Package: causaleffect (via r-universe)

September 15, 2024

Version 1.3.16

Date 2022-10-28

Title Deriving Expressions of Joint Interventional Distributions and Transport Formulas in Causal Models

URL https://github.com/santikka/causaleffect/

Description Functions for identification and transportation of causal effects. Provides a conditional causal effect identification algorithm (IDC) by Shpitser, I. and Pearl, J. (2006) <http://ftp.cs.ucla.edu/pub/stat_ser/r329-uai.pdf>, an algorithm for transportability from multiple domains with limited experiments by Bareinboim, E. and Pearl, J. (2014) <http://ftp.cs.ucla.edu/pub/stat_ser/r443.pdf>, and a selection bias recovery algorithm by Bareinboim, E. and Tian, J. (2015) <http://ftp.cs.ucla.edu/pub/stat_ser/r445.pdf>. All of the previously mentioned algorithms are based on a causal effect identification algorithm by Tian , J. (2002) <http://ftp.cs.ucla.edu/pub/stat_ser/r309.pdf>.

License GPL (>= 2)

Imports igraph

Suggests R.rsp, XML

VignetteBuilder R.rsp

NeedsCompilation no

Author Santtu Tikka [aut, cre] (<https://orcid.org/0000-0003-4039-4342>)

Maintainer Santtu Tikka <santtuth@gmail.com>

Repository https://santikka.r-universe.dev

RemoteUrl https://github.com/santikka/causaleffect

RemoteRef HEAD

RemoteSha 78810a82c622ba64b1df4ec3b583acdb3683ca4b

Contents

`causaleffect-package` *Deriving Expressions of Joint Interventional Distributions and Transport Formulas in Causal Models*

Description

Do-calculus is concerned with estimating the interventional distribution of some action from the observed joint probability distribution of the variables in a given causal structure. All identifiable causal effects can be derived using the rules of do-calculus, but the rules themselves do not give any direct indication whether the effect in question is identifiable or not. Shpitser and Pearl (2006a) constructed an algorithm for identifying joint interventional distributions in causal models, which contain unobserved variables and induce directed acyclic graphs. A highly similar algorithm was constructed earlier by Tian (2002). The algorithm of Shpitser and Pearl (2006a) can be seen as a repeated application of the rules of do-calculus and known properties of probabilities, and it ultimately either derives an expression for the causal distribution or fails to identify the effect, in which case the effect is unidentifiable. Shpitser and Pearl (2006b) also presented a generalized algorithm for identification of conditional causal effects. causaleffect provides an implementation of this algorithm. In addition to ordinary identifiability, implementations of several other algorithms in causal inference are provided. These include algorithms for z-identifiability, transportability, z-transportability and meta-transportability of causal effects by Bareinboim and Pearl (2012, 2013a, 2013b, 2013c). Recently, many of these algorithms were combined under a single algorithm by Bareinboim and Pearl (2014), which is also provided along with an implementation of an algorithm for recovering from selection bias by Bareinboim and Tian (2015).

Graphs

Every causal model and selection diagram is depicted as an igraph graph with distinct attributes and special notation. Any bidirected edge corresponding to an unobserved variable must be denoted by using two unidirected edges with a `description` attribute of value `"U"`. Here is an example describing a simple causal model with only two vertices, X and Y, and a bidirected edge between them.

```
> g <- graph.formula(X -+ Y, Y -+ X)
> g <- set.edge.attribute(graph = g,
+ name = "description", index = 1:2, value = "U")
```

For selection diagrams, the vertices that correspond to selection variables must have a `description` attribute of value `"S"`. Here is an example of a simple selection diagram with a selection node S pointing to a non-selection variable Y. Because S precedes Y in the "-+" notation, S is given index 1 in the vertex sequence.

```
> d <- graph.formula(S -+ Y)
> d <- set.vertex.attribute(graph = d,
+ name = "description", index = 1, value = "S")
```

Author(s)

Santtu Tikka <santtuth@gmail.com>

References

Bareinboim E., Pearl J. 2012 Causal Inference by Surrogate Experiments: z-identifiability. *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 113–120.

