Conference Publication

Publication	
Title	Heuristics for determining the elimination ordering in the influence diagram evaluation with binary trees
Authors	R. Cabañas, A Cano, M Gómez-Olmedo, and A.L. Madsen
Year	2013
DOI	https://doi.org/10.3233/978-1-61499-330-8-65

Conference details	
Book title	Twelfth Scandinavian Conference on Artificial Intelligence: SCAI 2013
Series	Frontiers in Artificial Intelligence and Applications
Volume	257
Location	Aalborg, Denmark

Twelfth Scandinavian Conference on Artificial Intelligence
M. Jaeger et al. (Eds.)
IOS Press, 2013

doi:10.3233/978-1-61499-330-8-65

Heuristics for Determining the Elimination Ordering in the Influence Diagram Evaluation with Binary Trees

Rafael CABAÑAS [a] Andrés CANO [a] and Manuel GÓMEZ-OLMEDO [a]
Anders L. MADSEN [b,c]

[a] *Department of Computer Science and Artificial Intelligence*
CITIC-UGR, Granada University, Spain

[b] *HUGIN EXPERT A/S*
Aalborg, Denmark

[c] *Department of Computer Science*
Aalborg University, Denmark

Abstract. Finding an optimal elimination ordering is a NP-hard problem of crucial importance for the efficiency of the Influence Diagrams evaluation. Some of the traditional methods for determining the elimination ordering use heuristics that consider that potentials are represented as tables. However, if potentials are represented using binary trees traditional methods may not offer the best results. In the present paper, two new heuristics that consider that potentials are represented as binary trees are proposed. As a result, the storage requirements for evaluating an ID with binary trees is reduced.

Keywords. Influence Diagrams, elimination ordering, heuristics, binary trees, variable elimination

Introduction

An Influence Diagram (ID) [7] is a Probabilistic Graphical Model for decision analysis under uncertainty. Probability and utility functions attached to an ID represent, respectively, the uncertainty and the user preferences in the decision problem. In general, we will talk about potentials (not necessarily normalized). The evaluation of IDs related to complex decision problems becomes infeasible due to its computational cost: The set of information states exceeds the storage capacity of PCs or the optimal policy cannot be determined sufficiently fast. Some of the approximate methods use alternative representations for potentials such as *binary trees* (BTs). This kind of trees offers the possibility of taking advantage of *contextual-weak independencies* [2,17], allowing a smaller representation for the potential. Moreover, if BTs are too large, they can be pruned and converted into smaller trees, thus leading to approximate and more efficient algorithms.

Several approaches have been proposed to evaluate IDs such as *Variable Elimination* [8,16]. This method starts with a set of potentials and it eliminates one variable at

each time. As it happens with the corresponding method for Bayesian networks (BNs) [18], the efficiency of VE depends heavily on the optimality of the elimination order. In fact, any method for finding an optimal ordering in a BN, which is an NP-hard problem [10], can be adapted for IDs. Some of the most efficient methods for BNs are greedy algorithms that choose at each step the next variable to remove using a heuristic [14,10,4]. These algorithms try to minimise the complexity of the operations involved in the evaluation.

The problem of traditional heuristics is that they assume that potentials are represented using tables, and the complexity of the operations might be different if potentials are represented using BTs. In the present paper two new heuristics that consider that potentials are represented using BT are proposed. These heuristics estimates the size of potentials generated when removing a variable.

The paper is organized as follows: Section 1 introduces some basic concepts about IDs, Variable Elimination, traditional heuristics and BTs; Section 2 describes key issues about the new heuristics proposed; Section 3 includes the experimental work and results; finally Section 4 details our conclusions and lines for future work.

1. Preliminaries

1.1. Influence Diagrams

An ID [7] is a Probabilistic Graphical Model for decision analysis under uncertainty which contains three kinds of nodes: *chance nodes* that represent random variables; *decision nodes* that correspond with the actions which the decision maker can control; and *utility nodes* that represent decision maker preferences.

