Arithmetic Circuits of the Noisy-Or Models

Jirka Vomlel and Petr Savický

Academy of Sciences of the Czech Republic

PGM'08, Hirtshals, Denmark

Contents

- BN2O models.

Contents

- BN2O models.
- Methods exploiting the local structure of noisy-or models.

Contents

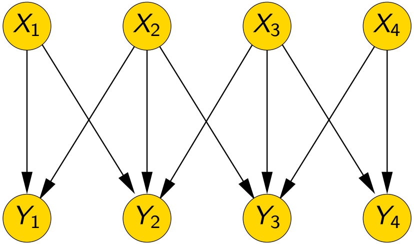
- BN2O models.
- Methods exploiting the local structure of noisy-or models.
- Arithmetic circuits - a measure of inference complexity.

Contents

- BN2O models.
- Methods exploiting the local structure of noisy-or models.
- Arithmetic circuits - a measure of inference complexity.
- Results of experiments.

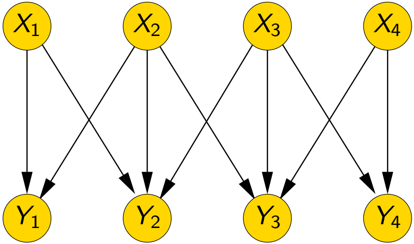
BN2O model

- Bipartite graph with two levels of Boolean variables: $X_j, j = 1, \ldots, x$ and $Y_i, i = 1, \ldots, y$.



BN2O model

- Bipartite graph with two levels of Boolean variables: $X_j, j = 1, \ldots, x$ and $Y_i, i = 1, \ldots, y$.

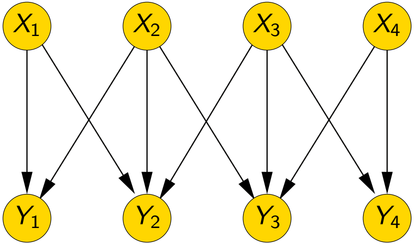


- CPT of Y_i is a noisy-or gate:

$$P(Y_i = 0 | X_{Pa(i)} = x_{Pa(i)}) \quad = \quad \prod_{j \in Pa(i)}^{n} (p_{i,j})^{x_j} \quad ,$$

BN2O model

- Bipartite graph with two levels of Boolean variables: $X_j, j = 1, \ldots, x$ and $Y_i, i = 1, \ldots, y$.



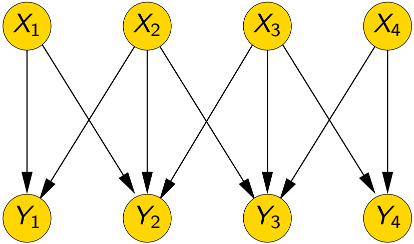
- CPT of Y_i is a noisy-or gate:

$$P(Y_i = 0 | X_{Pa(i)} = x_{Pa(i)}) \quad = \quad \prod_{j \in Pa(i)}^{n} (p_{i,j})^{x_j} \quad ,$$

- where $p_{i,j}$ is the inhibition probability for the parent X_j of Y_i.

BN2O model

- Bipartite graph with two levels of Boolean variables: $X_j, j = 1, \ldots, x$ and $Y_i, i = 1, \ldots, y$.



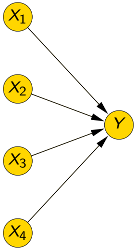
- CPT of Y_i is a noisy-or gate:

$$P(Y_i = 0 | X_{Pa(i)} = x_{Pa(i)}) \quad = \quad \prod_{j \in Pa(i)}^{n} (p_{i,j})^{x_j} \quad ,$$

- where $p_{i,j}$ is the inhibition probability for the parent X_j of Y_i.
- Y_i is false only if all its parents with value true are inhibited.