Bareinboim E., Pearl J. 2013a A General Algorithm for Deciding Transportability of Experimental Results. *Journal of Causal Inference*, **1**, 107–134.

Bareinboim E., Pearl J. 2013b Meta-Transportability of Causal Effects: A Formal Approach. *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, 135–143.

Bareinboim E., Pearl J. 2013c Causal Transportability with Limited Experiments. *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, 95–101.

Bareinboim E., Pearl J. 2014 Transportability from Multiple Environments with Limited Experiments: Completeness Results. *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, 280–288.

Bareinboim E., Tian J. 2015 Recovering Causal Effects From Selection Bias. *In Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 3475–3481.

Pearl J. 2009 *Causality: Models, Reasoning and Inference*, New York: Cambridge University Press.

Shpitser I., Pearl J. 2006a Identification of Joint Interventional Distributions in Recursive semi-Markovian Causal Models. *Proceedings of the 21st National Conference on Artificial Intelligence*, **2**, 1219–1226.

Shpitser I., Pearl J. 2006b Identification of Conditional Interventional Distributions. *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, 427–444.

Tian J. 2002 Studies in Causal Reasoning and Learning. PhD thesis, Department of Computer Science, University of California, Los Angeles.

Tian, J., Pearl J. 2002 On Testable Implications of Causal Models with Hidden variables. *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, 519–527.

Tikka, S., Karvanen J. 2017 Identifying Causal Effects with the R Package causaleffect. Journal of Statistical Software, **76(12)**, 1–30.

Tikka, S., Karvanen J. 2017 Simplifying Probabilistic Expressions in Causal Inference. Journal of Machine Learning Research, **18(36)**, 1–30.

Tikka, S., Karvanen J. 2018 Enhancing Identification of Causal Effects by Pruning. Journal of Machine Learning Research, **18(194)**, 1–23.

`aux.effect` *Identify a causal effect using surrogate experiments*

Description

This function returns an expression for the joint distribution of the set of variables (`y`) given the intervention on the set of variables (`x`) using auxiliary experiments on a set (`z`) if the effect is identifiable. Otherwise an error is thrown describing the graphical structure that witnesses non-identifiability.

Usage

```
aux.effect(y, x, z, G, expr = TRUE, simp = TRUE,
  steps = FALSE, primes = FALSE, stop_on_nonid = TRUE)
```

Arguments

`y`	A character vector of variables of interest given the intervention.
`x`	A character vector of the variables that are acted upon.
`z`	A character vector describing the additional set available for manipulation.
`G`	An `igraph` object describing the directed acyclic graph induced by the causal model that matches the internal syntax.
`expr`	A logical value. If `TRUE`, a string is returned describing the expression in LaTeX syntax. Else, a list structure is returned which can be manually parsed by the function `get.expression`
`simp`	A logical value. If `TRUE`, a simplification procedure is applied to the resulting probability object. d-separation and the rules of do-calculus are applied repeatedly to simplify the expression.
`steps`	A logical value. If `TRUE`, returns a list where the first element corresponds to the expression of the causal effect and the second to the a list describing intermediary steps taken by the algorithm.
`primes`	A logical value. If `TRUE`, prime symbols are appended to summation variables to make them distinct from their other instantiations.
`stop_on_nonid`	A logical value. If `TRUE`, an error is produced when a non-identifiable effect is discovered. Otherwise recursion continues normally.

Value

If `steps = FALSE`, A character string or an object of class `probability` that describes the interventional distribution. Otherwise, a list as described in the arguments.

Author(s)

Santtu Tikka

References

Bareinboim E., Pearl J. 2012 Causal Inference by Surrogate Experiments: z-identifiability. *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 113–120.