We denote by $\mathscr{U}_C$ the set of chance nodes, by $\mathscr{U}_D$ the set of decision nodes, and by $\mathscr{U}_V$ the set of utility nodes. The decision nodes have a temporal order, $D_1,\ldots,D_n$, and the chance nodes are partitioned into a collection of disjoint sets according to when they are observed: $\mathscr{I}_0$ is the set of chance nodes observed before D_1, and $\mathscr{I}_i$ is the set of chance nodes observed after decision D_i is taken and before decision D_{i+1} is taken. Finally, $\mathscr{I}_n$ is the set of chance nodes observed after D_n. That is, there is a partial order:

$$\mathscr{I}_0 \prec D_1 \prec \mathscr{I}_1 \prec \cdots \prec D_n \prec \mathscr{I}_n$$

The *universe* of the ID is $\mathscr{U} = \mathscr{U}_C \cup \mathscr{U}_D = \{X_1,\ldots,X_m\}$. Let us suppose that each variable X_i takes values on a finite set $\Omega_{X_i} = \{x_1,\ldots,x_{|\Omega_{X_i}|}\}$. Each chance node X_i has a conditional probability distribution $P(X_i|pa(X_i))$ associated. In the same way, each utility node V_i has a utility function $U(pa(V_i))$ associated. In general, we will talk about potentials (not necessarily normalized). The set of all variables involved in a potential ϕ is denoted $dom(\phi)$, defined on $\Omega_{dom(\phi)} = \times\{\Omega_{X_i}|X_i \in dom(\phi)\}$. The elements of $\Omega_{dom(\phi)}$ are called configurations of ϕ. Therefore, a *probability potential* denoted by ϕ is a mapping $\phi : \Omega_{dom(\phi)} \to [0,1]$. A *utility potential* denoted by ψ is a mapping $\psi : \Omega_{dom(\psi)} \to \mathbb{R}$. The set of probability potentials is denoted by Φ while the set of utility potentials is denoted by Ψ.

Informational predecessors or parents of each decision D_i, denoted $pa(D_i)$, must include previous decisions and their informational predecessors (*no-forgetting assumption*). The goal of evaluating an ID is to obtain an *optimal policy* δ_i for each decision D_i, that is a function of a subset of its parents. The optimal policy maximizes the *expected utility* for the decision.

Expected Utility: *Let ID be an influence diagram over the universe $\mathscr{U} = \mathscr{U}_C \cup \mathscr{U}_D$ and let $\mathscr{U}_V$ the set of utility nodes. Let us suppose that the ID meets the no-forgetting assumption. Let the temporal order of the variables be described as $\mathscr{I}_0 \prec D_1 \prec \mathscr{I}_1 \prec \cdots \prec D_n \prec \mathscr{I}_n$. Then the expected utility for D_i is:*

$$EU(\mathscr{I}_0, D_1, \ldots, \mathscr{I}_{i-1}) = \frac{1}{P(\mathscr{I}_0, \ldots, \mathscr{I}_{i-1} | D_1, \ldots, D_{i-1})} \max_{D_i} \sum_{\mathscr{I}_i} \max_{D_{i+1}} \cdots \max_{D_n} \sum_{\mathscr{I}_n} \prod_{X \in \mathscr{U}_C} P(X|pa(X)) \left(\sum_{V \in \mathscr{U}_V} U(pa(V)) \right) \quad (1)$$

1.2. Variable Elimination

The *Variable Elimination* algorithm (VE) for IDs [9,13] has many similarities with the corresponding method for BNs [18]: it starts with a set of potentials and it eliminates one variable at each time. There are however differences. First of all, all variables in the decision problem are removed in reverse order of the partial ordering imposed by the information constraints. Secondly, chance variables are removed using sum-marginalization whereas decision variables are removed using max-marginalization. That is, it first sum-marginalizes $\mathscr{I}_n$, then max-marginalizes D_n, sum-marginalizes $\mathscr{I}_{i-1}$, etc. This type of elimination order is called a strong elimination order [10]. The procedure for removing a variable X_i is shown in Proposition 1. It can be observed that the sets of potentials Φ and Ψ change along the evaluation process.

