Triangulation heuristics for BN2O networks*

Petr Savicky[1] and Jiří Vomlel[2]

[1] Institute of Computer Science
Academy of Sciences of the Czech Republic
Pod vodárenskou věží 2,
182 07 Praha 8, Czech Republic
http://www.cs.cas.cz/savicky

[2] Institute of Information Theory and Automation of the AS CR,
Academy of Sciences of the Czech Republic
Pod vodárenskou věží 4,
182 08 Praha 8, Czech Republic
http://www.utia.cas.cz/vomlel

Abstract. A BN2O network is a Bayesian network having the structure of a bipartite graph with all edges directed from one part (the top level) toward the other (the bottom level) and where all conditional probability tables are noisy-or gates. In order to perform efficient inference graphical transformations of these networks are performed. The efficiency of inference is proportional to the total table size of tables corresponding to the cliques of the triangulated graph. Therefore in order to get efficient inference it is desirable to have small cliques in the triangulated graph. We analyze existing heuristic triangulation methods applicable to BN20 networks after transformations parent divorcing and tensor rank-one decomposition and suggest several modifications. Both theoretical and experimental results confirm that tensor rank-one decomposition yields better results than parent divorcing in randomly generated BN2O networks, which we tested.

1 Introduction

A BN2O network is a Bayesian network having the structure of a directed bipartite graph with all edges directed from one part (the top level) toward the other (the bottom level) and where all conditional probability tables are noisy-or gates. Since the table size for a noisy-or gate is exponential in the number of its parents, graphical transformations of these networks are performed in order to reduce the table size and allow efficient inference. This paper deals with two transformations - parent divorcing [6] and rank-one decomposition [5, 10, 8]. Typically, in order to get an inference structure, the graph obtained by one

* The authors were supported by the Ministry of Education of the Czech Republic under the projects 1M0545 (P. Savicky), 1M0572, and 2C06019 (J. Vomlel). J. Vomlel was also supported by the Eurocores LogICCCC Project LcpR (ICC/08/E010) and by the project 201/09/1891 of the Grant Agency of the Czech Republic.

of the above transformations is further transformed by the following two consecutive steps - moralization and triangulation - that result in an undirected triangulated graph. The efficiency of inference is proportional to the total table size (tts) of tables corresponding to the cliques of the triangulated graph. The size of the largest clique minus one is often called the graph treewidth (tw). Since BN2O network consists only of binary variables, the size of the largest probability table is 2^{tw+1}. This is exponential, while the number of probability tables is polynomial, hence 2^{tw+1} approximates tts up to a polynomial factor.

In order to get efficient inference it is desirable to have small total table size in the triangulated graph. Both methods parent divorcing and rank-one decomposition were designed to minimize the size of probability tables before triangulation. In this paper, we consider the total table size after triangulation, which is the crucial parameter for the efficiency of the inference. From this point of view, parent divorcing appears to be clearly inferior. In Section 2 we show that the treewidth tw of the optimally triangulated graph of the BN2O network after rank-one decomposition (base ROD graph) is never larger than the treewidth of the model preprocessed using parent divorcing (PD) and the same holds for the total table size tts. Hence, if we can use optimal elimination ordering (EO) for the transformed graphs, using ROD we never get results worse by more than a linear term. Since the search for the optimal EO is NP-hard [11], we have to use heuristics. In this case, ROD is also not worse, since the upper bound on tw and tts for ROD holds efficiently, i.e. there is an efficient procedure, which transforms an EO for PD into an EO for base ROD graph with the required upper bound on tw and tts.

Having the above facts in mind, we concentrate in Section 3 on the search of a good EO for the BN2O graphs after the ROD transformation. We analyze existing heuristic triangulation methods applicable to BN20 networks and suggest several modifications. The experimental results in Section 4 confirm that these modifications further improve the quality of the obtained triangulation of randomly generated BN2O networks, which we used.

2 Transformations of BN2O networks

First, we briefly introduce the necessary graph notions. For more details see, e.g. [3].

Definition 1. An undirected graph G is triangulated if it does not contain an induced subgraph that is a simple cycle (i.e., a cycle without a chord) of length at least four.

Definition 2. A triangulation of G is a triangulated graph H that contains the same nodes as G and contains G as a subgraph.