See Also

`parse.graphml`, `get.expression`

Examples

```
library(igraph)

# simplify = FALSE to allow multiple edges
f <- graph.formula(W -+ Z, Z -+ X, X -+ Y, W -+ Y, # Observed edges
  W -+ Y, Y -+ W, Z -+ Y, Y -+ Z, Z -+ X, X -+ Z, simplify = FALSE)

# Here the bidirected edges are set to be unobserved in graph g
# This is denoted by giving them a description attribute with the value "U"
# The first 4 edges correspond to the observed edges, the rest are unobserved

f <- set.edge.attribute(f, "description", 5:10, "U")
aux.effect(y = "Y", x = "X", z = "Z", G = f)
```

`causal.effect` *Identify a causal effect*

Description

This function returns an expression for the joint distribution of the set of variables (`y`) given the intervention on the set of variables (`x`) conditional on (`z`) if the effect is identifiable. Otherwise an error is thrown describing the graphical structure that witnesses non-identifiability. If `steps = TRUE`, returns instead a list where the first element is the expression and the second element is a list of the intermediary steps taken by the algorithm.

Usage

```
causal.effect(y, x, z = NULL, G, expr = TRUE, simp = FALSE,
  steps = FALSE, primes = FALSE, prune = FALSE, stop_on_nonid = TRUE)
```

Arguments

y	A character vector of variables of interest given the intervention.
x	A character vector of the variables that are acted upon.
z	A character vector of the conditioning variables.
G	An igraph object describing the directed acyclic graph induced by the causal model that matches the internal syntax.
expr	A logical value. If TRUE, a string is returned describing the expression in LaTeX syntax. Else, a list structure is returned which can be manually parsed by the function get.expression.
simp	A logical value. If TRUE, a simplification procedure is applied to the resulting probability object. d-separation and the rules of do-calculus are applied repeatedly to simplify the expression.
steps	A logical value. If TRUE, returns a list where the first element corresponds to the expression of the causal effect and the second to the a list describing intermediary steps taken by the algorithm.
primes	A logical value. If TRUE, prime symbols are appended to summation variables to make them distinct from their other instantiations.
prune	A logical value. If TRUE, additional steps are taken to remove variables that are not necessary for identification.
stop_on_nonid	A logical value. If TRUE, an error is produced when a non-identifiable effect is discovered. Otherwise recursion continues normally.

Value

If steps = FALSE, A character string or an object of class probability that describes the interventional distribution. Otherwise, a list as described in the arguments.

Author(s)

Santtu Tikka

References

Shpitser I., Pearl J. 2006 Identification of Joint Interventional Distributions in Recursive semi-Markovian Causal Models. *Proceedings of the 21st National Conference on Artificial Intelligence*, **2**, 1219–1226.

Shpitser I., Pearl J. 2006 Identification of Conditional Interventional Distributions. *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, 427–444.

See Also

parse.graphml, get.expression

Examples

```
library(igraph)

# simplify = FALSE to allow multiple edges
g <- graph.formula(x -+ y, z -+ x, z -+ y , x -+ z, z -+ x, simplify = FALSE)

# Here the bidirected edge between X and Z is set to be unobserved in graph g
# This is denoted by giving them a description attribute with the value "U"
# The edges in question are the fourth and the fifth edge
g <- set.edge.attribute(graph = g, name = "description", index = c(4,5), value = "U")
causal.effect("y", "x", G = g)

# Pruning example
p <- graph.formula(x -+ z_4, z_4 -+ y, z_1 -+ x, z_2 -+ z_1,
  z_3 -+ z_2, z_3 -+ x, z_5 -+ z_1, z_5 -+ z_4, x -+ z_2, z_2 -+ x,
  z_3 -+ z_2, z_2 -+ z_3, z_2 -+ y, y -+ z_2,
  z_4 -+ y, y -+ z_4, z_5 -+ z_4, z_4 -+ z_5, simplify = FALSE)
p <- set.edge.attribute(p, "description", 9:18, "U")
causal.effect("y", "x", G = p, primes = TRUE, prune = TRUE)

# Simplification example
s <- graph.formula(x -+ y, w -+ x, w -+ z, z -+ y)
causal.effect("y", "x", G = s, simp = FALSE)
causal.effect("y", "x", G = s, simp = TRUE)
```