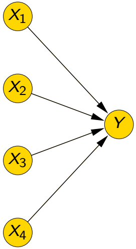
Compilation of a noisy-or gate - the standard BN approach

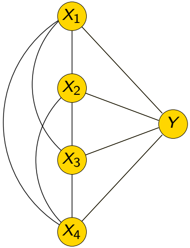
Lauritzen and Spiegelhalter (1988), Jensen et al. (1990), Shafer and Shenoy (1990)



Compilation of a noisy-or gate - the standard BN approach

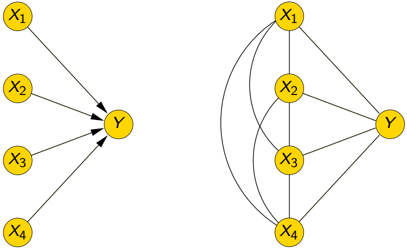
Lauritzen and Spiegelhalter (1988), Jensen et al. (1990), Shafer and Shenoy (1990)





Compilation of a noisy-or gate - the standard BN approach

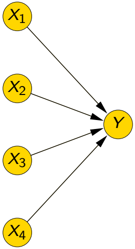
Lauritzen and Spiegelhalter (1988), Jensen et al. (1990), Shafer and Shenoy (1990)



The total table size is $2^5 = 32$.

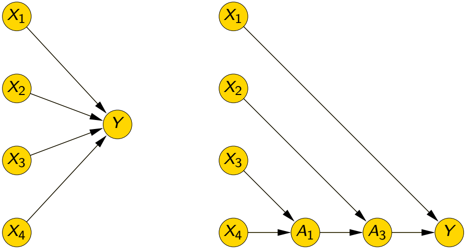
Compilation of a noisy-or gate - parent divorcing

Olesen et al. (1989)



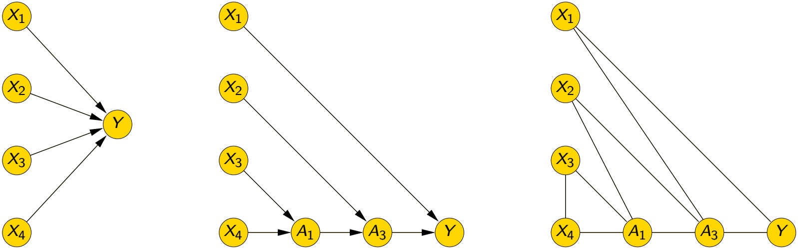
Compilation of a noisy-or gate - parent divorcing

Olesen et al. (1989)



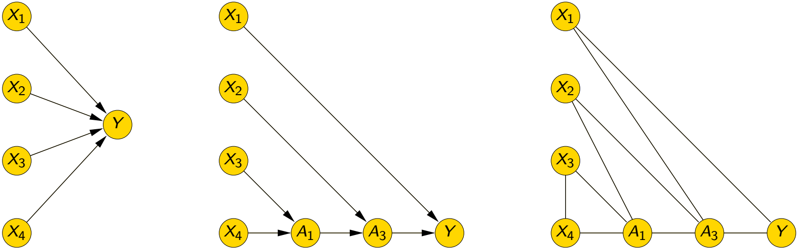
Compilation of a noisy-or gate - parent divorcing

Olesen et al. (1989)



Compilation of a noisy-or gate - parent divorcing

Olesen et al. (1989)



The total table size is $3 \cdot 2^3 = 24$.

Rank-one decomposition

Díez and Galán (2003), Vomlel (2002), Savický and Vomlel (2007)

$$P(Y_i = y_i | X_{Pa(i)} = x_{Pa(i)}) \;\; = \;\; (1 - 2y_i) \prod_{j \in Pa(i)} p_{i,j}^{x_j} + y_i \prod_{i=1}^{n} 1$$