Proposition 1 (Removal of a variable) *Let X_i be the variable to be removed, Φ and Ψ the set of all probability and utility potentials respectively. Then, the procedure for removing X_i is:*

1. *Select the sets of relevant potentials:*

$$\Phi_{X_i} = \{\phi \in \Phi | X_i \in dom(\phi)\} \quad \Psi_{X_i} = \{\psi \in \Psi | X_i \in dom(\psi)\}$$

2. *Combine potentials*

$$\phi_{X_i} = \prod \Phi_{X_i} \quad \psi_{X_i} = \prod \Phi_{X_i} (\sum \Psi_{X_i})$$

3. *Remove the variable X_i*
   - *If X_i is a chance variable, it is removed using sum-marginalization:*

$$\phi'_{X_i} = \sum_{X_i} \phi_{X_i} \quad \psi'_{X_i} = \sum_{X_i} \psi_{X_i}$$

   - *If X_i is a decision variable, it is removed using max-marginalization:*

$$\phi'_{X_i} = \max_{X_i} \phi_{X_i} \quad \psi'_{X_i} = \max_{X_i} \psi_{X_i}$$

4. *Update the potentials*

$$\Phi = (\Phi \backslash \Phi_{X_i}) \cup \{\phi'_{X_i}\} \quad \Psi = (\Psi \backslash \Psi_{X_i}) \cup \{\frac{\psi'_{X_i}}{\phi'_{X_i}}\}$$

The set of all variables contained in the potentials relevant for the removal of a variable is called *clique candidate* or *group*. That is $C_{X_i} = \{dom(\phi_{X_i}) \cup dom(\psi_{X_i})\}$ is the clique candidate created when X_i is removed. Even though the concept of clique is usually only used with triangulation algorithms, it corresponds with the variables involved during the removal of a variable using VE algorithm. The *size* of a clique candidate $|C_{X_i}|$ is the number of variables on it. The *weight* of a clique candidate is the product of the number of states of each variable. That is $w(C_{X_i}) = \prod_{X_j \in C_i} |\Omega_{X_j}|$. The size of a clique candidate corresponds with the number of variables in the potential ψ_{X_i} resulting of combining all the relevant potentials (Proposition 1, step 2). The combination of two potentials ϕ_1 and ϕ_2 is denoted $\phi_1 \otimes \phi_2$. The weight corresponds with the number of values (configurations) in the same potential. During the elimination process, it will be necessary to add new links (called *fill-in arcs*). For example, let us consider X_i a variable to remove, and the two variables $\{A,B\} \in C_{X_i}$. A fill-in arc is added between A and B if there is not any potential in $\Phi_{X_i} \cup \Psi_{X_i}$ containing both variables. The weight of an arc (A,B) is defined as $w(A,B) = |\Omega_A| \cdot |\Omega_B|$.

1.3. Finding an optimal elimination ordering

An element of crucial importance for the efficiency of the VE algorithm is finding an optimal elimination ordering, which can reduce the complexity of operations performed during the ID evaluation. As in the case of BNs, this optimization problem is NP-hard [10]. In fact, any method for finding an optimal ordering in a BN can be adapted for IDs. The single difference is that it must be consistent with the partial temporal order $\mathscr{I}_0 \prec D_1 \prec \mathscr{I}_1 \prec \cdots \prec D_n \prec \mathscr{I}_n$. Therefore the problem of finding an optimal elimination ordering for the variables in a ID consists on finding an ordering for each $\mathscr{I}_i$.

Several approaches have been proposed in order to search for close-to-optimal elimination orderings. Some of the most efficient methods are greedy algorithms that choose at each step the next variable to remove using a deterministic heuristic. Some of these heuristics are:

- **Minimum size**: this heuristic is based on selecting as the next variable to be removed that one which minimises the size of the clique candidate generated [14]. That is, it selects a variable X_i with a minimal $|C_{X_i}|$.
- **Minimum weight**: this heuristic is based on selecting as the next variable to be removed that one which minimises the weight of the clique candidate generated [10]. That is, it selects a variable X_i with a miminal $w(C_{X_i})$.
- **Cano and Moral**: this heuristic is very similar to *minumum weight*, at each case it chooses a variable X_i that minimises $w(C_{X_i})/|\Omega_{X_i}|$ [4]. That is, it first removes variables of a larger number of states.
- **Minimum fill-in arcs weight**: this heuristic selects a variable to remove that one which minimises the weights of the fill-in arcs added [5].

1.4. Binary Trees

Traditionally, potentials with discrete variables have been represented using tables. However, other representations have been proposed in order to reduce the storage size of the potentials and improve the efficiency of the evaluation algorithms. An example are bi-

nary trees (BTs) [3]. Figure 1 shows three different representations for the same utility potential: (a) table, (b) an exact BT and (c) an approximate BT.