`generalize` *Derive a transport formula for a causal effect between a target domain and multiple source domains with limited experiments*

Description

This function returns an expression for the transport formula of a causal effect between a target domain and multiple source domains with limited experiments. The formula is returned for the interventional distribution of the set of variables (`y`) given the intervention on the set of variables (`x`). Available experiments are depicted by a list (`Z`) where the first element describes the elements available at the target and the rest at the sources. The multiple domains are given as a list (`D`) where the first element is the underlying causal diagram without selection variables, and the rest correspond to the selection diagrams. If the effect is non-transportable, an error is thrown describing the graphical structure that witnesses non-transportability. The vertices of any diagram in (`D`) that correspond to selection variables must have a description parameter of a single character "S" (shorthand for "selection").

Usage

```
generalize(y, x, Z, D, expr = TRUE, simp = FALSE,
  steps = FALSE, primes = FALSE, stop_on_nonid = TRUE)
```

Arguments

y	A character vector of variables of interest given the intervention.
x	A character vector of the variables that are acted upon.
Z	A list of character vectors describing the available interventions at each domain.
D	A list of igraph objects describing the selection diagrams in the internal syntax.
expr	A logical value. If TRUE, a string is returned describing the expression in LaTeX syntax. Else, a list structure is returned which can be manually parsed by the function get.expression
simp	A logical value. If TRUE, a simplification procedure is applied to the resulting probability object. d-separation and the rules of do-calculus are applied repeatedly to simplify the expression.
steps	A logical value. If TRUE, returns a list where the first element corresponds to the expression of the transport formula and the second to the a list describing intermediary steps taken by the algorithm.
primes	A logical value. If TRUE, prime symbols are appended to summation variables to make them distinct from their other instantiations.
stop_on_nonid	A logical value. If TRUE, an error is produced when a non-identifiable effect is discovered. Otherwise recursion continues normally.

Value

If steps = FALSE, A character string or an object of class probability that describes the transport formula. Otherwise, a list as described in the arguments.

Author(s)

Santtu Tikka

References

Bareinboim E., Pearl J. 2014 Transportability from Multiple Environments with Limited Experiments: Completeness Results. *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, 280–288.

See Also

aux.effect, causal.effect, get.expression, meta.transport, parse.graphml, recover, transport

Examples

```
library(igraph)

# Selection diagram corresponding to the target domain (no selection variables).
# We set simplify = FALSE to allow multiple edges.
d1 <-  graph.formula(Z_1 -+ X, Z_2 -+ X, X -+ Z_3, Z_3 -+ W,
 Z_3 -+ U, U -+ Y, W -+ U, Z_1 -+ Z_3, # Observed edges
 Z_1 -+ Z_2, Z_2 -+ Z_1, Z_1 -+ X, X -+ Z_1,
```

```
 Z_2 -+ Z_3, Z_3 -+ Z_2, Z_2 -+ U, U -+ Z_2,
 W -+ Y, Y -+ W, simplify = FALSE)

# Here the bidirected edges are set to be unobserved in the selection diagram d1.
# This is denoted by giving them a description attribute with the value "U".
# The first 8 edges are observed and the next 10 are unobserved.
d1 <- set.edge.attribute(d1, "description", 9:18, "U")

# We can use the causal diagram d1 to create selection diagrams
# for two source domains, a and b.
d1a <- union(d1, graph.formula(S_1 -+ Z_2, S_2 -+ Z_3, S_3 -+ W))

# The variables "S_1", "S_2", and "S_3" are selection variables.
# This is denoted by giving them a description attribute with the value "S".
# The graph already has 7 vertices, so the last three depict the new ones.
d1a <- set.vertex.attribute(d1a, "description", 8:10, "S")

# Selection diagram corresponding to the second
# source domain is constructed in a similar fashion.
d1b <- union(d1, graph.formula(S_1 -+ Z_1, S_2 -+ W, S_3 -+ U))
d1b <- set.vertex.attribute(d1b, "description", 8:10, "S")

# We combine the diagrams as a list.
d.comb <- list(d1, d1a, d1b)

# We still need the available experiments at each domain.
z <- list(c("Z_1"), c("Z_2"), c("Z_1"))
# This denotes that the variable "Z_1" is available for intervention
# in both the target domain, and the second source domain.
# The variable "Z_2" is available for intervention in the first source domain.

generalize(y = "Y", x = "X", Z = z, D = d.comb)
```