Definition 3. A set of nodes $C \subseteq V$ of a graph $G = (V, E)$ is a clique if it induces a complete subgraph of G and it is not a subset of the set of nodes of any larger complete subgraph of G.

Definition 4. For any graph G, let $\mathcal{C}(G)$ be the set of all cliques of G.

Definition 5. The treewidth of a triangulation H of G is the maximum clique size in H minus one. The treewidth of G, denoted $tw(G)$, is the minimum treewidth over all triangulations H of G.

Definition 6. The table size of a clique C in an undirected graph is the product of the number of states of variables corresponding to the nodes of the clique C.

In this paper all variables are binary, therefore the table size of a clique C is $2^{|C|}$.

Definition 7. The total table size of a triangulation H of G is the sum of table sizes of all cliques of H. The total table size of a graph, denoted $tts(G)$, is the minimum total table size over all triangulations H of G.

Definition 8. Elimination ordering of an undirected graph $G = (V, E)$ is a bijection $f : V \to \{1, 2, \ldots, n\}$.

The meaning of this representation is that for every node u, the number $f(u)$ is the index of u in the represented ordering.

Definition 9. An elimination ordering $f : V \to \{1, 2, ..., n\}$ of an undirected graph $G = (V, E)$ is perfect if for all $v \in V$, the set

$$B(v) = \{w \in V : \{v, w\} \in E \text{ and } f(w) > f(v)\}$$

induces a complete subgraph of G.

If a graph possesses a perfect elimination ordering, then it is triangulated. If a graph $G = (V, E)$ is not triangulated, then we may triangulate it using any given elimination ordering f by considering the nodes in V in the order defined by f, i.e. such that $f(v) = 1, 2, \ldots, n$ and sequentially adding edges to E so that after considering node v the set $B(v)$ induces a complete subgraph of G.

Now, we restrict our attention to the family of BN2O networks and define the corresponding graphs.

Definition 10. $G = (U \cup V, E)$ is a graph of a BN2O network (BN2O graph) if it is an acyclic directed bipartite graph, where U is the set of nodes of the top level, V is the set of nodes of the bottom level, and E is a subset of the set of all edges directed from U to V, $E \subseteq \{(u_i, v_j) : u_i \in U, v_j \in V\}$.

See Fig. 1 for an example of a BN2O graph.

Since the conditional probability tables in the BN20 networks take a special form - they are noisy-or gates - we can transform the original BN2O graph and corresponding tables using methods exploiting their special form. In the sequel we deal with two methods - parent divorcing and rank-one decomposition. Since we restrict ourselves to analyze graph triangulation, we concentrate only on the graphical transformations performed when these methods are applied.

The first transformation is parent divorcing [6] and it avoids connecting all parents of each node of V (in the moralization step) by introducing auxiliary nodes in between nodes from U and V. The next definition describes the graph obtained by a specific form of PD together with the moralization step.

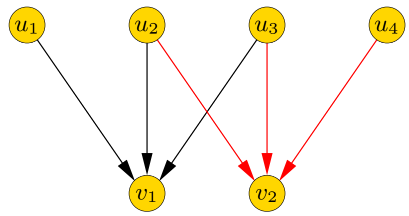

Fig. 1. A BN2O graph

Definition 11. *The parent divorcing (PD) graph of a BN2O graph $G = (U \cup V, E)$ is the undirected graph $G_{PD} = (U \cup V \cup W, H)$, where*

$$W = \cup_{v_i \in V} W_i \text{ and } H = \cup_{v_i \in V} H_i$$

and for each node $v_i \in V$ with $pa(v_i) = \{u_j \in U : (u_j, v_i) \in E\}$ the set of auxiliary nodes

$$W_i = \{w_{i,j}, j = 1, \ldots, k = |pa(v_i)| - 2\}$$

and the set of undirected edges

$$\begin{aligned} H_i = \; & \{ \; \{w_{i,1}, u_{j_1}\}, \{w_{i,1}, u_{j_2}\}, \{u_{j_1}, u_{j_2}\}, \\ & \{w_{i,2}, w_{i,1}\}, \{w_{i,2}, u_{j_3}\}, \{w_{i,1}, u_{j_3}\}, \\ & \ldots, \\ & \{w_{i,k}, w_{i,k-1}\}, \{w_{i,k}, u_{j_{k+1}}\}, \{w_{i,k-1}, u_{j_{k+1}}\}, \\ & \{v_i, w_{i,k}\}, \{v_i, u_{j_{k+2}}\}, \{w_{i,k}, u_{j_{k+2}}\} \;\; \} \;\; , \end{aligned}$$

where $\{u_{j_1}, \ldots, u_{j_{k+2}}\} = pa(v_i)$.