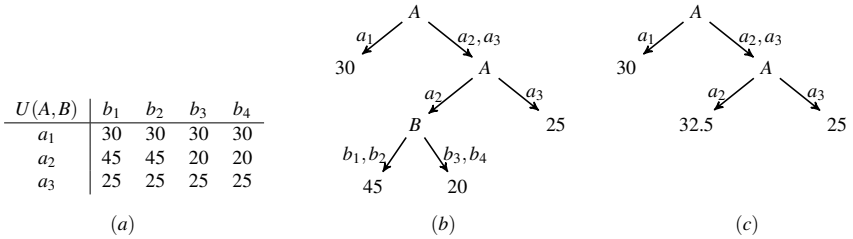


Figure 1. Potential represented as (a) a table, (b) an exact BT and (c) an approximate BT

The main advantage of BTs is that they allow the specification of *contextual-weak independencies* [2,17]. That is, identical values of the potential can be grouped into a single branch, allowing a smaller representation for the potential. For example, the table in Fig. 1 requires 12 values for representing the potential while the exact BT requires 7 nodes. If BTs are too large, they can be pruned and converted into smaller trees, thus leading to approximate algorithms. When a tree is pruned, leaves with a similar value are represented with a single leaf labelled with their mean. The prune is controlled with a threshold $\varepsilon \geq 0$. A low ε value will produce large trees with a low error, while a high ε value will produce small trees with a big error. When $\varepsilon = 0$, the exact prune is performed. That is, only identical values are grouped. When building a tree, variables are sorted in a way that the most informative variables must be situated in the highest nodes of the tree. This operation will reduce the error obtained when pruning a tree. More details for the building and pruning processes are given in [6,3].

A BT representing a potential ϕ is denoted $\mathscr{BT}_\phi$, the lower nodes are called leaves and they are labelled with a value of the potential. If the tree is completely expanded there is a leaf for each configuration in the potential.

Proposition 2 (Size of a BT) *Let $\mathscr{BT}$ be a BT and let $leaves(\mathscr{BT})$ be the number of leaves in the tree. The size of a $\mathscr{BT}$ completely expanded is its number of nodes (leaves and internal nodes) and can be calculated using the following expression:*

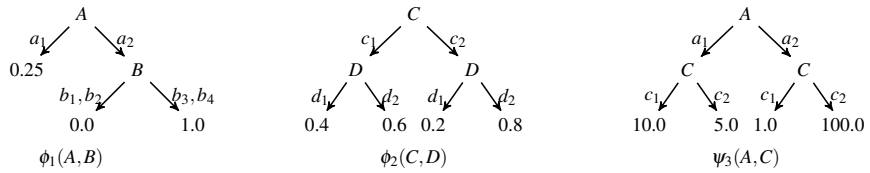
$$size(\mathscr{BT}) = 2 \cdot leaves(\mathscr{BT}) - 1 = 2 \cdot \prod_{X_i \in dom(\mathscr{BT})} |\Omega_{X_i}| - 1$$

2. Heuristics for trees

Traditional heuristics shown in Section 1.3 consider that potentials are represented using tables. If potentials are represented using pruned BTs, traditional heuristics may not offer the best results. This section describes the problems of traditional heuristics. Two new heuristics which consider that potentials are represented as BTs are proposed: *minimum combined tree* and *minimum marginalised tree* heuristics.

2.1. Problems with traditional heuristics

Let $\{A,B,C,D\}$ be a set of variables in an ID and the sizes of their respective domains are: $|\Omega_A| = 2$, $|\Omega_B| = 4$, $|\Omega_C| = 2$, $|\Omega_C| = 2$. Let us suppose that A and C are the two candidate variables to be removed. Assume the ID contains the following potentials:



The clique generated if variable A is eliminated is $C_A = \{A,B,C\}$. By contrast, if C is the variable removed, the clique generated is $C_C = \{A,C,D\}$. The weights of each clique can be calculated as follows:

$$w(C_A) = |\Omega_A| \cdot |\Omega_B| \cdot |\Omega_C| = 2 \cdot 4 \cdot 2 = 16$$

$$w(C_C) = |\Omega_A| \cdot |\Omega_C| \cdot |\Omega_D| = 2 \cdot 2 \cdot 2 = 8$$