`get.expression` *Get the expression of a probability object*

Description

This function converts an object of class `probability` returned by `aux.effect`, `causal.effect`, `generalize`, `meta.transport`, `recover` or `transport` with `expr = FALSE` into a string which represents the probability distribution. Currently only LaTeX syntax is available.

Usage

```
get.expression(x, primes = FALSE)
```

Arguments

`x`	An object of class `probability` which is an internal list structure describing the interventional distribution.

primes — A logical value. If TRUE, prime symbols are appended to summation variables to make them distinct from their other instantiations.

Value

A character string that describes the resulting distribution in LaTeX syntax.

Author(s)

Santtu Tikka

See Also

aux.effect, causal.effect, generalize, meta.transport, recover, transport

Examples

```
library(igraph)

# simplify = FALSE to allow multiple edges
g <- graph.formula(X -+ Y, Z -+ X, Z -+ Y , X -+ Z, Z -+ X, simplify = FALSE)

# Here the bidirected edge between X and Z is set to be unobserved in graph g
# This is denoted by giving them a description attribute with the value "U"
# The edges in question are the fourth and the fifth edge
g <- set.edge.attribute(graph = g, name = "description", index = c(4,5), value = "U")

x <- causal.effect(y = "Y", x = "X", z = NULL, G = g, expr = FALSE)
get.expression(x, primes = FALSE)
get.expression(x, primes = TRUE)
```

---

meta.transport — *Derive a transport formula for a causal effect between a target domain and multiple source domains*

Description

This function returns an expression for the transport formula of a causal effect between a target domain and multiple source domains. The formula is returned for the interventional distribution of the set of variables (y) given the intervention on the set of variables (x). The multiple source domains are given as a list of selection diagrams (D). If the effect is non-transportable, an error is thrown describing the graphical structure that witnesses non-transportability. The vertices of any diagram in (D) that correspond to selection variables must have a description parameter of a single character "S" (shorthand for "selection").

Usage

```
meta.transport(y, x, D, expr = TRUE, simp = TRUE,
  steps = FALSE, primes = FALSE, stop_on_nonid = TRUE)
```

Arguments

y	A character vector of variables of interest given the intervention.
x	A character vector of the variables that are acted upon.
D	A list of igraph objects describing the selection diagrams in the internal syntax.
expr	A logical value. If TRUE, a string is returned describing the expression in LaTeX syntax. Else, a list structure is returned which can be manually parsed by the function get.expression
simp	A logical value. If TRUE, a simplification procedure is applied to the resulting probability object. d-separation and the rules of do-calculus are applied repeatedly to simplify the expression.
steps	A logical value. If TRUE, returns a list where the first element corresponds to the expression of the transport formula and the second to the a list describing intermediary steps taken by the algorithm.
primes	A logical value. If TRUE, prime symbols are appended to summation variables to make them distinct from their other instantiations.
stop_on_nonid	A logical value. If TRUE, an error is produced when a non-identifiable effect is discovered. Otherwise recursion continues normally.

Value

If steps = FALSE, A character string or an object of class probability that describes the transport formula. Otherwise, a list as described in the arguments.

Author(s)

Santtu Tikka

References

Bareinboim E., Pearl J. 2013b Meta-Transportability of Causal Effects: A Formal Approach. *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, 135–143.