See Fig. 2 for an example of a PD graph.

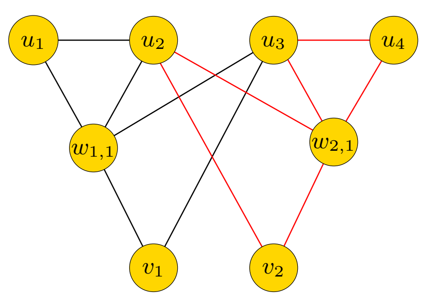

Fig. 2. The PD graph of BN2O graph from Fig. 1

The second transformation - rank-one decomposition - was originally proposed by Díez and Galán [5] for noisy-max models and extended to other models by Savicky and Vomlel [10, 8].

Definition 12. The rank-one decomposition (ROD) graph of a BN2O graph $G = (U \cup V, E)$ is the undirected graph $G_{ROD} = (U \cup V \cup W, F)$ constructed from G by adding an auxiliary node w_i for each $v_i \in V$, $W = \{w_i : v_i \in V\}$, and by replacing each directed edge $(u_j, v_i) \in E$ by undirected edge $\{u_j, w_i\}$ and adding an undirected edge $\{v_i, w_i\}$ for each $v_i \in V$:

$$F = \{\{u_j, w_i\} : (u_j, v_i) \in E\} \quad \cup \quad \{\{v_i, w_i\} : v_i \in V\}$$

See Fig. 1 for an example of an ROD graph.

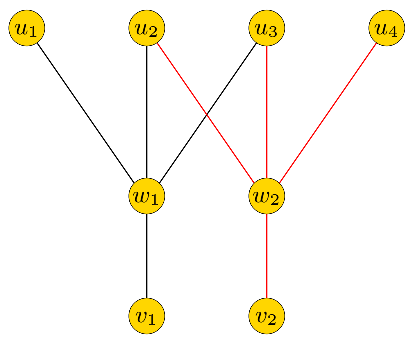

Fig. 3. The ROD graph of BN2O graph from Fig. 1

Note that nodes $v_i \in V$ are simplicial[3] in the ROD graph and have degree one therefore we can perform optimal triangulation of the ROD graph by optimal triangulation of its subgraph generated by nodes $U \cup W$ [3]. This graph will be called base ROD graph or shortly BROD graph. For the treewidth it holds

$$tw(G_{ROD}) = \max\{1, tw(G_{BROD})\}$$

and for the total table size

$$tts(G_{ROD}) = tts(G_{BROD}) + 4|W| \ .$$

See Figure 4 for the BROD graph of BN2O graph from Fig 1.

Definition 13. A graph H is a minor of a graph G if H can be obtained from G by any number of the following operations:

- node deletion,
- edge deletion, and
- edge contraction[4].

[3] A node is simplicial in G if its neighbors generate a complete subgraph of G.

[4] Edge contraction is the operation that replaces two adjacent nodes u and v by a single node w that is connected to all neighbors of u and v.

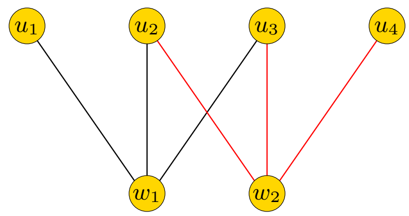

Fig. 4. The BROD graph of BN2O graph from Fig 1 i.e., the subgraph on the nodes $U \cup W$ of the ROD graph from Fig. 3.