Rank-one decomposition

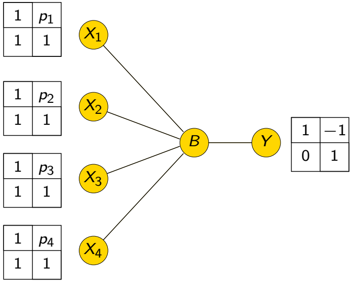
Díez and Galán (2003), Vomlel (2002), Savický and Vomlel (2007)

$$
\begin{aligned}
P(Y_i = y_i | X_{Pa(i)} = x_{Pa(i)}) &= (1 - 2y_i) \prod_{j \in Pa(i)} p_{i,j}^{x_j} + y_i \prod_{i=1}^{n} 1 \\
&= \sum_{b_i=0}^{1} \xi(b_i, y_i) \cdot \prod_{j \in Pa(i)} \varphi_{i,j}(b_i, x_j)
\end{aligned}
$$

Rank-one decomposition

Díez and Galán (2003), Vomlel (2002), Savický and Vomlel (2007)

$$
\begin{aligned}
P(Y_i = y_i | X_{Pa(i)} = x_{Pa(i)}) &= (1 - 2y_i) \prod_{j \in Pa(i)} p_{i,j}^{x_j} + y_i \prod_{i=1}^{n} 1 \\
&= \sum_{b_i=0}^{1} \xi(b_i, y_i) \cdot \prod_{j \in Pa(i)} \varphi_{i,j}(b_i, x_j)
\end{aligned}
$$



Correspondence to tensor rank-one decomposition

Savický and Vomlel (2007)

A decomposition of a conditional probability table $P(Y|X_1, \ldots, X_n)$ using the auxiliary variable B

$$P(Y_i|X_{Pa(i)}) \;=\; \sum_B \xi(B, Y_i) \cdot \prod_{j \in Pa(i)} \varphi_{i,j}(B, X_j)$$

that has the minimal number of states of B

Correspondence to tensor rank-one decomposition

Savický and Vomlel (2007)

A decomposition of a conditional probability table $P(Y|X_1, \ldots, X_n)$ using the auxiliary variable B

$$P(Y_i|X_{Pa(i)}) \;\; = \;\; \sum_B \xi(B, Y_i) \cdot \prod_{j \in Pa(i)} \varphi_{i,j}(B, X_j)$$

that has the minimal number of states of B

is in fact a (minimal) tensor rank-one decomposition of tensor $P(Y_i|X_{Pa(i)})$.

Correspondence to tensor rank-one decomposition

Savický and Vomlel (2007)

A decomposition of a conditional probability table $P(Y|X_1, \ldots, X_n)$ using the auxiliary variable B

$$P(Y_i|X_{Pa(i)}) = \sum_B \xi(B, Y_i) \cdot \prod_{j \in Pa(i)} \varphi_{i,j}(B, X_j)$$

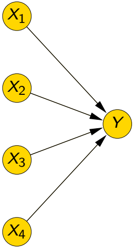
that has the minimal number of states of B

is in fact a (minimal) tensor rank-one decomposition of tensor $P(Y_i|X_{Pa(i)})$.

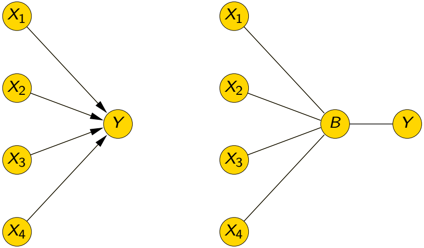
Definition (Tensor of rank one)

A tensor has rank one if it is the outer product of vectors.