If the *minimum weight* heuristic is used, then variable C is selected as the next variable to eliminate. The weights $w(C_A)$ and $w(C_A)$ correspond to the sizes of the tables representing the potentials $\psi_A = \phi_1 \otimes \psi_3$ and $\psi_C = \phi_2 \otimes \psi_3$ respectively. That is, the resulting potentials from combining all the relevant potentials to remove A or C respectively. However, if potentials are represented using BTs previously pruned, the weight might not correspond with the size of the resulting potential. Moreover, the variable that generates a clique of minimal weight might not be the variable that generates a minimal BT. For example, Figures 2 and 3 shows the resulting BTs of combining potentials relevant for removing A and B respectively. The BT representing ψ_A contains 9 nodes whereas the BT representing ψ_C contains 15. Therefore, the best option is to choose the variable A to be removed. However using *minimum weight* heuristic the worst option is chosen (removing variable C). An example where traditional heuristics do not give the best results.

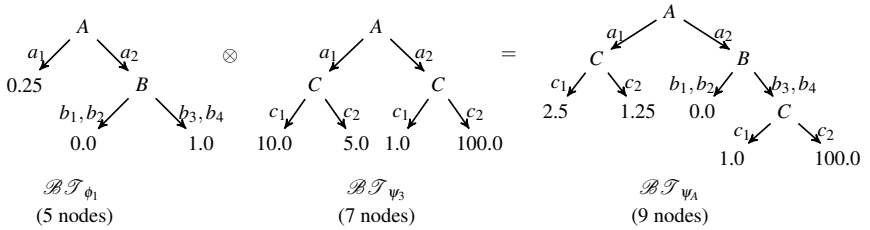


Figure 2. Combination of probability and utility potentials performed if variable A is chosen to be removed

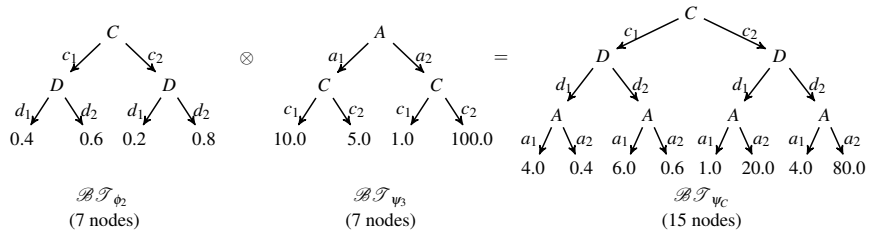


Figure 3. Combination of probability and utility potentials performed if variable C is chosen to be removed

2.2. Minimum combined tree

If the VE algorithm is used to evaluate an ID, the highest memory requirements are achieved when combining all relevant potentials to remove a variable (Proposition 1, step 2). Here the *minimum combined tree* heuristic is proposed, which is similar to *minimum weight* since it aims to calculate the size of the resulting potential of the combination. However, this new heuristic considers that potentials are represented using trees.

Let us suppose that we aim to remove a variable Y, the resulting potential of combining all the relevant potentials is $\psi_Y = (((\phi_1) \otimes \phi_2) \otimes \ldots \otimes \phi_n) \otimes (\psi_1 + \ldots + \psi_m)$. The clique generated (variables involved) is denoted C_Y. The size of the tree representing ψ_Y can be estimated using Eq. (2).

$$size(\mathscr{BT}_{\psi_Y}) \simeq size(\mathscr{BT}_{\phi_1}) + leaves(\mathscr{BT}_{\phi_1}) \cdot (2 \cdot \prod_{\substack{X_i \in C_Y \\ X_i \notin dom(\phi_1)}} |\Omega_{X_i}| - 2) \qquad (2)$$