Lemma 1. *The BROD graph is a graph minor of the PD graph.*

Proof. For each set of edges H_i in the PD graph (see Definition 11) we delete the edge $\{u_{j_1}, u_{j_2}\}$ and contract edges

$$\{v_i, w_{i,k}\}, \{w_{i,k}, w_{i,k-1}\}, \ldots, \{w_{i,2}, w_{i,1}\}$$

and name the resulting node w_i. By these edge contractions the node w_i gets connected by undirected edges to all $u_j \in pa(v_i)$. By repeating this procedure for all $i, v_i \in V$ we get the BROD graph. □

Theorem 1. *The treewidth of the BROD graph is never larger than the treewidth of the PD graph.*

Proof. Due to Lemma 1 the BROD graph is a graph minor of the PD graph. Therefore we can apply the well-known theorem (see, e.g. Lemma 16 in [2]) that the treewidth of a graph minor is never larger than the treewidth of the graph itself. □

Lemma 2. *Let G be a triangulated undirected graph and H be the resulting graph after a contraction of an edge. Then $tts(H) \leq tts(G)$.*

Proof. Let H be the resulting graph after a contraction of an edge $\{u, v\}$ in G replaced by node w in H. Let ϕ be a mapping of nodes of G onto the nodes of H such that it is identity, except $\phi(u) = w$ and $\phi(v) = w$. Let us prove that for every clique D in H there exists a clique C in G such that $D = \phi(C)$. This assertion is obvious for cliques of H not containing node w. Le D be a clique in H containing node w. For $|D| = 1$ is the assertion is also obvious. For $|D| = 2$ it holds that $D = \phi(\{u, a\}) = \phi(\{v, a\}) = \phi(\{u, v, a\})$, where $a \neq w$ is a node from D. Furthermore, either $\{u, a\}$, $\{v, a\}$, or $\{u, v, a\}$ is a clique in G.

Now assume that $|D| \geq 3$. Denote

$$\begin{aligned} C_1 &= (D \setminus \{w\}) \cup \{u\} \\ C_2 &= (D \setminus \{w\}) \cup \{v\} \\ C_3 &= (D \setminus \{w\}) \cup \{u, v\} \ . \end{aligned}$$

It holds that $D = \phi(C_1) = \phi(C_2) = \phi(C_3)$. To show that either C_1 or C_2 is a complete subgraph of G assume by contradiction that neither C_1 nor C_2 is a complete subgraph of G. Then there would exist nodes $a, b \in D \setminus \{w\}$ such that (a, u) and (b, v) are not edges in G. Since w is connected by an edge to all nodes from $D \setminus \{w\}$, $a \neq b$ and (a, b), (a, v), and (b, u) are edges of G. Consequently the cycle (a, v, u, b) does not have a chord in G, which is in contradiction with the assumption that G is triangulated.

Hence, for some $i = 1, 2$, C_i is complete in G. Therefore one of $C_i, i = 1, 2, 3$ must be a clique in G - none of the strict supersets of C_3 can be a clique in G since this would contradict the assumption that D is a clique.

The properties of the mapping ϕ imply that there is an injective mapping from $\mathcal{C}(H)$ to $\mathcal{C}(G)$ non-decreasing the size of the cliques. Hence, we have

$$\sum_{A \in \mathcal{C}(H)} 2^{|A|} \leq \sum_{A \in \mathcal{C}(G)} 2^{|A|} \quad ,$$

which implies $tts(H) \leq tts(G)$, since G and H are triangulated. □

Lemma 3. *Let $G = (V, E)$ be a triangulated graph with a perfect elimination ordering f and $H = (U, F)$ be the graph constructed from G by edge $\{u, v\}$ contraction with the resulting node named w. Further let $f(u) < f(v)$. Then H is triangulated and its elimination ordering g constructed from f by:*

$$g(a) = \begin{cases} f(a) & \text{if } f(a) < f(u) \\ f(v) & \text{if } a = w \\ f(a) - 1 & \text{otherwise.} \end{cases}$$

is perfect.

Proof. By the definition of perfect elimination ordering (Definition 9) it is sufficient to show that for all nodes $a \in U$ the set

$$B_H(a) = \{b \in U : \{a, b\} \in F \quad \text{and} \quad f(b) > f(a)\}$$

induces a complete subgraph of H. Since f is a perfect elimination ordering of G it holds for all nodes $a \in V$ that the set

$$B_G(a) = \{b \in V : \{a, b\} \in E \quad \text{and} \quad f(b) > f(a)\}$$

induces a complete subgraph of G. Nodes $a \in U \setminus \{w\}$ have either $B_H(a) = B_G(a)$ or $B_H(a) = (B_G(a) \setminus \{u, v\}) \cup \{w\}$. In both cases these sets induces a complete subgraph of H. Every node $z \in B_G(u)$ is connected by an edge with v in G since $v \in B_G(u)$. Therefore $\{x \in B_G(u), f(x) > f(v)\} \subseteq B_G(v)$. Consequently, $B_H(w) = B_G(v)$ and induces a complete subgraph of H. □

The following two lemmas hold for general graphs, not necessarily triangulated.