See Also

parse.graphml, get.expression, transport, generalize

Examples

```
library(igraph)

# Selection diagram corresponding to the first source domain.
# We set simplify = FALSE to allow multiple edges.
d1 <- graph.formula(X -+ Z, W_1 -+ W_2, W_2 -+ Z,
  W_3 -+ Z, X -+ W_3, W_2 -+ X, Z -+ Y, # Observed edges
  S_1 -+ X, S_2 -+ W_2, S_3 -+ W_3, S_4 -+ Y, # Edges related to selection variables
  X -+ W_3, W_3 -+ X, X -+ W_2, W_2 -+ X, X -+ W_1,
  W_1 -+ X, W_1 -+ Z, Z -+ W_1, simplify = FALSE)
```

```
# Here the bidirected edges are set to be unobserved in the selection diagram d1.
# This is denoted by giving them a description attribute with the value "U".
# The first 7 edges are observed and the next 4 are related to the selection variables.
# The rest of the edges are unobserved.
d1 <- set.edge.attribute(d1, "description", 12:19, "U")

# The variables "S_1", "S_2", "S_3" and "S_4" are selection variables.
# This is denoted by giving them a description attribute with the value "S".
d1 <- set.vertex.attribute(d1, "description", 7:10, "S")

# Selection diagram corresponding to the second
# source domain is constructed in a similar fashion.
d2 <- graph.formula(X -+ Z, W_1 -+ W_2, W_2 -+ Z, W_3 -+ Z,
 X -+ W_3, W_2 -+ X, Z -+ Y, # Observed edges
 S_1 -+ X, S_2 -+ W_2, S_3 -+ W_1,
 S_4 -+ Y, S_5 -+ Z, # Edges related to selection variables
 X -+ W_3, W_3 -+ X, X -+ W_2, W_2 -+ X, X -+ W_1,
 W_1 -+ X, W_1 -+ Z, Z -+ W_1, simplify = FALSE)
d2 <- set.edge.attribute(d2, "description", 13:20, "U")
d2 <- set.vertex.attribute(d2, "description", 7:11, "S")

# We combine the diagrams as a list.
d.comb <- list(d1, d2)
meta.transport(y = "Y", x = "X", D = d.comb)
```

`parse.graphml` *Prepare GraphML files for internal use*

Description

This function reads GraphML files created by a graphical editor, which describe directed acyclic graphs. The R package XML is utilized to parse the contents of the files to suit the internal format used by causal inference functions. Bidirected arcs are replaced by two unobserved directed arcs, and the resulting XML file is coerced into an igraph object. This function also serves as a wrapper for files that already correspond to the internal format. Names for the nodes of the graph can be supplied or read directly from the input file.

Usage

```
parse.graphml(file, format = c("standard", "internal"),
  nodes = c(), use.names = TRUE)
```

Arguments

`file`	The connection to read from.
`format`	A character constant describing how bidirected arcs are denoted in the GraphML file. Option `standard` corresponds to bidirected arcs that are notated with a graphical parameter describing an arrow at each end of the arc or no arrows at all. Option `internal` matches the format that `standard` graphs are coerced into.

This option should be used only if all bidirected arcs in the graph are denoted by two directed arcs which have a description parameter of a single character "U" (shorthand for "unobserved"). Selection variables should have a description parameter of a single character "S".

`nodes` A character vector that describes the names of the nodes in the graph. This is ignored if `use.names` is `TRUE`.

`use.names` A logical value indicating whether the names of the nodes should be read from the file or not.

Value

An object of class igraph that describes the causal diagram. The parsed graph can now be used by other functions of the package.

Author(s)

Santtu Tikka

---

recover *Recover a causal effect from selection bias*

Description

This function attempts to recover the causal effect of the set of variables (`y`) given the intervention on the set of variables (`x`) in graph (`G`) containing a single selection variable. Otherwise an error is thrown describing the graphical structure that witnesses non-identifiability. The vertex of (`G`) that corresponds to the selection variable must have a description parameter of a single character "S" (shorthand for "selection"). If `steps = TRUE`, returns instead a list where the first element is the expression and the second element is a list of the intermediary steps taken by the algorithm.

Usage

```
recover(y, x, G, expr = TRUE, simp = TRUE,
  steps = FALSE, primes = FALSE, stop_on_nonid = TRUE)
```

Arguments

`y` A character vector of variables of interest given the intervention.