This heuristic computes the size of $\mathscr{BT}_{\psi_Y}$ from the size of the first potential $\mathscr{BT}_{\phi_1}$ involved in the combination. It checks which of the variables in C_Y are not present in $\mathscr{BT}_{\phi_1}$ and it supposes that a sub-tree with all these variables is added to each leaf of $\mathscr{BT}_{\phi_1}$. Using this heuristic it is selected a variable Y with a minimal $\mathscr{BT}_{\psi_Y}$. For example, let us consider the potentials shown in Section 2.1, then the sizes calculated are:

$$size(\mathscr{BT}_{\psi_A}) \simeq size(\mathscr{BT}_{\phi 1}) + leaves(\mathscr{BT}_{\phi 1}) \cdot (2 \cdot |\Omega_C| - 2) = 5 + 3 \cdot (2 \cdot 2 - 2) = 11$$
$$size(\mathscr{BT}_{\psi_C}) \simeq size(\mathscr{BT}_{\phi 2}) + leaves(\mathscr{BT}_{\phi 2}) \cdot (2 \cdot |\Omega_A| - 2) = 7 + 4 \cdot (2 \cdot 2 - 2) = 15$$

Therefore, using the heuristic *minimum combined tree* variable A is selected since it generates a minimal tree when combining all the potentials involved. Notice that if trees are completely expanded, the size computed is the real one. If trees have been previously pruned, it is not possible to compute the real size without combining them, which is not efficient.

2.3. Minimum marginalised tree

It is also proposed another heuristic that selects a variable that generates a minimal tree after removing the variable (Proposition 1, step 3). This heuristic is called *minimum marginalised tree* and it is quite similar to the *Cano and Moral* heuristic. In this case, variable Y that minimises $size(\mathscr{BT}_{\psi_Y})/|\Omega_Y|$ is selected. As it happen with *minimum combined tree*, it computed an approximation of the tree size.

3. Experimental Work

The aim of the experimental work is to show that the use of specific heuristics for trees reduces the storage size of the potentials during the evaluation. For that purpose two real world IDs are used. First an ID used for the treatment of gastric NHL disease [12] with 3 decisions, 1 utility node and 17 chance nodes. Second, an ID from IctNeo System for jaundice management [1] which contains 2 decisions, 1 utility node and 23 chance nodes. Both IDs are evaluated using BTs for representing the potentials. The heuristics used for determining the elimination order are *minimum weight* (MIN_WEIGHT), *minimum combined tree* (MIN_COMB_TREE), *minimum marginalised tree* (MIN_MARG_TREE) and *minimum fill-in arcs weight* (MIN_FILL_WEIGHT). All BTs are pruned before starting evaluation. The threshold ε used for pruning is ranged in the interval $[0, 0.1]$. All the algorithms are implemented in Java with the Elvira Software[1].

During the evaluation of both IDs, the storage requirements for representing all potentials are analysed (number of nodes). In particular, the measurements are performed after combining all relevant potentials for removing each variable (Proposition 1, step 2). It is important to reduce the size of potentials after combinations since at this step largest potentials are generated. Figure 4 shows size of all potentials stored in memory during the NHL ID (left) and IctNeo (right) evaluation using different heuristics and threshold values. For space restrictions only results using the thresholds values 0, 0.05 and 0.075 are shown. The vertical axis indicates the storage size using a logarithmic scale. The horizontal axis indicate the variable removed. It can be observed that when the exact prune is performed (threshold value 0), there is not relevant differences in the storage requirements. In fact, at some stages of the evaluation, MIN_WEIGHT and MIN_FILL_WEIGHT require less storage size. However, using a threshold for pruning greater than 0, specific heuristics for BTs (MIN_COMB_TREE and MIN_MARG_TREE) offer better results. When a high threshold value is used, the size of a BT is much smaller than the size of a table representing the same potential. Since MIN_WEIGHT and MIN_FILL_WEIGHT assume that potentials are represented with tables, it is not able to take advantage of such information.

The storage requirements can be analyzed in more detail with Tables 1 and 2, which show the mean and maximum storage requirements in number of nodes for evaluating NHL and IctNeo IDs respectively. Specific heuristics (MIN_COMB_TREE and MIN_MARG_TREE) require less storage size if an important prune is performed.