Lemma 4. *Let S be a set of some sets inducing complete subgraphs of a graph H. Then S is the set of all cliques of H iff S contains only incomparable pairs of sets and each set inducing a complete subgraph of H is a subset of an element of S.*

Proof. The set of all cliques satisfies the properties from the lemma. For the opposite direction assume that S satisfies the properties from the lemma. Then, S contains all cliques, since a clique is not a subset of any other complete subgraph of H. Since S contains only incomparable pairs of complete subgraphs and contains all cliques, it cannot contain any complete subgraph, which is not a clique.

Lemma 5. *If a graph H is obtained from a graph G by removing an edge $\{u, v\}$, then*

$$\sum_{A \in \mathcal{C}(G)} 2^{|A|} \geq \sum_{A \in \mathcal{C}(H)} 2^{|A|} \ .$$

Proof. Let $R = \mathcal{C}(G)$. Let S be the set of sets inducing complete subgraphs of H of the following two types. Cliques of G not containing both nodes u, v are elements of S. For each clique C of G, which contains both u, v, the sets inducing complete subgraphs $C \setminus \{u\}$ and $C \setminus \{v\}$ of H are elements of S. Hence, S is a set of some sets inducing the complete subgraphs of H. By construction, we have

$$\sum_{A \in R} 2^{|A|} \geq \sum_{A \in S} 2^{|A|}$$

since the elements of S, which are not in R, form pairs of sets of the same size such that R contains their union, which is by one node larger.

Let S' be the subset of S containing only those elements of S, which are not properly contained in some other element of S. Clearly, we have

$$\sum_{A \in R} 2^{|A|} \geq \sum_{A \in S'} 2^{|A|}$$

It is now sufficient to prove that S' is the set of all cliques of H by proving that S' satisfies the properties from Lemma 4. By construction, it contains only incomparable sets. Let A induce complete subgraph of H. Then A also induces a complete subgraph of G and is contained in some clique C of G. If C does not contain both nodes u, v, then C is in S. If C contains both u, v, then A is contained in some of the sets $C \setminus \{u\}$ and $C \setminus \{v\}$ and both of them are in S. Hence, A is contained in an element of S. Consequently, it is also contained in an element of S'. □

Theorem 2. *For any given elimination ordering f of the PD graph we can efficiently construct an elimination ordering g of the BROD graph such that the treewidth (and the total table size) of the BROD graph triangulated using g is not larger than the treewidth (and the total table size, respectively) of the PD graph triangulated using f.*

Proof. Let f be an elimination ordering for G_{PD}, which yields a triangulation G^f_{PD}. Let us construct a triangulation G' of the G_{BROD} from G^f_{PD} using the same sequence of edge contractions as in the proof of Lemma 1. Along these transformations we apply Lemma 3 to get an elimination ordering g for G' and by repeated application of Lemma 2, we obtain $tts(G') \leq tts(G^f_{PD})$.

Graph G' has the same nodes as G_{BROD} and contains G_{BROD} as a subgraph. Let G^g_{BROD} be the triangulation of G_{BROD} obtained using the ordering g. In each step of the process of triangulation of G_{BROD} using g we add only edges, which belong to G'. Hence, the resulting graph G^g_{BROD} is a subgraph of G'. Consequently, by repeated use of Lemma 5 for all edges of G', which do not belong to G^g_{BROD}, we obtain $tts(G^g_{BROD}) \leq tts(G')$. This proves the statement concerning total table size. The statement concerning treewidth follows from the fact that G^g_{BROD} is a graph minor of G^f_{PD} and, hence, cannot have larger treewidth. □

Corollary 1. *The total table size of the BROD graph is never larger than the total table size of the PD graph.*

Proof. Use Theorem 2 for elimination ordering f, which yields a triangulation of PD graph with the smallest total table size. □

3 Triangulation heuristics

In the previous section we have shown that the PD graph is inferior to the BROD graph in the sense that we can always triangulate the BROD graph so that its treewidth (or total table size) is not greater than the treewidth (or total table size, respectively) of the PD graph. Therefore, in this section, we will pay attention to efficient triangulation of the BROD graph.

