0
  W -> X : I_X = 1
  W -> Y : A = 0
  A -> Y
  U -> X : I_X = 1
  U -> Y : A = 1
"
dosearch(data8, query8, graph8)

## Export the DOT diagram of the derivation as an SVG file
## to the working directory via the DOT package.
## By default, only the identifying part is plotted.
## PostScript format is also supported.
if (requireNamespace("DOT", quietly = TRUE)) {
  d <- get_derivation(data1, query1, graph1,
                      control = list(draw_derivation = TRUE))
  DOT::dot(d$derivation, "derivation.svg")
}


## End(Not run)
```

get_benchmark *Benchmark a specific run of the search*

Description

Returns the benchmarking information of an object of class "dosearch".

Usage

```
get_benchmark(x, run_again = FALSE, include_rules = FALSE)
```

Arguments

`x`	an object of class "dosearch".
`run_again`	a logical value. If `TRUE`, run the search again to obtain the benchmarking information if it was not requested in the function call that produced `x`.
`include_rules`	A logical value. If `TRUE`, also benchmark the time taken by each inference rule separately.

Value

A list with one or two elements. The first is always a numeric value of the total time taken by the search in milliseconds. The second is a numeric vector of the time taken by each inference rule (in the internal C++ implementation) of the search in milliseconds if `include_rules = TRUE`.

Author(s)

Santtu Tikka

Examples

```
data <- "P(x,y,z)"
query <- "P(y|do(x))"
graph <- "
  x -> y
  z -> x
  z -> y
"
x <- dosearch(data, query, graph, control = list(benchmark = FALSE))
get_benchmark(x, run_again = TRUE)
```

get_derivation — *Retrieve the derivation of a causal query*

Description

Returns the derivation of causal query of an object of class "dosearch".

Usage

```
get_derivation(x, run_again = FALSE, draw_all = FALSE)
```

Arguments

`x`	an object of class "dosearch".
`run_again`	a logical value. If `TRUE`, run the search again to obtain a derivation for the query if one was not requested in the function call that produced `x`.
`draw_all`	a logical value. If `TRUE`, the derivation will contain every step taken by the search. If `FALSE`, only steps that resulted in identification are returned.

Author(s)

Santtu Tikka

Examples

```
data <- "P(x,y,z)"
query <- "P(y|do(x))"
graph <- "
  x -> y
  z -> x
  z -> y
"
x <- dosearch(data, query, graph, control = list(draw_derivation = FALSE))
get_derivation(x, run_again = TRUE)
```

get_formula — *Retrieve the identifying formula of a causal query*

Description

Returns the identifying formula describing a causal query of an object of class "dosearch".

Usage

```
get_formula(x, run_again = FALSE)
```

Arguments

x	an object of class "dosearch".
run_again	a logical value. If TRUE, run the search again to obtain a formula for the query if one was not requested in the function call that produced x.

Value

A character string representing the query in terms of the input data.

Author(s)

Santtu Tikka

Examples

```
data <- "P(x,y,z)"
query <- "P(y|do(x))"
graph <- "
  x -> y
  z -> x
  z -> y
"
x <- dosearch(data, query, graph, control = list(formula = FALSE))
get_formula(x, run_again = TRUE)
```

---

is_identifiable   *Query whether the target distribution was identifiable or not*

Description

Returns the a logical value describing the identifiability of a causal query of an object of class "dosearch".

Usage

```
is_identifiable(x)
```

Arguments

x	an object of class "dosearch".

Value

A logical value. If TRUE, the target distribution is identifiable from the available inputs.

Author(s)

Santtu Tikka

Examples

```
data <- "P(x,y,z)"
query <- "P(y|do(x))"
graph <- "
  x -> y
  z -> x
  z -> y
"
x <- dosearch(data, query, graph)
is_identifiable(x)
# TRUE
```

Index