	Mean storage size			Maximum storage size		
	$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.075$	$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.075$
MIN_WEIGHT	$6.51 \cdot 10^4$	325	115	$4.03 \cdot 10^5$	2670	676
MIN_COMB_TREE	$8.33 \cdot 10^4$	192	89.2	$7.33 \cdot 10^5$	**1010**	**340**
MIN_MARG_TREE	$9.92 \cdot 10^4$	**190**	**86.8**	$1.61 \cdot 10^6$	**1010**	**340**
MIN_FILL_WEIGHT	$\mathbf{2.25 \cdot 10^4}$	508.15	225.15	$\mathbf{1.41 \cdot 10^5}$	2730	1360

Table 1. Mean and maximum storage requirements in number of nodes for the evaluation of the NHL ID

[1] `http://leo.ugr.es/~elvira`

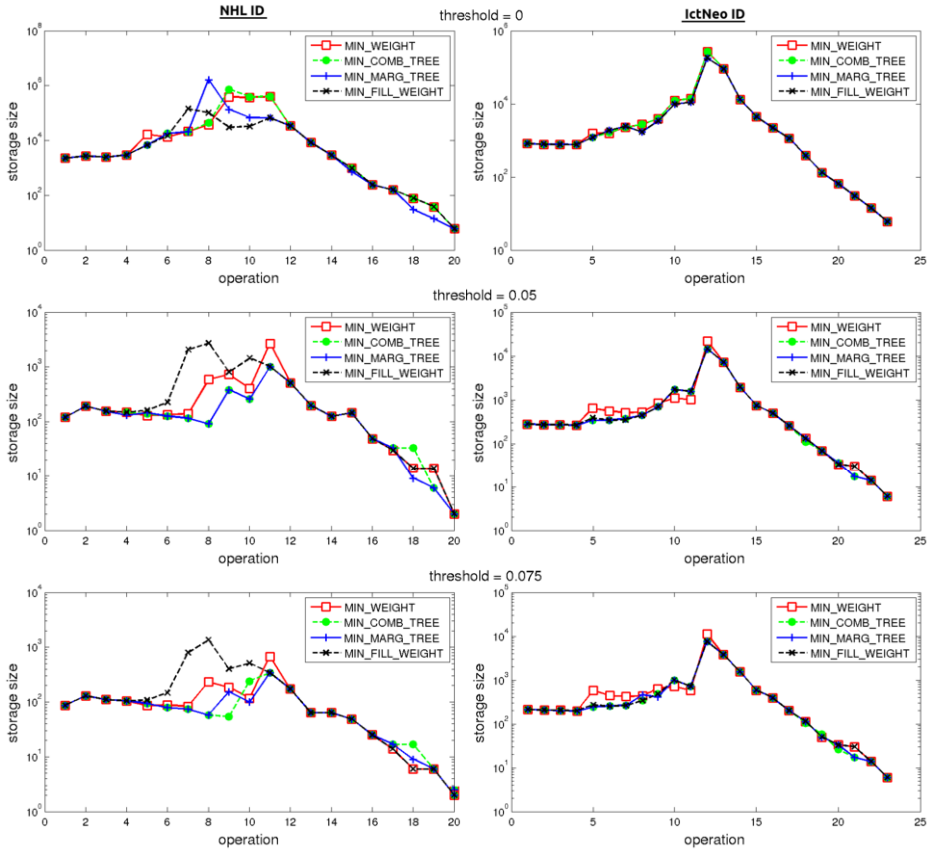


Figure 4. Size of all potentials stored in memory during the NHL ID (left) and IctNeo ID (right) evaluation comparing the heuristics *minimum weight*, *minumum combined tree*, *minumum marginalised* and *minimum fill-in arcs weight* using BTs and different threshold values.

	Mean storage size			Maximum storage size		
	$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.075$	$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.075$
MIN_WEIGHT	$1.84 \cdot 10^4$	1690	984	$2.7 \cdot 10^5$	$2.16 \cdot 10^4$	$1.13 \cdot 10^4$
MIN_COMB_TREE	$1.84 \cdot 10^4$	**1390**	**798**	$2.7 \cdot 10^5$	$\mathbf{1.44 \cdot 10^4}$	**7510**
MIN_MARG_TREE	$\mathbf{1.42 \cdot 10^4}$	**1390**	800	$\mathbf{1.8 \cdot 10^5}$	$\mathbf{1.44 \cdot 10^4}$	**7510**
MIN_FILL_WEIGHT	$\mathbf{1.42 \cdot 10^4}$	**1390**	800	$\mathbf{1.8 \cdot 10^5}$	$\mathbf{1.44 \cdot 10^4}$	**7510**

Table 2. Mean and maximum storage requirements in number of nodes for the evaluation of the IctNeo ID

4. Conclusions and Future Work

Finding an elimination ordering that minimises the size of potentials during the evaluation is an element of crucial importance for the efficiency of the VE algorithm. Some greedy algorithms use deterministic heuristics for choosing the next variable to remove. However, these heuristics are not appropriate if potentials are represented using BTs. In the present paper, two new heuristics that estimate the sizes of intermediate potentials during the evaluation are proposed. In the experimentation, it is shown that the use of these heuristics reduce the storage requirements. In particular, these heuristics offer the best results if high threshold for pruning the BTs are used.