First, we applied several well-know triangulation heuristics to the BROD graph. We tested minfill [7], maximum cardinality search [9], minwidth [7], H1, and H6 [4]. Minfill gave by far the best results. The minfill heuristics that returns an elimination ordering f of $G = (V, E)$ is described in Table 1.

Table 1. The minfill algorithm

For $i = 1, \ldots, |V|$ do:

1. For $u \in V$ define set of edges $F(u) = \{\{u_1, u_2\} : \{u_1, u\} \in E, \{u_2, u\} \in E\}$ to be added for elimination of u.
2. Select a node $v \in V$ which adds the least number of edges when eliminated, i.e., $v \in \arg\min_{u \in V} |F(u) \setminus E|$, breaking ties arbitrarily.
3. Set $f(v) = i$.
4. Make v a simplicial node in G by adding edges to G, i.e., $G = (V, E \cup F(v))$.
5. Eliminate v from the graph G, i.e. replace G by its induced subgraph on $V \setminus \{v\}$.

Return f.

Minfill of the PD graph used for the BROD graph

In our experiments we have observed for some BN2O graphs that the triangulation of PD graph by minfill lead to a graph with a smaller total table size than the triangulation of BROD graph by minfill. This may seem to contradict the results from the previous section but it does not since the triangulation heuristics do not guarantee that they find optimal triangulation. In order to avoid this unwanted phenomenon, we can use the elimination ordering f found by minfill for the PD graph and construct an elimination ordering g for the BROD graph using the construction given in the proof of Lemma 2. This lemma guarantees that the total table size of the BROD graph triangulated using g is not larger than the total table size of the PD graph triangulated using f. We refer to this method as *minfill-pd* and use it as a base method for the comparisons in Section 4.

Minfill with n steps look-ahead

Since the minfill heuristics is computationally very fast for networks of moderate size, one can minimize the total number of edges added to the graph after more than one node is eliminated, i.e., one can look n steps ahead. Of course, this method scales exponentially, therefore it is computationally tractable only for small n. We refer to this method as *minfill-n-ahd*.

Minfill that prefers nodes from the larger level

The following proposition motivates another modification of the minfill heuristics.

Proposition 1. *Let $G = (U \cup W, F)$ be a BROD graph. Then*

$$tw(G) \leq \min\{|U|, |W|\} \ .$$

This upper bound on the treewidth suggests a modification of the minfill heuristics. We can enforce edges to be filled in the smaller level only by taking nodes from the larger level into the elimination ordering first. Within the larger level we can use the minfill heuristics to choose the elimination ordering of nodes from this level. This gives treewidth at most the number of nodes in the smaller level. The nodes from the smaller level are included in the elimination ordering after the nodes from the larger level. We will refer this method as *minfill-pll*.

By combination of methods we understand running several methods independently and selecting the best result. In particular, we tested this type of combination of *minfill-pll* and *minfill* and refer to this combination as *minfill-comb*.

4 Experiments

The test benchmark

In this section we experimentally compare the proposed triangulation heuristics on 1300 randomly generated BN2O networks. The BN2O graphs were generated with varying values of the following parameters:

- x, the number of nodes in the top level,
- y, the number of nodes in the bottom level, and
- e, the average number of edges per node in the bottom level.

For each x-y-e type, $x, y = 10, 20, 30, 40, 50$ and $e = 3, 5, 7, 10, 14, 20$ (excluding those with $e \geq x$) we generated randomly ten BN2O graphs.

All triangulation heuristics were tested on the BROD graphs G_{BROD}. We used the total table size tts of the graph G^h_{BROD} triangulated by a triangulation heuristics h as the criterion for the comparisons. We used the *minfill-pd* method as the base method against which we compared all other tested methods. Since randomness is used in the triangulation heuristics we run each heuristics ten times on each model and selected a triangulation with the minimal value of total table size tts.

Experimental results

For each tested model we computed the decadic logarithm ratio

$$r(pd, h) = \log_{10} tts\big(G^{pd}_{BROD}\big) - \log_{10} tts\big(G^{h}_{BROD}\big) ,$$

where h stands for the tested triangulation heuristics. In Table 2 we give frequencies of several intervals of log-ratio $r(pd, h)$ values of the tested heuristics in the test benchmark.