`x` A character vector of the variables that are acted upon.

`G` An `igraph` object describing a causal model with a single selection variable in the internal syntax.

`expr` A logical value. If `TRUE`, a string is returned describing the expression in LaTeX syntax. Else, a list structure is returned which can be manually parsed by the function `get.expression`

simp	A logical value. If TRUE, a simplification procedure is applied to the resulting probability object. d-separation and the rules of do-calculus are applied repeatedly to simplify the expression.
steps	A logical value. If TRUE, returns a list where the first element corresponds to the expression of the causal effect and the second to the a list describing intermediary steps taken by the algorithm.
primes	A logical value. If TRUE, prime symbols are appended to summation variables to make them distinct from their other instantiations.
stop_on_nonid	A logical value. If TRUE, an error is produced when a non-identifiable effect is discovered. Otherwise recursion continues normally.

Value

If steps = FALSE, A character string or an object of class probability that describes the interventional distribution. Otherwise, a list as described in the arguments.

Author(s)

Santtu Tikka

References

Bareinboim E., Tian J. 2015 Recovering Causal Effects From Selection Bias. *In Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 3475–3481.

See Also

parse.graphml, get.expression, generalize, meta.transport

Examples

```
library(igraph)

# We set simplify = FALSE to allow multiple edges.
g <- graph.formula(W_1 -+ X, W_2 -+ X, X -+ Y, # Observed edges
  W_2 -+ S, # The selection variable S
  W_1 -+ W_2, W_2 -+ W_1, W_1 -+ Y, Y -+ W_1, simplify = FALSE)

# Here the bidirected edges are set to be unobserved in the selection diagram d.
# This is denoted by giving them a description attribute with the value "U".
# The first five edges are observed, the rest are unobserved.
g <- set.edge.attribute(g, "description", 5:8, "U")

# The variable "S" is a selection variable. This is denoted by giving it
# a description attribute with the value "S".
g <- set.vertex.attribute(g, "description", 5, "S")

recover(y = "Y", x = "X", G = g)
```

surrogate.outcome *Derive a formula for a causal effect using surrogate outcomes*

Description

This function returns an expression for the causal effect of interest using surrogate outcomes. The formula is returned for the interventional distribution of the set of variables (y) given the intervention on the set of variables (x). Available experimental data are depicted by a list (S) where each element is a list with two elements, Z and W, that are character vectors describing the experiments and the outcome variables, respectively.

Usage

```
surrogate.outcome(y, x, S, G, expr = TRUE,
  steps = FALSE, primes = FALSE, stop_on_nonid = TRUE)
```

Arguments

y	A character vector of variables of interest given the intervention.
x	A character vector of the variables that are acted upon.
S	A list describing the available experimental data.
G	An igraph object describing the directed acyclic graph induced by the causal model that matches the internal syntax.
expr	A logical value. If TRUE, a string is returned describing the expression in LaTeX syntax. Else, a list structure is returned which can be manually parsed by the function get.expression
steps	A logical value. If TRUE, returns a list where the first element corresponds to the expression of the transport formula and the second to the a list describing intermediary steps taken by the algorithm.
primes	A logical value. If TRUE, prime symbols are appended to summation variables to make them distinct from their other instantiations.
stop_on_nonid	A logical value. If TRUE, an error is produced when a non-identifiable effect is discovered. Otherwise recursion continues normally.

Value

If steps = FALSE, A character string or an object of class probability that describes the causal effect. Otherwise, a list as described in the arguments.

Author(s)

Santtu Tikka

References

Bareinboim E., Pearl J. 2014 Transportability from Multiple Environments with Limited Experiments: Completeness Results. *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, 280–288.