As regards future directions of research, it could be interesting to find any heuristic for determining the ordering for combining potentials relevant during the removal of a variable. Finding an optimal ordering will reduce the complexity of operations and therefore the efficiency of the evaluation. Some methods such as *Symbolic Probabilistic Inference* (SPI) [15,11] reorder the combination and marginalization operations to reduce the complexity of computations. Thus the SPI algorithm could be adapted for working with BTs.

Acknowledgments

This research was supported by the Spanish Ministry of Economy and Competitiveness under project TIN2010-20900-C04-01, the European Regional Development Fund (FEDER) and the FPI scholarship programme (BES-2011-050604). The authors have been also partially supported by "Junta de Andalucía" under projects TIC-06016 and P08-TIC-03717.

References

[1] C Bielza, M Gómez, Ríos S. Insua, J. A. Fernández del Pozo, Barreno García P., S Caballero, and M. Sánchez Luna. Ictneo system for jaundice management. *Revista de la Real Academia de Ciencias Exactas, Físicas y Naturales*, 92(4):307–315, 1998.

[2] 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 AI*, pages 115–123. Morgan Kaufmann Publishers Inc., 1996.

[3] R. Cabañas de Paz, M. Gómez, and A. Cano. Approximate inference in influence diagrams using binary trees. In *Proceedings of the 6th European Workshop on Probabilistic Graphical Models, PGM 2012*, pages 43–50, 2011.

[4] A. Cano and S. Moral. Heuristic algorithms for the triangulation of graphs. *Advances in Intelligent Computing—IPMU'94*, pages 98–107, 1995.

[5] Hugin Expert. Hugin api–reference manual, version 7.0. *Hugin Expert A/S*, 2008.

[6] M. Gómez and A. Cano. Applying numerical trees to evaluate asymmetric decision problems. *ESCQARU*, pages 196–207, 2003.

[7] R. A. Howard and J. E. Matheson. Influence diagram retrospective. *Decision Analysis*, 2(3):144–147, 2005.

[8] F. Jensen, F. V. Jensen, and S. L. Dittmer. From Influence Diagrams to junction trees. In *Proceedings of the Tenth international conference on Uncertainty in artificial intelligence*, pages 367–373. Morgan Kaufmann Publishers Inc., 1994.

[9] F.V. Jensen and T.D. Nielsen. *Bayesian networks and decision graphs*. Springer Verlag, 2007.

[10] U. Kjærulff. Triangulation of graphs–algorithms giving small total state space. 1990.

[11] Z. Li and B. d'Ambrosio. Efficient inference in bayes networks as a combinatorial optimization problem. *International Journal of Approximate Reasoning*, 11(1):55–81, 1994.

[12] P.J.F. Lucas and B. Taal. Computer-based decision support in the management of primary gastric non-hodgkin lymphoma. *UU-CS*, (1998-33), 1998.

[13] A.L. Madsen and F.V. Jensen. Lazy evaluation of symmetric Bayesian decision problems. In *Proceedings of the 15th Conference on Uncertainty in AI*, pages 382–390. Morgan Kaufmann Publishers Inc., 1999.

[14] D.J. Rose. A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. *Graph theory and computing*, 183:217, 1972.

[15] R. D. Shachter, B. D'Ambrosio, and B. Del Favero. Symbolic probabilistic inference in belief networks. In *AAAI*, volume 90, pages 126–131, 1990.

[16] R.D. Shachter. Evaluating influence diagrams. *Operations research*, pages 871–882, 1986.

[17] S.K.M. Wong and C.J. Butz. Contextual weak independence in Bayesian networks. In *Proceedings of the 15th conference on Uncertainty in AI*, pages 670–679. Morgan Kaufmann Publishers Inc., 1999.

[18] N.L. Zhang and D. Poole. Exploiting causal independence in Bayesian network inference. *Journal of Artificial Intelligence Research*, 5:301–328, 1996.