From the table we can see that in average all tested heuristics perform significantly better than *minfill-pd*, since positive differences of the logarithms are more frequent and achieve larger absolute value. On the other hand, most of the heuristics are worse than minfill-pd for some of the models. Since triangulation heuristics *minfill*, *minfill-pll*, and *minfill-pd* are computationally fast on moderately large networks, the best solution seems to be to run all of these three heuristics and select the best solution. Already *minfill-comb*, which is the combination of *minfill* and *minfill-pll* eliminates most of the cases, where *minfill* is worse than *minfill-pd*.

5 Conclusions

In this paper we compare two transformations of BN2O networks that allow more efficient probabilistic inference - parent divorcing and rank-one decomposition. We prove that the rank-one decomposition is superior to parent divorcing since with the rank-one decomposition we can always get at least as small total table

Table 2. Frequency of $r(pd, h)$ values of the heuristics tested on the test benchmark.

Intervals of $r(pd, h)$	*minfill*	*minfill-1ahd*	*minfill-2ahd*	*minfill-pll*	*minfill-comb*
$(-3, -2]$	5	0	0	0	0
$(-2, -1]$	26	14	9	0	0
$(-1, -0.05]$	96	82	76	116	2
$(-0.05, 0.05]$	518	535	536	695	637
$(0.05, 1]$	328	339	350	177	334
$(1, 2]$	116	115	113	101	114
$(2, 3]$	101	103	104	99	101
$(3, 4]$	29	31	31	31	31
$(4, 5]$	27	27	27	27	27
$(5, 6]$	34	33	33	33	33
$(6, 7]$	9	10	10	10	10
$(7, 8]$	3	3	3	3	3
$(8, 9]$	8	8	8	8	8

size of the resulting triangulated graph as with the parent divorcing, but in many cases, we may achieve much better result. We perform experiments with different triangulation heuristics and suggest few modifications of the minfill heuristics for BN2O networks. The experiments reveal that all proposed heuristics perform significantly better on average than the heuristics based on elimination ordering derived from PD graph, but none of the heuristics is universally the best. In order to get the best result for all models we suggest to run the minfill heuristics on PD graph *minfill-pd*, minfill heuristics on the BROD graph *minfill*, and minfill that prefers nodes from the larger level on the BROD graph *minfill-pll* and select the best solution from these three. If there is enough computational resources running the minfill n step look-ahead on the BROD graph *minfill-n-ahd* can help to find even better triangulation.

Acknowledgments

We would like to thank Mark Chavira for providing us with the code extracted from Ace [1], which we have used for the construction of PD graphs and for the computation of elimination orderings in these graphs.

References

1. Ace. A Bayesian network compiler. `http://reasoning.cs.ucla.edu/ace/`, 2008.
2. H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1–2):1–45, 1998.
3. H. L. Bodlaender, A. M. C. A. Koster, and F. Van Den Eijkhof. Preprocessing rules for triangulation of probabilistic networks. *Computational Intelligence*, 21(3):286–305, 2005.

4. A. Cano and S. Moral. Heuristic algorithms for the triangulation of graphs. In B. Bouchon-Meunier, R. R. Yager, and L. A. Zadeh, editors, *Advances in Intelligent Computing – IPMU '94: Selected Papers*, pages 98–107. Springer, 1994.
5. F. J. Díez and S. F. Galán. An efficient factorization for the noisy MAX. *International Journal of Intelligent Systems*, 18:165–177, 2003.
6. K. G. Olesen, U. Kjærulff, F. Jensen, F. V. Jensen, B. Falck, S. Andreassen, and S. K. Andersen. A MUNIN network for the median nerve — a case study on loops. *Applied Artificial Intelligence*, 3:384–403, 1989. Special issue: Towards Causal AI Models in Practice.
7. D. J. Rose. A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. *Graph Theory and Computing*, pages 183–217, 1972.
8. P. Savicky and J. Vomlel. Exploiting tensor rank-one decomposition in probabilistic inference. *Kybernetika*, 43(5):747–764, 2007.
9. R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM J. Comput.*, 13:566–579, 1984.
10. J. Vomlel. Exploiting functional dependence in Bayesian network inference. In *Proceedings of the 18th Conference on Uncertainty in AI (UAI)*, pages 528–535. Morgan Kaufmann Publishers, 2002.
11. M. Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM J. Algebraic and Discrete Methods*, 2:77–79, 1981.