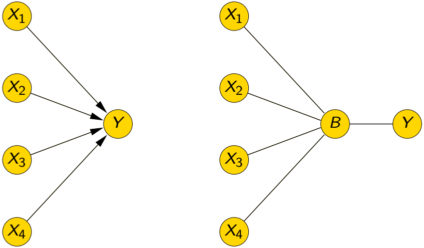
Compilation of a noisy-or gate - rank-one decomposition



Compilation of a noisy-or gate - rank-one decomposition

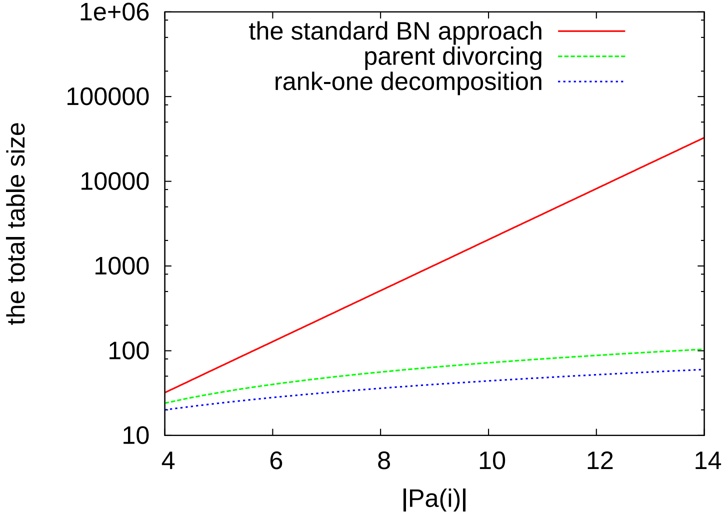


Compilation of a noisy-or gate - rank-one decomposition



The total table size is $5 \cdot 2^2 = 20$.

Comparisons for the noisy-or gate



Arithmetic circuits

Definition (Arithmetic circuit (AC))

An AC is a rooted, directed acyclic graph whose leaf nodes correspond to its inputs and whose other nodes are labeled with multiplication and addition operations. The root node corresponds to the output of the AC.

Arithmetic circuits

Definition (Arithmetic circuit (AC))

An AC is a rooted, directed acyclic graph whose leaf nodes correspond to its inputs and whose other nodes are labeled with multiplication and addition operations. The root node corresponds to the output of the AC.

Circuit inputs:

Arithmetic circuits

Definition (Arithmetic circuit (AC))

An AC is a rooted, directed acyclic graph whose leaf nodes correspond to its inputs and whose other nodes are labeled with multiplication and addition operations. The root node corresponds to the output of the AC.

Circuit inputs:

- *BN parameters*

$$\theta_{x|\mathbf{u}} \quad = \quad P(X = x|\mathbf{U} = \mathbf{u})$$

Arithmetic circuits

Definition (Arithmetic circuit (AC))

An AC is a rooted, directed acyclic graph whose leaf nodes correspond to its inputs and whose other nodes are labeled with multiplication and addition operations. The root node corresponds to the output of the AC.

Circuit inputs:

- *BN parameters*

$$\theta_{x|\mathbf{u}} = P(X = x|\mathbf{U} = \mathbf{u})$$

- *evidence indicators*

$$\lambda_x = \begin{cases} 1 & \text{if state } x \text{ of } X \text{ is consistent with evidence } \mathbf{e} \\ 0 & \text{otherwise.} \end{cases}$$

If there is no evidence for X, then $\lambda_x = 1$ for all states x of X.

Arithmetic circuits

Definition (Arithmetic circuit (AC))

An AC is a rooted, directed acyclic graph whose leaf nodes correspond to its inputs and whose other nodes are labeled with multiplication and addition operations. The root node corresponds to the output of the AC.

Circuit inputs:

- *BN parameters*

$$\theta_{x|\mathbf{u}} \quad = \quad P(X = x|\mathbf{U} = \mathbf{u})$$

- *evidence indicators*

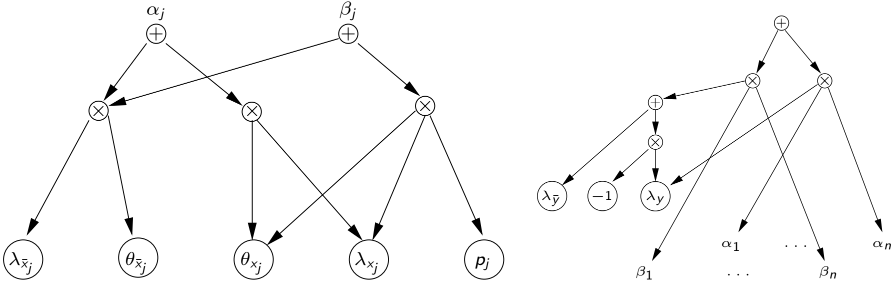
$$\lambda_x \quad = \quad \begin{cases} 1 & \text{if state } x \text{ of } X \text{ is consistent with evidence } \mathbf{e} \\ 0 & \text{otherwise.} \end{cases}$$

If there is no evidence for X, then $\lambda_x = 1$ for all states x of X.

Circuit output:

- probability of evidence $P(\mathbf{e})$.

AC of a noisy-or gate



Arithmetic circuits (ACs) - Part I

- After an upward pass through the AC we get $P(\mathbf{e})$ in the root node.

Arithmetic circuits (ACs) - Part I

- After an upward pass through the AC we get $P(\mathbf{e})$ in the root node.
- When it is followed by a downward pass through the AC we get $P(X, \mathbf{e})$ for all BN variables X.

Arithmetic circuits (ACs) - Part I

- After an upward pass through the AC we get $P(\mathbf{e})$ in the root node.
- When it is followed by a downward pass through the AC we get $P(X, \mathbf{e})$ for all BN variables X.
- An AC may be used to represent the computations in a junction tree.

Arithmetic circuits (ACs) - Part I

- After an upward pass through the AC we get $P(\mathbf{e})$ in the root node.
- When it is followed by a downward pass through the AC we get $P(X, \mathbf{e})$ for all BN variables X.
- An AC may be used to represent the computations in a junction tree.
- An AC may also represent more efficient computations due to specific properties of the initial BN (e.g., determinism, context specific independence).

Arithmetic circuits (ACs) - Part I

- After an upward pass through the AC we get $P(\mathbf{e})$ in the root node.
- When it is followed by a downward pass through the AC we get $P(X, \mathbf{e})$ for all BN variables X.
- An AC may be used to represent the computations in a junction tree.
- An AC may also represent more efficient computations due to specific properties of the initial BN (e.g., determinism, context specific independence).
- The size of an AC (i.e. number of its edges) can be used as a measure of inference complexity

Arithmetic circuits (ACs) - Part II

- Darwiche et al. proposed two different methods for constructing ACs of BNs - c2d and tabular - both are implemented in a BN compiler Ace (by Chavira and Darwiche).

Arithmetic circuits (ACs) - Part II

- Darwiche et al. proposed two different methods for constructing ACs of BNs - c2d and tabular - both are implemented in a BN compiler Ace (by Chavira and Darwiche).
- Ace uses the parent divorcing method for preprocessing noisy-or models.

Arithmetic circuits (ACs) - Part II

- Darwiche et al. proposed two different methods for constructing ACs of BNs - c2d and tabular - both are implemented in a BN compiler Ace (by Chavira and Darwiche).
- Ace uses the parent divorcing method for preprocessing noisy-or models.
- We use the size of ACs to compare the effect of preprocessing Bayesian networks by Ace's parent divorcing giving (what we call) the original model and by rank-one decomposition giving the transformed model.

Experiments

Experiments were performed by Ace running on
`aligator.utia.cas.cz`: 8x AMD Opteron 8220, 64GB RAM
but the maximum possible memory for 32 bit Ace is 3.6 GB RAM.

Experiments

Experiments were performed by Ace running on
`aligator.utia.cas.cz`: 8x AMD Opteron 8220, 64GB RAM
but the maximum possible memory for 32 bit Ace is 3.6 GB RAM.