See Also

generalize, causal.effect, get.expression

Examples

```
library(igraph)

# We set simplify = FALSE to allow multiple edges.
g <-  graph.formula(W -+ X, W -+ Z, X -+ Z, Z -+ Y, # Observed edges
  X -+ Z, Z -+ X, simplify = FALSE)

# We set the bidirected edges
g <- set.edge.attribute(g, "description", 5:6, "U")

# We construct the set of available experimental data
s <- list(
  list(Z = c("X"), W = c("Z"))
)

surrogate.outcome(y = "Y", x = "X", S = s, G = g)
```

---

transport *Derive a transport formula for a causal effect between two domains*

Description

This function returns an expression for the transport formula of a causal effect between two domains. The formula is returned for the interventional distribution of the set of variables (y) given the intervention on the set of variables (x) in a selection diagram (D). If the effect is non-transportable, an error is thrown describing the graphical structure that witnesses non-transportability. The vertices of (D) that correspond to selection variables must have a description parameter of a single character "S" (shorthand for "selection"). By default, every variable is available for intervention in the source. If only a subset of the variables is available, then the set (z) can be used to derive specific z-transportability.

Usage

```
transport(y, x, z = NULL, D, expr = TRUE, simp = TRUE,
  steps = FALSE, primes = FALSE, stop_on_nonid = TRUE)
```

Arguments

y	A character vector of variables of interest given the intervention.
x	A character vector of the variables that are acted upon.
z	A character vector of variables available for intervention. NULL value corresponds to ordinary transportability.
D	An igraph object describing a selection diagram in the internal syntax.
expr	A logical value. If TRUE, a string is returned describing the expression in LaTeX syntax. Else, a list structure is returned which can be manually parsed by the function get.expression
simp	A logical value. If TRUE, a simplification procedure is applied to the resulting probability object. d-separation and the rules of do-calculus are applied repeatedly to simplify the expression.
steps	A logical value. If TRUE, returns a list where the first element corresponds to the expression of the causal effect and the second to the a list describing intermediary steps taken by the algorithm.
primes	A logical value. If TRUE, prime symbols are appended to summation variables to make them distinct from their other instantiations.
stop_on_nonid	A logical value. If TRUE, an error is produced when a non-identifiable effect is discovered. Otherwise recursion continues normally.

Value

If steps = FALSE, A character string or an object of class probability that describes the transport formula. Otherwise, a list as described in the arguments.

Author(s)

Santtu Tikka

References

Bareinboim E., Pearl J. 2013a A General Algorithm for Deciding Transportability of Experimental Results. *Journal of Causal Inference*, **1**, 107–134.

Bareinboim E., Pearl J. 2013c Causal Transportability with Limited Experiments. *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, 95–101.

See Also

parse.graphml, get.expression, generalize, meta.transport

Examples

```
library(igraph)

# We set simplify = FALSE to allow multiple edges.
d <- graph.formula(X -+ Z, Z -+ W, W -+ V, V -+ Y, S -+ V, # Observed edges
  X -+ Z, Z -+ X, V -+ Y, Y -+ V, X -+ Y, Y -+ X, simplify = FALSE)
```

```
# Here the bidirected edges are set to be unobserved in the selection diagram d.
# This is denoted by giving them a description attribute with the value "U".
# The first five edges are observed, the rest are unobserved.
d <- set.edge.attribute(d, "description", 6:11, "U")

# The variable "S" is a selection variable. This is denoted by giving it
# a description attribute with the value "S".
d <- set.vertex.attribute(d, "description", 6, "S")

transport(y = "Y", x = "X", D = d)
```

`verma.constraints` *Find Verma constraints for a given graph*

Description

This functions computes functional constraints known as Verma constraints for a joint distribution of a given semi-Markovian causal model.

Usage

```
verma.constraints(G)
```

Arguments

`G`	An `igraph` object describing the directed acyclic graph induced by the causal model that matches the internal syntax.

Value

A list of lists, each with five components corresponding to the functional constraint. The two equal c-factors that imply the functional independence are described by `lhs.cfactor` and `rhs.cfactor` and their expressions are given by `lhs.expr` and `rhs.expr` respectively. The independent variables are given by `vars`.

Author(s)

Santtu Tikka

References

Tian, J., Pearl J. 2002 On Testable Implications of Causal Models with Hidden variables. *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, 519–527.

Index