We carried out experiments with BN2O models of various sizes:

Experiments

Experiments were performed by Ace running on
`aligator.utia.cas.cz`: 8x AMD Opteron 8220, 64GB RAM
but the maximum possible memory for 32 bit Ace is 3.6 GB RAM.

We carried out experiments with BN2O models of various sizes:

- `x` is the number of nodes in the top level,

Experiments

Experiments were performed by Ace running on
`aligator.utia.cas.cz`: 8x AMD Opteron 8220, 64GB RAM
but the maximum possible memory for 32 bit Ace is 3.6 GB RAM.

We carried out experiments with BN2O models of various sizes:

- `x` is the number of nodes in the top level,
- `y` is the number of nodes in the bottom level,

Experiments

Experiments were performed by Ace running on
`aligator.utia.cas.cz`: 8x AMD Opteron 8220, 64GB RAM
but the maximum possible memory for 32 bit Ace is 3.6 GB RAM.

We carried out experiments with BN2O models of various sizes:

- `x` is the number of nodes in the top level,
- `y` is the number of nodes in the bottom level,
- `e` is the total number of edges in the BN2O model, and `e/y` defines the number of parents for each node from the bottom level.

Experiments

Experiments were performed by Ace running on
`aligator.utia.cas.cz`: 8x AMD Opteron 8220, 64GB RAM
but the maximum possible memory for 32 bit Ace is 3.6 GB RAM.

We carried out experiments with BN2O models of various sizes:

- x is the number of nodes in the top level,
- y is the number of nodes in the bottom level,
- e is the total number of edges in the BN2O model, and e/y defines the number of parents for each node from the bottom level.

For each x-y-e type ($x, y = 10, 20, 30, 40, 50$ and $e/y = 2, 5, 10, 20$, excluding those with $e/y > x$) we generated randomly ten models.

Experiments

Experiments were performed by Ace running on
`aligator.utia.cas.cz`: 8x AMD Opteron 8220, 64GB RAM
but the maximum possible memory for 32 bit Ace is 3.6 GB RAM.

We carried out experiments with BN2O models of various sizes:

- x is the number of nodes in the top level,
- y is the number of nodes in the bottom level,
- e is the total number of edges in the BN2O model, and e/y defines the number of parents for each node from the bottom level.

For each x-y-e type ($x, y = 10, 20, 30, 40, 50$ and $e/y = 2, 5, 10, 20$, excluding those with $e/y > x$) we generated randomly ten models.
For every node from the bottom level we randomly selected e/y nodes from the top level as its parents.

Experiments

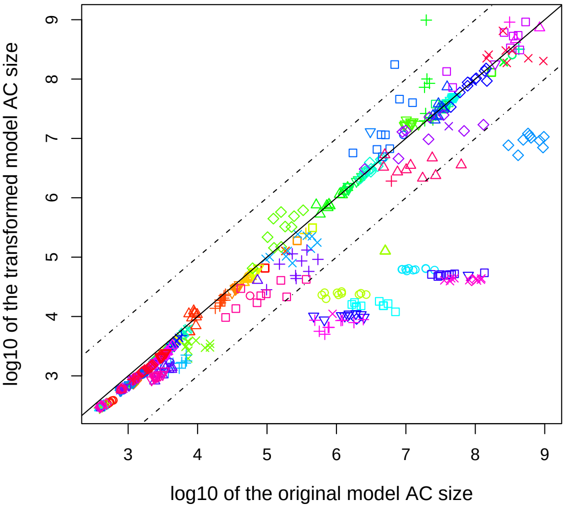
Experiments were performed by Ace running on
`aligator.utia.cas.cz`: 8x AMD Opteron 8220, 64GB RAM
but the maximum possible memory for 32 bit Ace is 3.6 GB RAM.

We carried out experiments with BN2O models of various sizes:

- $\mathtt{x}$ is the number of nodes in the top level,
- $\mathtt{y}$ is the number of nodes in the bottom level,
- $\mathtt{e}$ is the total number of edges in the BN2O model, and $\mathtt{e}/\mathtt{y}$ defines the number of parents for each node from the bottom level.

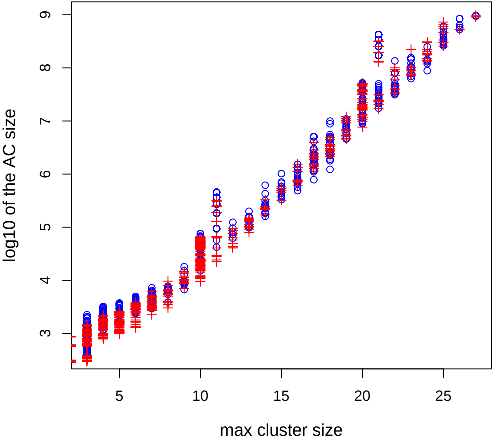
For each $\mathtt{x}$-$\mathtt{y}$-$\mathtt{e}$ type ($\mathtt{x}, \mathtt{y} = 10, 20, 30, 40, 50$ and $\mathtt{e}/\mathtt{y} = 2, 5, 10, 20$, excluding those with $\mathtt{e}/\mathtt{y} > \mathtt{x}$) we generated randomly ten models.
For every node from the bottom level we randomly selected $\mathtt{e}/\mathtt{y}$ nodes from the top level as its parents.
All models and results are available at:
`http://www.utia.cz/vomlel/ac/`

Transformed vs. original model AC size



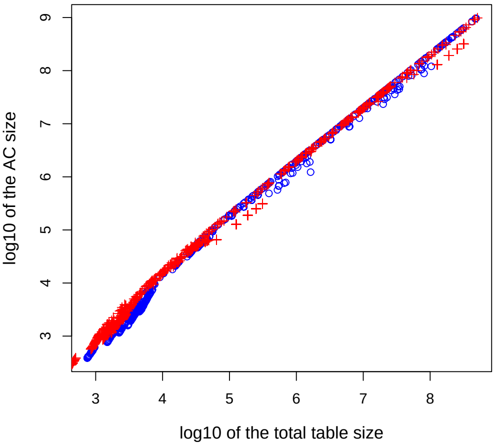
Dependence of the AC size on the size of a largest clique

for the original and the transformed models

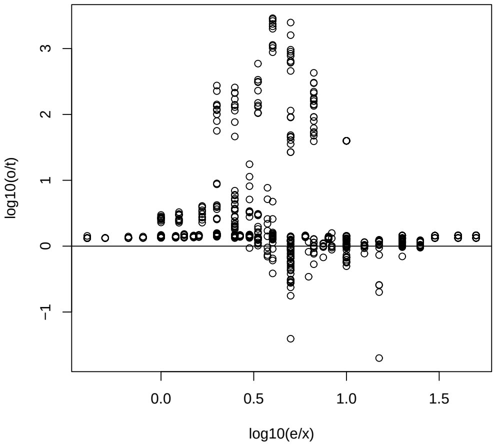


Dependence of the AC size on the total table size

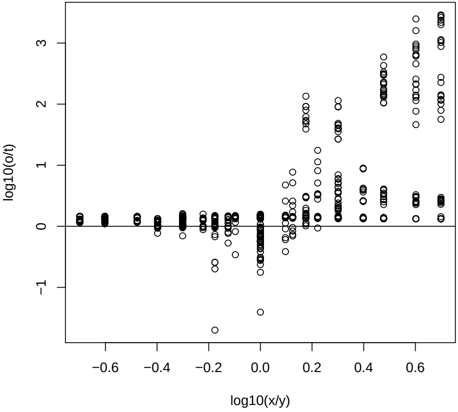
for the original and the transformed models



Dependence of the AC size reduction on the relative number of edges



Dependence of the AC size reduction on the ratio of the number of nodes in the first and the second levels



Summary of the experiments

- The AC of the transformed model was smaller in 88% of the BN2O models solved by Ace for both - the original and the transformed models.

Summary of the experiments

- The AC of the transformed model was smaller in 88% of the BN2O models solved by Ace for both - the original and the transformed models.
- In several cases we got significant reductions in the AC size - in a few cases multiple order of magnitude.

Summary of the experiments

- The AC of the transformed model was smaller in 88% of the BN2O models solved by Ace for both - the original and the transformed models.
- In several cases we got significant reductions in the AC size - in a few cases multiple order of magnitude.
- There are also eleven cases where the AC of the transformed model is at least three times larger - we will comment on these cases on the next slide.

Summary of the experiments

- The AC of the transformed model was smaller in 88% of the BN2O models solved by Ace for both - the original and the transformed models.
- In several cases we got significant reductions in the AC size - in a few cases multiple order of magnitude.
- There are also eleven cases where the AC of the transformed model is at least three times larger - we will comment on these cases on the next slide.
- For some of larger models Ace ran out of memory – for 26% of the original models and 21% of the transformed models.

Summary of the experiments

- The AC of the transformed model was smaller in 88% of the BN2O models solved by Ace for both - the original and the transformed models.
- In several cases we got significant reductions in the AC size - in a few cases multiple order of magnitude.
- There are also eleven cases where the AC of the transformed model is at least three times larger - we will comment on these cases on the next slide.
- For some of larger models Ace ran out of memory – for 26% of the original models and 21% of the transformed models.
- For 85% of the tested BN2O models the tabular method led to smaller ACs than c2d.

Summary of the experiments

- The AC of the transformed model was smaller in 88% of the BN2O models solved by Ace for both - the original and the transformed models.
- In several cases we got significant reductions in the AC size - in a few cases multiple order of magnitude.
- There are also eleven cases where the AC of the transformed model is at least three times larger - we will comment on these cases on the next slide.
- For some of larger models Ace ran out of memory – for 26% of the original models and 21% of the transformed models.
- For 85% of the tested BN2O models the tabular method led to smaller ACs than c2d.
- The AC size depends on the total table size in the resulting model.

Comments to the eleven cases with a significant loss

- The transformed model is a graph minor of the original model.

Comments to the eleven cases with a significant loss

- The transformed model is a graph minor of the original model.
- Hence, the treewidth of the transformed model is never larger than the treewidth of the original model.

Comments to the eleven cases with a significant loss

- The transformed model is a graph minor of the original model.
- Hence, the treewidth of the transformed model is never larger than the treewidth of the original model.
- However, if a heuristic method is used for triangulation then it may happen that we get larger treewidth for the triangulated graph of the transformed model.

Comments to the eleven cases with a significant loss

- The transformed model is a graph minor of the original model.
- Hence, the treewidth of the transformed model is never larger than the treewidth of the original model.
- However, if a heuristic method is used for triangulation then it may happen that we get larger treewidth for the triangulated graph of the transformed model.
- We believe this is the main reason behind the few large losses of the transformed model.

Comments to the eleven cases with a significant loss

- The transformed model is a graph minor of the original model.
- Hence, the treewidth of the transformed model is never larger than the treewidth of the original model.
- However, if a heuristic method is used for triangulation then it may happen that we get larger treewidth for the triangulated graph of the transformed model.
- We believe this is the main reason behind the few large losses of the transformed model.
- We conducted additional experiments with all eleven models with significant loss in the AC size.

Comments to the eleven cases with a significant loss

- The transformed model is a graph minor of the original model.
- Hence, the treewidth of the transformed model is never larger than the treewidth of the original model.
- However, if a heuristic method is used for triangulation then it may happen that we get larger treewidth for the triangulated graph of the transformed model.
- We believe this is the main reason behind the few large losses of the transformed model.
- We conducted additional experiments with all eleven models with significant loss in the AC size.
- In all of these cases we were able to reduce the deterioration factor to less than three using a better triangulation method provided by Hugin.